

<목차>

2021년 1월 18일 월요일 오전 9:52

단원	번호	상세 목차	수업내용
R기본문법	1	R이란 무엇인가?	
	2	R 설치 및 R studio 설치	
	3	R 의 자료구조	바로가기
	4	SQL 과 R 과 비교 (연산자)	
	5	SQL 과 R 과 비교 (함수)	
	6	SQL 과 R 과 비교 (그룹함수)	
	7	SQL 과 R 과 비교 (조인)	바로가기
	8	SQL과 R 과 비교 (집계결과)	바로가기
	9	SQL과 R 과 비교 (집합연산자)	바로가기
	10	SQL 과 R 과 비교 (서브쿼리)	바로가기
	11	SQL 과 R 과 비교 (순위출력)	바로가기
	12	R 샤이니 사용해서 분석결과 자동화 하기	바로가기
	13	막대 그래프 그리기	바로가기
	14	원형 그래프 그리기	바로가기
	15	라인 그래프 그리기	바로가기
	16	사분위수 그래프 그리기	바로가기
	17	히스토그램 그래프 그리기	바로가기
	18	특수 그래프 그리기	바로가기
	19	R 샤이니 이용해서 데이터 시각화 자동화 하기	바로가기
	20	데이터 시각화 자동화 코드를 이해하기 위한 5가지 단계	바로가기
	21	데이터 시각화 화면을 홈페이지로 만들기	바로가기
	22	데이터 분석을 편하게... (R샤이니 자동화 스크립트 개선1)	바로가기
단원	번호	상세목차	수업내용
1장 기계학습 소개	1	머신러닝 이란?	바로가기
	2	머신러닝의 종류 3가지는 ?	

	3	데이터 분석을 편하게 (R에서 수행하는 자동화 스크립트)	바로가기
단원	번호	상세목차	수업내용
2장 데이터 관리와 이해	1	R의 자료구조의 종류	바로가기
	2	데이터의 전반적인 관찰(평균,중앙,최빈,표준편차,분산)	바로가기
	3	수치형 데이터 살펴보기(히스토그램, 정규분포)	
	4	범주형 데이터 살펴보기(산포도 그래프)	바로가기
	5	CrossTable (이원교차표)	바로가기
	6	R에서의 if 문과 loop 사용법	바로가기
	7	데이터 분석을 편하게... (샤이니 자동화 스크립트 개선2)	바로가기
단원	번호	상세목차	수업내용
3장 knn 알고리즘	1	knn 알고리즘 이란?	바로가기
	2	knn 알고리즘 수학식 이해	바로가기
	3	knn 알고리즘의 원리를 코드로 이해	바로1 , 바로2
	4	유방암 데이터로 knn 알고리즘 실습	바로가기
	5	적절한 k 값을 찾기 위한 시각화	바로가기
	6	my_knn 함수를 날 코딩으로 직접 만들기	바로가기
	7	데이터 분석을 편하게... (샤이니 자동화 스크립트 개선3)	바로가기
단원	번호	상세목차	수업내용
4장 나이브베이즈	1	나이브 베이즈 알고리즘이란 ?	바로가기
	2	나이브 베이즈 수학식 이해	바로가기
	3	나이브 베이즈 원리를 코드로 이해	바로가기
	4	독버섯과 정상버섯을 컴퓨터가 분류할 수 있을까 ?	바로가기
	5	선호하는 영화 장르를 컴퓨터가 알아 맞힐 수 있을까?	바로가기
	6	적절한 laplace 값을 알아내기 위한 시각화	바로가기
	7	데이터 분석을 편하게... (나이브 베이즈 활용 함수 생성)	바로가기
	8	데이터 분석을 편하게... (자동화 스크립트 개선4)	바로가기
		데이터 분석을 편하게... (확률로 나오게)	바로가기
	번호	상세목차	수업내용

단원			
5장 의사결정트리	1	의사결정트리란 ?	바로가기
	2	엔트로피와 정보획득량	바로가기
	3	엔트로피와 정보획득량 수학식의 이해1	바로가기
	4	엔트로피와 정보획득량 수학식의 이해2	바로가기
	5	화장품을 구매할것으로 예상되는 고객은?	바로가기
	6	은행 대출 채무를 불이행할 것 같은 고객이 누구인가 ?	바로가기
	7	파생변수 생성의 중요성	바로가기
	8	의사결정트리 자동화 스크립트 구현	바로가기
	9	적절한 trials 값을 알아내기 위한 시각화	바로가기
	10	규칙 기반 알고리즘이란 ?	바로가기
	11	oneR 알고리즘으로 독버섯 분류 실습	바로가기
	12	Jriper 알고리즘으로 독버섯 분류 실습	바로가기
	13	데이터 분석을 편하게... (머신러닝 모델 활용 스크립트)	
단원	번호	상세목차	수업내용
6장 회귀분석	1	단순 선형 회귀 분석 이론	바로가기
	2	단순 선형 회귀 분석 수학식의 이해	바로가기
	3	탄닌 함유량과 애벌래 성장간의 관계	바로가기
	4	우주 왕복선 챌린저호 폭발 원인 분석	바로가기
	5	코스피 지수 수익률과 삼성, 현대 주식 수익률 상관관계	바로가기
	6	다중 선형 회귀 이론	바로가기
	7	다중 선형 회귀 분석 수학식의 이해	바로가기
	8	스마트폰 만족에 미치는 영향도 분석	바로가기
	9	미국 대학 입학에 가장 영향이 높은 과목 분석	바로가기
	10	보험회사의 보험료 산정에 미치는 요소 분석	바로가기
	11	회귀트리와 모델트리 이론	바로가기
	12	회귀 트리로 와인 품질 측정 분류기 생성 하는 방법	바로가기
	13	모델 트리로 와인 품질 측정 분류기 생성하는 방법	바로가기
	14	모델 트리로 보스톤 지역의 집값 예측하는 모델 생성	바로가기
	15	오라클에서 바로 SQL로 회귀 분석 구현하는 방법	바로가기
	16	R 마크다운 사용하여 분석 보고서 만들기	바로가기
	17	데이터 분석을 편하게... (머신러닝 모델 활용 스크립트)	바로가기

단원	번호	상세목차	수업내용
7장 신경망	1	신경망이란 ?	바로가기
	2	퍼셉트론	바로가기
	3	활성화 함수 소개	
	4	콘크리트 강도를 예측하는 신경망 모델 생성	바로가기
	5	와인의 품질을 분류하는 신경망 모델 생성	바로가기
	6	보스톤의 집값을 예측하는 신경망 모델 생성	바로가기
	7	1957년에 나온 퍼셉트론을 컴퓨터로 구현하기	바로가기
	8	신경망을 활용한 데이터 분석 R 마크다운으로 정리	바로가기

단원	번호	상세목차	수업내용
8장 연관규칙	1	쿠팡의 사례	바로가기
	2	Apriori 알고리즘 이란 ?	
	3	맥주와 기저귀 사례 테스트	바로가기
	4	보습학원이 있는 건물에 가장 많이 있는 매장 업종은 ?	바로가기
	5	영화 라라랜드의 긍정적 평가와 부정적 평가의 연관분석	바로가기

단원	번호	상세목차	수업내용
9장 k-means	1	k-means 이론	바로가기
	2	k-means 기본 실습	바로가기 바로가기
	3	영어, 수학 점수로 학생 분류	바로가기
	4	SNS 의 글로 같은 성향을 가진 사람들을 분류	바로가기

유방암 데이터를 kmeans 로 테스트 : [바로가기](#)

k-means 함수 날코딩으로 구현하기 : [바로가기](#)

단원	번호	상세목차	수업내용
10장	1	흔동행렬을 사용한 성능척도	

성능평가		
	2	카파통계량
	3	민감도와 특이도
	4	정밀도와 재현율
	5	성능 트레이드 오프 시각화(Roc 곡선)

단원	번호	상세목차	수업내용
11장 성능 개선	1	K-foldout	바로가기
	2	caret 패키지를 이용한 모델 파라미터 자동 튜닝	
	3	앙상블 기법 - bagging	
	4	앙상블 기법 - boosting	

01/14

2021년 1월 14일 목요일 오후 3:54

■ R을 이용한 머신러닝 수업

* R 설치

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

관리자 권한으로 실행, 편집 --> GUI 설정에서 글씨크기 바꾸기
emp3.csv를 R로 로드합니다.

■ 작업 디렉토리 설정 및 emp.csv 로드 방법

- setwd("d:\data")
- getwd()

emp3.csv를 d 드라이브 밑에 data 폴더 밑에 둔다.

- emp <- read.csv("emp3.csv", header=T)
- emp

emp 데이터의 월급으로 원형 그래프를 그리시오!

- pie(emp\$sal, col=rainbow(14))

14개의 색깔을 보여달라!!

* R studio 설치

구글에서 R studio 설치라고 검색하고 설치하기!

- **리눅스와 하둡 제출물**

1. 노트정리(오늘까지 조장에게 제출)
2. 리눅스와 하둡 포트폴리오 (NCS 제출물) 금요일 저녁 6시전까지 게시판에 제출

데이터 분석가가 갖추어야 할 소양 : 파이썬, R, 하둡, SQL -----> 1월, 2월

딥러닝 개발자, 인공지능 개발자 : 파이썬, 딥러닝, 리눅스, 마리아 디비 ---> 2월, 3월

1. 리눅스와 하둡 포트폴리오
 2. R은 포트폴리오 또는 케글 순위
 3. 빅데이터 기사 시험: 4월 17일 (이론, 실기)
-

■ R을 통해서 배워야 하는 큰 그림

1. 탐색적 데이터 분석 : 현상에서 패턴을 발견하는 것 (하둡, 리눅스 포트폴리오)
2. 통계적 추론 : 현상에서 인과적인 결론을 도출하는 것 (통계 스터디: 검정, 추정)
3. 기계학습 (머신러닝) : 현상을 예측하는 것 (R 수업, 케글 상위권 도전)

케글(kaggle) ?

전세계 데이터 분석가들이 특정 데이터 분석 주제를 두고 선의의 경쟁을 해서 좋은 데이터 분석에 대해서 순위를 부여하고 상금도 주고 취업할 때 스카우트 제의도 하는 구글에 인수된 회사

■ R이란 무엇인가?

뉴질랜드의 auckland 대학의 robert gentleman과 Ross Ihaka가 1995년에 개발한 소프트웨어. 데이터 분석을 위한 통계 및 시각화를 지원하는 무료 소프트웨어.

파이썬과 비교했을 때의 장점?

R이 데이터 시각화 했을 때 더 예쁘게 나옴.

■ R을 사용해야 하는 이유?

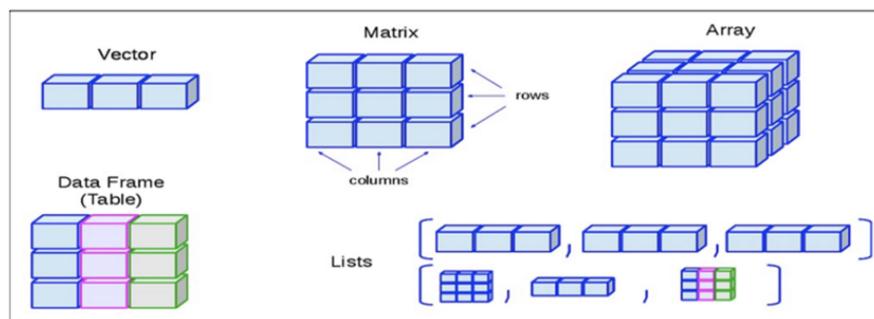
1. R은 무료다.
2. data 분석을 위해 가장 많이 사용하는 통합 플랫폼임.
3. 복잡한 데이터를 다양한 그래프로 표현 가능함.
4. 분석을 위한 데이터를 쉽게 저장 및 조작 가능함.
5. 누그든지 유용한 패키지를 생성해 공유 가능하고 새로운 기능에 대한 전파가 빠름
6. 어떠한 os에도 설치가 가능. (아이폰에도 설치 가능함)

구글의 코랩을 이용하면 GPU를 사용해서 파이썬 코드를 무료로 수행할 수 있는데 코랩에서 R코드도 지원함.

■ R의 자료구조

1. vector : 같은 데이터 타입을 갖는 1차원 배열 구조
 2. matrix : 같은 데이터 타입을 갖는 2차원 배열 구조
 3. array : 같은 데이터 타입을 갖는 다차원 배열 구조
 4. data frame : 각각의 데이터 타입을 갖는 컬럼으로 이루어진 2차원 배열 구조
예 : RDBMS의 테이블과 유사함
 5. list : 서로 다른 데이터 구조 (vector, data frame, matrix, array) 인 데이터 타입이 중첩된 구조
- [빅데이터 기사시험 이론 문제] : data frame의 정의

■ R 의 자료구조 그림



■ R을 사용하여 기본 데이터 검색

문제2~문제8 참조

■ 연산자 총정리

1. 산술 연산자 : * / + -
2. 비교 연산자 : > , < , >=, <=, ==, !=
3. 논리 연산자 : & : and (벡터화 된 연산)
 && : and (벡터화 되지 않은 연산)
 | : or (벡터화 된 연산)
 || : or (벡터화 되지 않은 연산)
 ! : not

예제1 : 벡터화 된 연산

```
x <- c(1,2,3)
x
(x > c(1,1,1) ) & (x < c(3,3,3) )
```

결과:

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

예제2 : 벡터화 되지 않은 연산

```
x <- 1
x
( (x>-2) && (x<2) )
```

결과:

```
[1] TRUE
```

설명 : `combine`을 사용하면 벡터화 된 것입니다.

■ 연결 연산자

문제9 참조

연결연산자가 좀 더 예쁘게 나오게 하는 방법:

좀 더 예쁘게 나오게 하는법:

1. `data.table` 패키지를 설치한다.

```
install.packages("data.table")
```

2. `data.table`을 사용하겠다고 지정한다.

```
library(data.table)
```

3. `data.table`을 사용한다.

```
data.table(emp$ename, '의 직업은', emp$job)
```

■ 기타 비교 연산자

오라클 -----R
1. in %in%
2. like grep
3. is null is.na
4. between .. and emp\$sal >= 1000 & emp\$sal <= 3000

■ 중복제거

오라클 -----R
distinct unique

■ 정렬 작업

오라클 -----R
order by
1. data frame에서 order 옵션
2. doBy 패키지를 설치하고 orderBy 함수를 사용

■ 현재 R에서 사용되고 있는 result와 같은 변수의 목록 확인 방법

ls()

■ 변수를 지우고 싶다면?

rm(result)
result

■ 문자함수

오라클 -----R
upper toupper
lower tolower
substr substr
replace gsub

■ 숫자함수

오라클 -----R
round round

trunc	trunc
mod	%%
power	2^3 (2의 3승)

■ 날짜함수

오라클 ----- R

sysdate	Sys.Date()
add_months	difftime
months_between	내장함수 없음
last_day	내장함수 없음
next_day	내장함수 없음

■ 변환함수

오라클 ----- R

to_char	as.character
to_number	as.integer
to_date	as.Date

예:

```
SQL> select ename, to_char( hiredate, 'day')
   from emp;
```

```
R> data.table( emp$ename, format( as.Date(emp$hiredate), '%A' ) )
```

설명 : format의 옵션 : %A : 요일

- %Y : 년도 4자리
- %y : 년도 2자리
- %m : 달
- %d : 일

■ 변환함수

오라클 ----- R

nvl 함수	is.na
decode 함수	ifelse
case 함수	ifelse

데이터 분석가

si	vs	sm
데이터 분석을 위한 시스템 구축 하면서 수시로		개발해 놓은 분석 시스템을 유지보수
데이터 분석을 위한 큰 툴을 마련		요청되는 데이터 분석
데이터 분석가, 데이터 엔지니어, dba 프리랜서, 회사 정규직		데이터 분석가, 데이터 엔지니어 프리랜서, 회사 소속

분석을 위한 언어 선택을 처음에 결정:

SQL, R, 파이썬

R을 배우는 이유 - 이 수업 끝날때쯤 R을 활용해서 이루어야 할 목표:

케글 상위권도전
빅데이터 기사시험 필기, 실기 따기

■ 그룹함수

Oracle	vs	R
1. max		max
2. min		min
3. sum		sum
4. avg		mean
5. count		length(세로) table(가로)

■ SQL과 R 비교 (조인)

Oracle	vs	R
equi join		
non equi join		merge
outer join		
self join		

dept.csv를 내려받아 dept라는 변수에 로드하시오!

■ R 수업 복습

1. R을 배워야 하는 이유?

(작은 목표) : 케글에서 상위권 도전

순위: 쉬운 것 ---> 어려운 것 ---> 상금 있는 것

- 이력서: 케글의 competition 상위 몇%입니다 (증명서 캡쳐) 라고 넣기
- 빅데이터 기사 시험 합격

(최종 목표) : 데이터 분석가, 딥러닝 개발자

2. R 기본 문법 시작화

■ 집합 연산자

오라클	vs	R
1. Union all		rbind
2. union		rbind + unique
3. intersect		intersect
4. minus		setdiff

Version 1.3.1093

- R을 켈 때마다 emp 데이터 프레임을 구성하는 작업을 자동으로 되게하는 방법
- C:\Program Files\R\R-4.0.3\etc

매모장을 관리자 권한으로 연다.

열기 들어가서

C:\Program Files\R\R-4.0.3\etc 의 Rprofile.site에서
setwd("d:\data")
emp <- read.csv("emp3.csv")
dept <- read.csv("dept.csv")

union의 union all과의 차이점? 1. 중복된 데이터를 제거
2. 데이터를 정렬한다.

```
SQL> select ename, sal, deptno
      from emp
      where deptno in (10, 20)
      union
      select ename, sal, deptno
```

```
from emp  
where deptno = 20;
```

```
R> unique( rbind(emp[ emp$deptno %in% c(10,20), c("ename", "sal", "deptno")],  
emp[ emp$deptno ==20, c("ename", "sal", "deptno") ]))
```

oracle과 같이 정렬하려면?

```
x <- unique( rbind(emp[ emp$deptno %in% c(10,20), c("ename", "sal", "deptno")],  
emp[ emp$deptno ==20, c("ename", "sal", "deptno") ]))  
library(doBy)  
orderBy(~ename, x)
```

■ SQL의 서브쿼리를 R로 구현하기

* 오라클의 서브쿼리의 종류 3가지

1. 단일행 서브쿼리
2. 다중행 서브쿼리
3. 다중 컬럼 서브쿼리

■ R에서 순위를 출력하는 방법

문법 : rank 함수

예제: 이름, 월급, 월급에 대한 순위를 출력하시오!

```
data.table( 이름=emp$ename, 월급=emp$sal, 순위=rank(-emp$sal, ties.method="min"))
```

<결과>

이름 월급 순위		
1:	KING	5000 1
2:	BLAKE	2850 5
3:	CLARK	2450 6
4:	JONES	2975 4
5:	MARTIN	1250 10
6:	ALLEN	1600 7
7:	TURNER	1500 8
8:	JAMES	950 13
9:	WARD	1250 10
10:	FORD	3000 2
11:	SMITH	800 14
12:	SCOTT	3000 2
13:	ADAMS	1100 12
14:	MILLER	1300 9

설명 : rank에 마이너스(-)를 사용하면 월급이 높은것부터 출력됩니다.

ties.method 옵션: 1. min : 오라클의 rank와 같다.

2. first : 오라클의 rank와 같은데 순위가 같은 데이터가 있으면 인덱스 순서가 먼저 나온

데이터를 높은 순위로 부여합니다.

3. max : 2등이 두명이면 둘다 3등으로 출력합니다.

(같은 등수 여러명이면 해당 등수 건너뛰고 그 다음 등수로 출력)

```
> x <- data.table( 이름=emp$ename, 월급=emp$sal, 순위=rank(-emp$sal,  
ties.method="min") )
```

```
> library(doBy)
```

```
> orderBy(~순위, x)
```

결과:

	이름	월급	순위
1:	KING	5000	1
2:	FORD	3000	2
3:	SCOTT	3000	2
4:	JONES	2975	4
5:	BLAKE	2850	5
6:	CLARK	2450	6
7:	ALLEN	1600	7
8:	TURNER	1500	8
9:	MILLER	1300	9
10:	MARTIN	1250	10
11:	WARD	1250	10
12:	ADAMS	1100	12
13:	JAMES	950	13
14:	SMITH	800	14

```
> x <- data.table( 이름=emp$ename, 월급=emp$sal, 순위=rank(-emp$sal,  
ties.method="max") )  
> orderBy(~순위, x)
```

	이름	월급	순위
1:	KING	5000	1
2:	FORD	3000	3
3:	SCOTT	3000	3
4:	JONES	2975	4
5:	BLAKE	2850	5
6:	CLARK	2450	6
7:	ALLEN	1600	7
8:	TURNER	1500	8

```
9: MILLER 1300  9  
10: MARTIN 1250 11  
11: WARD 1250  11  
12: ADAMS 1100 12  
13: JAMES  950  13  
14: SMITH  800  14
```

※ 오라클의 **dense_rank**와 같은 함수는 무엇인가?

```
SQL> select ename, sal, dense_rank() over (order by sal desc) 순위  
      from emp;
```

```
R> library(dplyr)  
> x <- data.table( 이름=emp$ename, 월급=emp$sal, 순위=dense_rank(-emp$sal) )  
  
R> library(doBy)  
> orderBy(~순위, x )
```

■ R로 막대 그래프 그리기

" 머신러닝 수업때와 케글에 데이터 분석 시 사용하기 편하도록 함수로 그래프 그리는 것을 자동화 시키기"

문제127번 ~ 문제137번

R 함수 생성 문법:

함수명 <- function(){ 함수코드 }

실행: 함수명 적기

■ R로 원형 그래프 그리기

문제138번~

■ R로 라인 그래프 그리기

시간 순서에 따른 데이터의 변화를 볼 때 유용한 그래프

■ emp3.csv를 R studio 켜면 자동으로 로드 되게끔 하는 방법:

C:\Program Files\R\R-4.0.3\etc 의 Rprofile.site를 메모장을 통해 연다.

메모장을 관리자 권한으로 실행하고 C:\Program Files\R\R-자신의버전번호\etc\Rprofile.site 파일을 엽니다.

그리고 맨 아래줄에 다음의 내용을 추가하고 저장한 다음 Rstudio 를 실행합니다.

```
library(utils)
setwd("d://data")
emp <-read.csv("emp3.csv")
dept <-read.csv("dept.csv")
library(dplyr)
library(data.table)
```

■ 어제 배웠던 내용

R에서 데이터 시각화

1. 막대그래프
2. 원형 그래프
3. 라인 그래프

현업에서도, 머신러닝 수업때도, 빅데이터 시험때도 시각화를 해야함.

그래서 자동화 스크립트를 생성하는 것이 좋다.

■ R에서 히스토그램 그래프 그리기

빅데이터 기사시험의 히스토그램 그래프의 정의?

히스토그램은 하나의 속성에 대한 데이터의 분포를 시각적으로 표현하는 그래프입니다.

- 수제비 2권 3-9 참고

예제 : 머신러닝 R책의 2장의 히스토그램 그래프 설명의 예제

```
> usedcars <- read.csv("usedcars.csv")
> View(usedcars)
```

1. 전체 건수를 확인합니다.

```
nrow(usedcars)
```

2. 컬럼이 몇개인지 확인합니다.

```
ncol(usedcars)
```

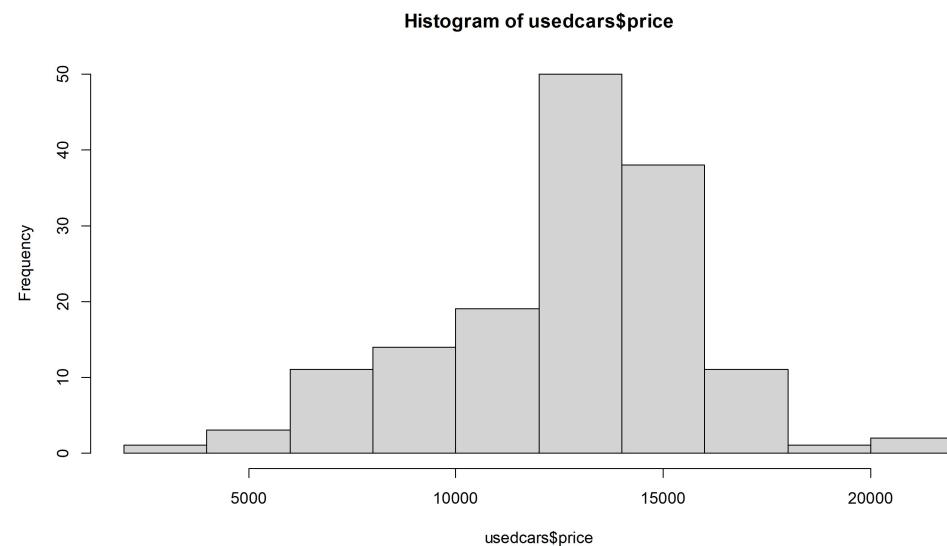
결과: 6

3. 각 컬럼들의 데이터에 대한 통계정보를 확인하시오.

```
summary(usedcars)
```

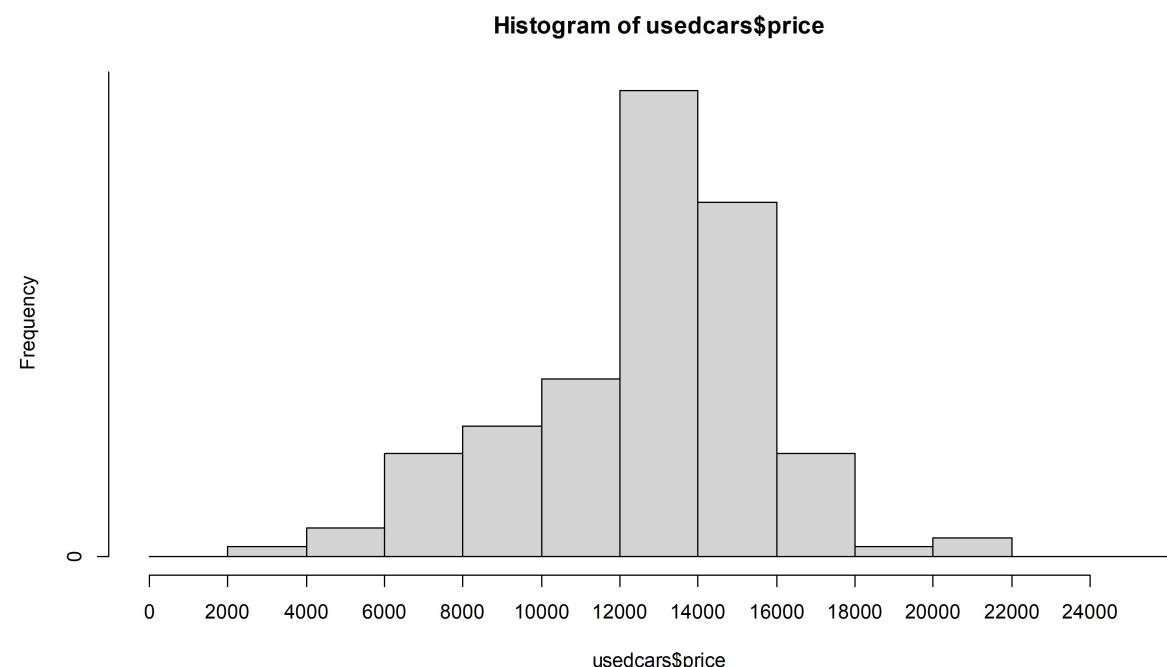
4. 중고차 가격에 대한 히스토그램 그래프를 그리시오!

```
hist(usedcars$price)
```



5. 히스토그램 x축의 간격이 좀 더 이해하기 쉽게 나올 수 있도록 그래프를 조정하시오!

```
hist(usedcars$price, at=seq(0, 24000, by=2000), breaks= seq(0, 26000, by=2000) )
```



■ 사분위수 그래프 (박스 플롯 그래프)

박스플롯은 많은 데이터를 그림을 이용하여 집합의 범위와 중앙값을 빠르게 확인할 수 있으며 또한 통계적으로 이상치값이 있는지 빠르게 확인이 가능한 시각화 기법입니다.

- 빅데이터 기사시험 수제비 2권 3-9페이지

평균값과 중앙값과 최빈값만으로는 데이터 분석을 하기 부족한 경우가 있다.

평균 데이터는 데이터의 중심이 어디쯤인지 알려주지만 특정 데이터가 평균을 중심으로 어떻게 분포가 되어있는지는 알려주지 않는다.

예제: 어느 농구단의 감독이 아래의 3명의 농구선수 중 한명을 선택하려고 한다.

아래의 3명의 선수의 게임별 점수를 가지고 한명을 고른다면 어떤 선수를 골라야 하는가?

② 농구 선수 3명이 각각의 게임당 득점한 점수

```
x1 <- c(7,8,9,9,10,10,11,11,12,13)  
x2 <- c(7,9,9,10,10,10,10,11,11,13 )  
x3 <- c(1,1,7,7,10,10,10,11,13,30 )
```

■ 데이터의 분포를 확인하는 방법

1. 범위
2. 사분위수 범위
3. 히스토그램 그래프

■ 범위

데이터의 폭을 확인할 때 사용합니다.

```
range(x1)  
range(x2)  
range(x3)
```

범위는 그 자체로는 데이터의 폭만 설명할 뿐 그 안에서 데이터가 분포되는 방식은 설명해주지 않는다.

특히 이상치에 민감합니다.

3번째 선수의 경우 어쩌다 한번 잘한 게임(30점)인 이상치 때문에 범위가 넓어져 버렸다. 그래서 분석하기가 어려워졌다. 그러므로 이상치로부터 멀어질 필요가 있다. 이상치로부터 멀어지고 가운데 있는 데이터에만 집중하게 해주는게 사분위수 범위이다.

선배기수 : mc365에서 데이터 분석 요청이 들어와서 데이터 분석을 하고 시각화와 설명을 함께 문서로 만들어서 가져가 발표

* 데이터를 통계치로 분석하는 순서

평균값 ---> 중앙값 ---> 최빈값 ---> 범위 ---> 사분위수 범위 ---> 분산값---> 표준편차

mean median range boxplot var std

※ IQR ?

InterQuartile Range , Q1과 Q3의 차이인 사분위수 범위

IQR(usedcars\$price) # 3909.5

14904.5 - 10995.0 = 3909.5

- 빅데이터 기사시험 수제비 1권 2-12 참조

사분위수 범위는 자료들의 중간 50%에 포함되는 자료의 산포도를 나타냅니다.

사분위수 범위는 1사분위수와 3사분위수 사이의 차이입니다.

■ 그래프

1. 막대그래프
2. 원형그래프
3. 라인 그래프
4. 히스토그램 그래프
5. 사분위 그래프(박스 그래프)
6. 지도 그래프
7. 워드 클라우드 그래프
8. 소리를 시각화

■ 지도 그래프

map 패키지를 설치하고 중국지도만 확대해서 출력하시오!

```
install.packages("maps")
install.packages("mapproj")
library(maps)
library(mapproj)

map("world")
```

■ 워드 클라우드 그리기

감정분석을 위한 시각화로 워드 클라우드를 사용합니다.

※ R 로 워드클라우드를 그리려면 R java 를 설치해야합니다.

1. 아래의 사이트에서 java 64비트를 다운로드 받습니다.

<http://www.java.com/en/download/manual.jsp>

2. 자바 설치시 대상 폴더 변경으로 설치

3. 아래와 같이 환경 설정을 합니다. (Rstudio에서)

```
Sys.setenv(JAVA_HOME="C:\Program Files\Java\jre1.8.0_281")
```

4. R java 를 설치합니다.

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

```
install.packages("KoNLP")
install.packages("wordcloud")
install.packages("plyr")
```

```
library(KoNLP)
library(wordcloud)
library(plyr)
```

```
useSejongDic() # 세종 사전에 있는 한글을 R로 로드하는 명령어
```

```
setwd("d:\data") # 워킹디렉토리를 소환
winter <- readLines('winter.txt')
```

```
nouns <- extractNoun(winter) # 명사 단어만 추출
nouns <- unlist(nouns)
nouns <- nouns[nchar(nouns)>=2] # 단어중에 2자 이상인것만
cnouns <- count(nouns) # 단어별 건수를 출력한다.
```

색깔 추가

```
pal <- brewer.pal(6,"Dark2")
pal <- pal[-(1)]
```

글씨 폰트 설정

```
windowsFonts(malgun=windowsFont("맑은 고딕"))
```

```
wordcloud( words=cnouns$x, # 단어
            freq=cnouns$freq, # 단어 빈도수
            colors=pal, # 색깔
            min.freq=3, # 빈도수가 3개 이상인것만 시각화
```

```
random.order=F,      # F 로 하게 되면 큰글씨부터 출력이 되면서  
family="malgun")    # 중앙에서 퍼지게 한다.  
                      # 맑음 글씨체로 시각화 하겠다.
```

■ 어제까지 배운 R수업 내용

1. R을 왜 배워야하는지?
2. R 설치 --> window, linux
3. R 기본 문법 (데이터 검색)
4. R로 시각화 하기
 - 막대 그래프
 - 원형 그래프
 - 라인 그래프
 - 히스토그램 그래프
 - 사분위수 그래프
 - 워드 클라우드 그래프
 - 지도 그래프

위의 스크립트들을 잘 정리해서 사용하기 편하도록 자동화 스크립트로 생성

■ 데이터 시각화 자동화 스크립트를 R studio를 결때마다 편하게 사용할 수 있게 하는 방법

1. 그래프 3개 자동화 스크립트를 가져온다. (my_func 함수)
2. 그래프 자동화 기본 스크립트를 쉽게 사용 하는 방법

<https://cafe.daum.net/oracleoracle/SZTZ/1990> 참고

3. 더 쉽게 실행하려면?

```
source('my_func2.R')
r<- function(){source('my_func2.R')}
r()
```

4. Rprofile.site에 추가해서 R 결때마다 자동으로 불러오게 하려면?
메모장을 관리자 권한으로 열고 아래의 위치로 가서 Rprofile.site를 연다.

C:\Program Files\R\R-4.0.3\etc

5. 어제는 3개의 그래프만 추가했는데 어제 그렸던 히스토그램 그래프와 사분위수 그래프 my_func2.R에 추가합니다.

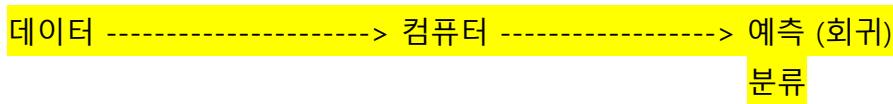
▣ 데이터 분석 모형

A. 통계기반 분석 모형

1. 기술 통계 : 평균, 분산, 표준편차, 왜도 첨도, 빈도 등에 대한 대표적인 통계적 수치를 가지고 분석
2. 상관 분석 : 둘 또는 여러 변수 사이의 연관관계를 분석
3. 회귀 분석 : 하나 이상의 독립변수들이 종속변수에 미치는 영향을 추정할 수 있는 통계 기법
4. 분산 분석 : 두개 이상의 집단간 비교를 수행할 때 집단 내의 분산의 비교로 얻은 분포를 이용하여 가설
 검정을 수행하는 방법
5. 주성분 분석 : 많은 변수의 분산 방식의 패턴을 간결하게 표현하는 주성분 변수를 원래 변수의 선형 결합
 으로 추출하는 통계기법
6. 판별 분석 : 집단에 대한 정보로부터 집단을 구별할 수 있는 판별 규칙을 만들고 다변량 기법으로 조사된
 집단에 대한 정보를 활용하여 새로운 개체가 어떤 집단인지 탐색하는 통계 기법

B. 데이터 마이닝 기반 분석 모형

C. 머신러닝 기반 분석 모형



- **지도학습**: 지도학습은 정답인 레이블(label)이 포함되어 있는 학습 데이터를 통해 컴퓨터를 학습시키는 방법

예제: knn(3장), 나이브 베이즈(4장), 의사결정트리와 랜덤포레스트(5장),
회귀분석(6장), 신경망과 서포트벡터머신(7장), 연관분석(8장)

- **비지도 학습**: 입력 데이터에 대한 정답인 레이블(label)이 없는 상태에서 데이터가 어떻게 구성되었는지 알아내는 기계학습 방법

예제 : k-means(9장)

- **강화학습**: 환경에 대해 적응해 나가는 경험 데이터를 쌓으면서 학습해 나가는 기

예제: 딥마인드의 알파고

★ 우리의 목표 : 케글 데이터 분석 세계대회에서 전 세계인들과 겨루 상위권 도전

■ R을 활용한 머신러닝 2장

기본책을 여러 번 반복해서 읽고 머리속에 체계화하기!!

1장 : 기계학습에 대한 소개

2장 : 3장부터 소개될 머신러닝 데이터 분석시 필요한 R 명령어와 그래프, 함수 소개

3장 : knn 머신러닝 ...

머신러닝 데이터 분석을 위한 R 기본 문법, 그래프, 함수를 소개

1. Factor가 무엇?
2. 이원 교차표 (CrossTable)
3. 데이터를 R로 불러오는 여러가지 방법

■ 1. Factor가 무엇?

• R의 자료구조

1. vector : 같은 데이터 탑입을 갖는 1차원 배열구조
예: c() 함수를 이용해서 구성할 수 있다.

```
a <- c( 1, 2, 3, 4, 5 )
str(a)
```
2. matrix : 같은 데이터 탑입을 갖는 2차원 배열구조
3. array : 같은 데이터 탑입을 갖는 다차원 배열구조
4. data frame : rdbms의 테이블과 같이 행과 열로 이루어진 자료구조
5. list : 서로 다른 데이터 구조의 중첩된 구조

▣ factor (책 69 페이지)

팩터:

범주나 순위 변수를 나타내기 위해 사용하는 특별한 종류의 벡터(vector)이다.

머신러닝 데이터 분석 시 factor를 알아야 하는 이유?

순위 데이터를 모델링하는 머신러닝 알고리즘은 순서 팩터를 기대하기 때문

예제: 팩터 = 일반 벡터 + level

예 : a <- c("middle", "low", "high") #벡터를 생성함

```
> a  
"middle" "low" "high"  
  
> str(a)  
chr [1:3] "middle" "low" "high" (벡터)  
  
> a2 <-factor(a)  
> str(a2)  
Factor w/ 3 levels "high","low","middle": 3 2 1
```

설명 : 팩터는 벡터와는 다르게 순서라는 개념이 들어가 있는데 위의 a2의 팩터의 경우 순서가 알파벳 순서, abcd 순서로 순서의 개념이 들어가 있습니다.

```
order(a2, decreasing=F)  
> a2[order(a2, decreasing=F)]  
  
> order(a2, decreasing=F)  
[1] 3 2 1  
  
> a2[order(a2, decreasing=F)]  
[1] high low middle  
Levels: high low middle  
  
> a[order(a, decreasing=F)]  
[1] "high" "low" "middle"
```

우리가 생각하는 low middle high 순서로 순서를 부여해서 a3 factor를 구성

a3<- factor(a, order=TRUE, leve=c("low", "middle", "high"))

a3

str(a3)

[1] middle low high

Levels: low < middle < high <---- 이렇게 순서를 부여할 수 있어서 머신러닝 모델 학습 시킬때 어떤게 높고 낮은지 즉 어떤게 좋고 나쁜지 어떤게 악성이고 어떤게 양성인지를 알려주면서 머신러닝 모델을 학습 시켜야 삽습이 됩니다.

팩터(factor)란?

1. 범주(값의 목록)을 갖는 vector
2. factor() 함수를 통해서 생성
3. factor는 nominal, ordinal 형식 2가지가 존재함
4. nominal은 level 순서의 값이 무의미하며 알파벳 순서로 정의됨

```
예 : a <- c("middle", "low", "high") #벡터를 생성함
```

```
a2 <-factor(a)
```

```
a2
```

5. ordinal은 level 순서의 값을 직접 정의해서 원하는 순서를 정함

```
dP:
```

```
a3<- factor( a, order=TRUE, leve=c("low", "middle", "high") )
```

```
a3
```

```
str(a3)
```

머신러닝 모델을 학습 시킬때는 데이터를 factor로 제공해야 한다.

왜냐하면 뭐가 크고 작은지, 뭐가 좋고 나쁜지 알아야 하기 때문이다.

예제: 유방암 데이터의 라벨은 factor로 줘야한다.

```
wisc <- read.csv("wisc_bc_data.csv", header=T, stringsAsFactor=F )
```

설명: stringAsFactor=F는 wisc_bc_data의 데이터 중에 문자형 데이터를 Factor로 변형하지 말고 문자형으로 쓰겠다.

그래서 아래와 같이 정답(label)컬럼인 diagnosis가 문자형(chr)입니다.

```
$diagnosis : Factor
```

그런데 위와같이 정답은 factor로 주지않고 문자형을 주면 B와 M중에서 뭐가 좋고 나쁜지 알수 없으므로 아래와 같이 Factor로 변환해서 머신러닝 모델에게 제공해 줘야함.

```
wisc <- read.csv("wisc_bc_data.csv", header=T, stringsAsFactor=T )
```

```
str(wisc)
```

■ R을 활용한 머신러닝

1장. 기계학습이 무엇인지? 기계학습의 종류

2장. 머신러닝을 공부하기 위해서 기본적으로 알아야하는 내용

- A. factor가 무엇인지?
- B. R에서 데이터를 로드하는 방법 4가지 (책 84페이지)
- C. 수치변수 탐색 방법

■ R에서 데이터를 로드하는 방법 4가지 (책 p.84)

1. csv 파일을 로드하는 방법
2. xlsx 파일을 로드하는 방법
3. text 파일을 로드하는 방법
4. database와 R을 연동하는 방법

■ 1. csv 파일을 로드하는 방법

```
예제: setwd("d:\data")
      emp <- read.csv("emp.csv", header=T, stringsAsFactor=T)
```

설명: `read.csv` 함수는 `utils` 패키지에 내장되어 있는 함수입니다.
`stringsAsFactor=T` 은 문자형 Factor형으로 변환하는 옵션입니다.

■ 2. xlsx 파일을 로드하는 방법

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

dept <- read.xlsx( "dept.XLS", 1 )
                    ↑
                    sheet 번호
```

```
> dept<-read.xlsx("dept.XLS",1)
> dept
  DEPTNO    DNAME      LOC
1     50      DDD NEW YORK
2     70        aa    bb
3     20  RESEARCH  DALLAS
4     30     SALES      AA
5     40 OPERATIONS  BOSTON
```

■ 3. text 파일을 로드하는 방법

```
niv <- readLines("NIV.txt")
niv
```

■ 4. database와 연동해서 R로 로드하는 방법(R과 오라클 연동)

1. 오라클에 접속합니다.

```
C:\Users\stu>sqlplus "as/ sysdba"
C:\Users\stu>sqlplus "sys/oracle as sysdba"
```

```
C:\Users\stu>sqlplus "sqlplus scott/tiger"
```

2. db와 연동하기 위해서 필요한 패키지를 설치합니다.

```
install.packages("DBI")
install.packages("RJDBC")
```

```
library("DBI")
library("RJDBC")
```

안되면 "서비스"에 들어가서 oracle 다 켜져있는지 확인하고 안되어있으면 켜주기!!

```
driver <- JDBC("oracle.jdbc.driver.OracleDriver", "ojdbc8.jar")
```

설명: 오라클과 R을 연동하려면 ojdbc8.jar 파일이 있어야 합니다. 워킹 디렉토리에 ojdbc8.jar를 가져다 둡니다.

3. 도스창을 열고 리스너의 상태를 확인합니다.

```
C:\Users\stu03>lsnrctl status
```

여기서 확인할 내용은 포트번호와 서비스 이름입니다.

포트번호: 1522

서비스 이름: orcl2

4. 도스창에서 위의 정보로 오라클에 접속이 되는지 확인합니다.

```
C:\Users\stu03> sqlplus scott/tiger@127.0.0.1:1522/orcl2
```

sys로 접속시

```
C:\Users\stu03> sqlplus sys/oracle@127.0.0.1:1522/orcl2 as sysdba
```

5. 아래의 R 명령어로 오라클의 데이터를 쿼리합니다.

(R에서 치기)

--scott인 사람

```
oracle_db <- dbConnect( driver, 'jdbc:oracle:thin:@//127.0.0.1:1522/orcl2', 'scott', 'tiger')
```

#2. query and load

```
emp_query <- "select * from emp"
emp_data <- dbGetQuery( oracle_db, emp_query)
emp_data
```

```
emp_data <- dbGetQuery( oracle_db, emp_query )
emp_data
```

- 수치 데이터 탐색 순서

1. summary 함수로 평균, 중앙, 최소, 최댓값을 확인합니다.

■ **summary** 함수를 사용한다.

데이터의 각 변수(컬럼)에 대해서 최대, 최소, 평균, 중앙값을 요약해서 보여줌

이 값들을 살펴보면 데이터의 중심이 어떻게 되는지 확인할 수 있습니다.

예 : 중고차 가격의 평균값, 중앙값, 최댓값, 최솟값이 어떻게 되는지 확인

```
car <- read.csv("usedcar.csv")
car
summary(car)
```

결과에 대한 설명: 중고차 최소가격: 3800 달러(456만원)
평균: 12962 달러(천 5백 5십 5만원)
최대 : 21992 달러(2천 6백만원)

2. 이상치가 있는지 확인한다.

■ 이상치 확인하는 방법

이상치를 확인해야하는 이유?

- 이상치가 있으면 머신러닝 학습시 학습이 잘 안된다.
- 이상치를 제거한 보편적이고 일반적인 데이터로 학습시키는 것이 중요!!!

예시: 3. 이상치를 확인한다.

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

```
outlier( car$price ) # 결과: 3800
```

또 다른 방법

```
x2 <- boxplot( car$price )
x2$out # 결과: 21992 20995 4899 3800
```

이상치를 포함시켜서 학습을 시켰는데 모델의 정확도가 높지 않다면 이상치를 제거하고 학습 시켜서 모델의 정확도가 올라가는지 확인해야 한다.

3. 데이터의 편향여부를 확인해야 합니다.

■ 편향

평균값과 중앙값을 비교해서 편향여부를 확인할 수 있습니다.

평균값 > 중앙값 --> 이상치 때문에 평균이 올라간 것

데이터가 오른쪽으로 편향 됨

평균값 < 중앙값 --> 데이터가 왼쪽으로 편향됨

예제 : 3. 중고차 가격 데이터의 편향 여부를 확인한다.

```
summary(car$price)
```

결과:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
3800	10995	13592	12962	14904	21992

평균값: 12962 < 중앙값 : 13592 (따라서 [왼쪽편향](#))

예제: 중고차의 마일리지(주행거리)의 편향여부를 확인합니다.

```
summary(car$mileage)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
4867	27200	36385	44261	55125	151479

평균값: 44261 > 중앙값: 36385 (따라서 오른쪽편향)

설명 : 평균값이 중앙값보다 크다면 데이터가 오른쪽으로 편향되었으므로 이상치가 있는지 확인을 해봐야합니다.

4. 히스토그램 그래프를 확인하여 데이터를 시각화해 봅니다.

예 : 4. 자동화 스크립트 r()을 실행한다.

```
r()
```



오른쪽으로 편향되었다.

5. 사분위 그래프를 확인해 봅니다.

예: 5. 사분위그래프를 확인해본다.

머신러닝 학습시킬때 이상치를 반드시 확인해봐야 한다.

6. 왜도값과 첨도값을 확인합니다.

a. 왜도: 데이터의 좌우로 기울어짐 정도

왜도값 > 0 : 오른쪽으로 꼬리가 길다.

왜도값 < 0 : 왼쪽으로 꼬리가 길다.

b. 첨도: 위아래 뾰족한 정도

첨도값이 3에 가까울수록 정규분포에 해당, 3보다 작은 경우는 완만한 곡선,

3보다 크면 뾰족한 곡선

가급적 데이터가 정규분포형태를 보여야 학습이 잘된다.

예제: 중고차의 주행거리의 왜도값을 확인하기

```
install.packages("fBasics")
library(fBasics)
skewness( car$mileage )
```

```
[1] 1.231805
attr("method")
```

[1] "moment"

7. 산포도 그래프를 그려봅니다.

■ 산포도 그래프

두 변수간의 관계를 파악할 때 사용하는 그래프

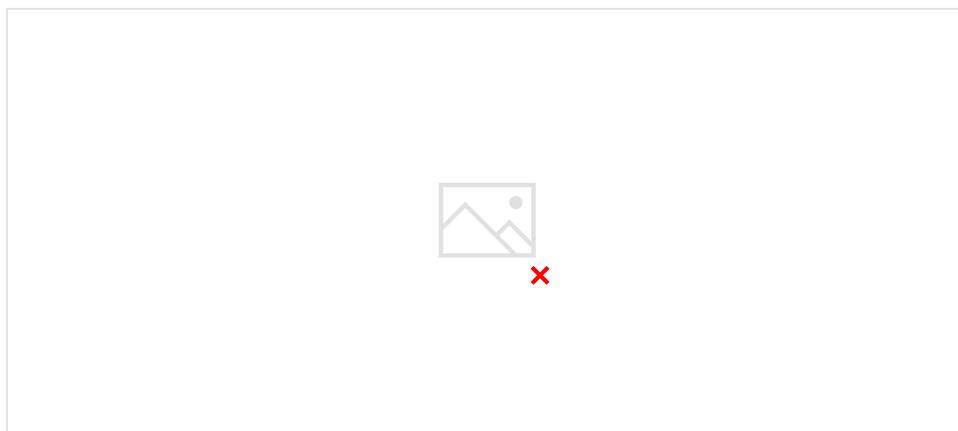
특히 두 변수간의 관계가 양의 관계인지 음의 관계인지를 파악할 때 유용합니다.

빅데이터 기사시험: 4-51 두 데이터(변량)간의 상관관계 유무를 xy평면상에 시각적으로 나타내는 그림이다.

예제: 사원 테이블의 커미션과 월급과의 관계를 산포도 그래프로 그리시오!

```
plot( emp$comm, emp$sal, pch=21, col='red', bg='red')
```

설명: 그래프를 보면 커미션을 받는 사원들의 월급이 대체적으로 작은것을 확인할 수 있다.
이 사원들의 직업은 SALESMAN이라서 커미션으로 수익을 올리기 때문이라 볼 수 있다.





문제167. 산포도 그래프를 자동화 스크립트에 7번에 추가하시오!

```
sanpodo <- function() {  
  
  fname <- file.choose()  
  table <- read.csv(fname, header=T, stringsAsFactor=F)  
  
  print(data.table(colnames(table)))  
  
  xcol_num <- as.numeric(readline('x축 컬럼 번호: '))  
  ycol_num <- as.numeric(readline('y축 컬럼 번호: '))  
  
  xcol <- colnames(table[xcol_num])  
  ycol <- colnames(table[ycol_num])  
  
  xcol2 <- table[,xcol]  
  ycol2 <- table[,ycol]  
  
  plot(xcol2,ycol2,  
    main=paste(xcol,'과',ycol,'의 산포도 그래프'),lwd=2,  
    xlab=xcol,ylab=ycol,col='blue',pch=21,bg='red')  
}
```

■ 2장의 내용

1. Factor를 이해
2. R로 데이터를 로드하는 4가지 방법
3. 수치형 데이터를 관찰하는 7가지 방법:
 - a. 이상치 확인
 - b. 데이터의 정규성 확인
 - c. 상관관계 확인
4. 이원교차표 확인하는 방법

■ 관계의 관찰: 이원교차표 (책 109 페이지)

두 명목 변수간의 관계를 관찰하기 위해 이원 교차표 사용

교차표는 하나의 변수값이 다른 변수 값에 의해 어떻게 변하는지 관찰할 수 있다는 점에서 산포도와 유사함

예:

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

두 변수간의 관계를 확인하는 기준 방법

```
attach(emp)
tapply( empno, list(deptno, job), length, default=0 )
```

이원 교차표로 확인하는 방법

```
library(gmodels)
CrossTable( x=emp$deptno, y=emp$job )
```

※ Crosstable 해석?

1. 행합계
2. 열합계
3. 표합계에 대한 셀의 상대적 비율을 나타냄
4. 카이제곱 통계

예:

```
CrossTable( x=emp$deptno, y=emp$job, chisq=TRUE )
```

결과:

Cell Contents	

N	
Chi-square contribution	
N / Row Total	
N / Col Total	
N / Table Total	

|-----|

Total Observations in Table: 14

	emp\$job						
emp\$deptno	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN	Row Total	Total
10	0	1	1	1	0	3	
	0.429	0.024	0.198	2.881	0.857		
	0.000	0.333	0.333	0.333	0.000	0.214	
	0.000	0.250	0.333	1.000	0.000		
	0.000	0.071	0.071	0.071	0.000		
20	2	2	1	0	0	5	
	2.314	0.229	0.005	0.357	1.429		
	0.400	0.400	0.200	0.000	0.000	0.357	
	1.000	0.500	0.333	0.000	0.000		
	0.143	0.143	0.071	0.000	0.000		
30	0	1	1	0	4	6	
	0.857	0.298	0.063	0.429	3.048		
	0.000	0.167	0.167	0.000	0.667	0.429	
	0.000	0.250	0.333	0.000	1.000		
	0.000	0.071	0.071	0.000	0.286		
Column Total	2	4	3	1	4	14	
	0.143	0.286	0.214	0.071	0.286		

Statistics for All Table Factors

Pearson's Chi-squared test

Chi^2 = 13.41667 d.f. = 8 p = 0.09829496

■ R 수업 복습

1. R을 왜 배워야 하는지? 데이터 분석가

- 데이터 분석을 위한 풍부한 패키지를 존재
- 훌륭한 데이터 시각화
- 빅데이터 기사 시험 준비

2. R 기본 문법: 데이터 검색

3. R을 이용한 시각화: 1. 막대 그래프

- 2. 원형 그래프
- 3. 히스토그램 그래프
- 4. 박스 그래프
- 5. 산포도 그래프
- 6. 라인 그래프

데이터 시각화를 편하게 할 수 있도록 자동화 코드를 개발

■ 데이터 시각화 자동화 코드 구현

my_func2.R

■ R을 활용한 머신러닝 2장의 복습

" 머신러닝 수업을 위해 알아야하는 기본 R 문법들"

1. **팩터(factor)란 무엇인가?** 머신러닝 학습을 위해서는 반드시 데이터를 문자형이 아닌 팩터로 변환해줘야 함, 그래서 팩터를 이해하고 있어야 함
팩터 : 벡터(vector) + 순서(순위)
뭐가 좋고 나쁜지를 컴퓨터에게 알려주려면 팩터로 변환해야 합니다.

2. R로 데이터를 로드하는 4가지 방법

3. 수치형 데이터를 관찰하는 7가지 방법

--> 이상치확인(박스그래프)

데이터의 정규성 확인(히스토그램 그래프)

상관관계 확인(산포도 그래프)

4. 이원교차표 --> 머신러닝 학습 결과를 확인할 때 반드시 필요한 결과가 이원교차표 결과입니다.

```
car <- read.csv("usedcars.csv")
car$conservation<-car$color %in% c("Black","Gray","Silver","White")
library(gmodels)
CrossTable(car$model,car$conservation)
```

car\$model	car\$conservative		Row Total
	FALSE	TRUE	
SE	27	51	78
	0.009	0.004	
	0.346	0.654	0.520
	0.529	0.515	
	0.180	0.340	
SEL	7	16	23
	0.086	0.044	
	0.304	0.696	0.153
	0.137	0.162	
	0.047	0.107	
SES	17	32	49
	0.007	0.004	
	0.347	0.653	0.327
	0.333	0.323	
	0.113	0.213	
Column Total	51	99	150
	0.340	0.660	

설명 : 총 150대의 차들중에서 99대가 보수적인 색깔을 이루고 있다.

그럼 과연 차 모델과 색깔과 연관성은 있는가? **연관성이 얼마나 있는가?**

--> 카이제곱 검정 결과를 같이 확인해야한다. (빅데이터 기사시험 책 3-66)

■ 카이제곱 검정 정의?

카이제곱 검정값은 편차의 제곱값을 기대 빈도로 나눈값들의 합

카이제곱 분포는 1900년경에 칼 피어슨에 의해서 개발된 모집단에 대한 가설 검정이나 교차분석에서 유용하게 사용되는 분포이다.

예: 아래의 2X2 크로스 집계표를 보고 두 변수가 연관성이 있는지 살펴보시오.

관측빈도

	당뇨	정상	전체
비만	10	10	20
정상체중	15	65	80
전체	25	75	100

귀무가설: 두 변수는 연관성이 없다.

대립가설: 두 변수는 연관성이 있다.

↓

당뇨와 비만과 연관성이 있다.

기대빈도

	당뇨	정상
비만	$100 \times 20/100 \times 25/100 = 5$	$100 \times 20/100 \times 75/100 = 15$
정상체중	$100 \times 80/100 \times 25/100 = 20$	$100 \times 80/100 \times 75/100 = 60$

	당뇨	정상
비만	5	15
정상체중	20	60

$$\text{카이제곱값} = (10-5)^2/5 + (10-15)^2/15 + (15-20)^2/20 + (65-60)^2/60 = 8.3333$$

카이제곱값 : 8.33, 자유도 1 일때 p-value는?

`1-pchisq(q = 8.33, df=1, lower.tail=TRUE)
0.003899566`

설명 : 0.004의 확률이므로 유의수준이 5%이면 귀무가설이 기각되고 대립가설이 채택되므로 비만과 당뇨는 연관성이 있다.

카이제곱 값 ↑ ---> 확률 ↓ ---> 두변수의 연관성이 많다.

카이제곱값 ↓ ---> 확률 ↑ ---> 두 변수의 연관성이 적다.

```
car <- read.csv("usedcars.csv")
car$conservation<-car$color %in% c("Black","Gray","Silver","White")
library(gmodels)
CrossTable(car$model,car$conservation, chisq=TRUE)
```

책 페이지 112 밑에서 3번째줄 chisq=TRUE 파라미터를 지정하면 카이제곱 값 출력

Pearson's Chi-squared test

Chi^2 = 0.1539564 d.f. = 2 p = 0.92591
↑ ↑ ↑

p-value 확률이 매우 낮다면 두 변수가 연관성이 있다는 강한 증거를 제공한다.
 그런데 위의 경우는 확률이 93%로 셀 개수의 변화는 단지 우연때문이고 차모델과 색깔은
 실제 연관성이 있지 않을 가능성이 높다.

■ R에서의 if문 사용법

```
문법 : if ( 조건식 ) {
    조건식이 True일 때 실행되는 식
}
else if ( 조건식 ) {
    조건식이 True일 때 실행되는 식
}
else {
    위의 조건식들에 만족하지 않는 경우 실행되는 식
}
```

예제1 : if문을 이용해서 피타고라스의 직각삼각형 여부를 구현하시오!

```
triangle <- function() {
    low <- as.integer( readline(prompt='밑변의 길이:' ) )
    high <- as.integer( readline(prompt='높이의 길이:' ) )
    sli <- as.integer( readline(prompt='빗변의 길이:' ) )

    if (low^2 + high^2 == sli^2) { print( '직각삼각형이 맞습니다.' ) }
    else {print('직각삼각형이 아닙니다')}
}
```

triangle()

■ R에서의 loop문 사용법

문법: for (루프변수 in 반복할 리스트) { 반복할 실행문 }

예제1: for (i in 1:10) { print (i) }

예제2. 위의 for 문을 함수로 만들어서 실행하시오!

```
aaa <- function(x) { for ( i in 1:x ) {print (i) }
}
```

aaa(10)

예제3. 별을 3개 출력하시오!

```
rep('★', 3)
```

■ 3장. Knn 알고리즘

■ 3.1 knn 알고리즘이란 무엇인가?

ppt 첨부



knn 알고리
즘

설명: k nearest neighbor의 약자로 k 개의 최근접 이웃이라는 뜻입니다. 머신러닝 지도학습의 분류에 해당되는 알고리즘이다.

새로 들어온 데이터가 기존 데이터의 그룹에서 어느 그룹에 속하는지 찾을 때 거리가 가장 가까운 데이터의 그룹을 자기 그룹으로 선택하는 아주 간단한 알고리즘이다.

■ knn 알고리즘의 장/단점

- 장점 : 단순하고 효율적이다. 모델을 훈련시키지 않는다.
- 단점 : 모델을 생성하지 않아서 특징과 클래스 간의 관계를 이해하는 능력이 제한된다.
 - 적절한 k값을 모델 개발자가 직접 알아내야 함
 - ↳ 언더피팅과 오버피팅이 심하지 않은 적절한 k값을 실험을 통해서 알아내야 함

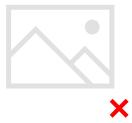
■ knn의 원리

새로 들어온 데이터가 기존 데이터 중에서 (악성종양, 양성종양) 중 어느 데이터에 더 인접해 있는지 거리를 계산해서 가장 가까운 거리에 있는 데이터를 자기 이웃으로 선택함

거리를 계산할 때 사용하는 수학식? 유클리드 거리 계산식 (책 121페이지)

■ 군집간의 거리 계산 (빅데이터 기사시험 이론 수제비 책 3-59)

연속형 변수거리:
1. 수학적 거리 : 유클리드 거리, 맨하튼 거리, 민코프스키 거리
2. 통계적 거리 : 표준화 거리, 마할라노비스 거리



3. 두 점 사이의 거리를 구한다 (R로 구하기)

a = c(2,4)
b = c(5,6)

```
sqrt( sum( (a-b) ^2 ) ) #3.605551
```

4. 위의 거리를 구하기 쉽도록 **distance**라는 함수를 만들어서 아래와 같이 실행되게 하시오!

```
distance(a,b) #3.605551
```

```
답 : distance <- function(a,b) {  
  return ( sqrt( sum( (a-b) ^2 ) ) )  
}  
  
distance( a,b )
```

■ knn 머신러닝 알고리즘을 이용하여 유방암 데이터의 양성과 악성을 분류하기

아산병원에서 유방암 진단을 하는 의사 선생님들이 사용할 인공지능 툴 생성

1단계 : 데이터 수집 ---> 데이터 출처, 데이터 설명

2단계 : 데이터 탐색과 준비 ---> 결측치, 이상치, 정규분포(히스토그램)을 확인해서 머신러닝 모델이

학습하기 적합한 데이터인지를 확인

3단계 : 데이터로 모델 훈련 ---> knn 알고리즘(유clidean 거리계산 공식)으로 모델 생성

4단계 : 모델 성능 평가 ---> 이원교차표를 통해서 모델의 정확도를 확인

5단계 : 모델 성능 개선 ---> 정확도가 제일 좋은 k 값을 알아내기

■ 1단계 : 데이터 수집

위스콘신 유방암 진단 데이터셋이며 이 데이터를 569개의 암 조직검사 예시가 되어있으며, 각 예시는 32개의 특징을 갖는다. 그 특징은 디지털 이미지에 존재하는 세포핵의 특성을 나타낸다.

반지름	조밀성
질감	오목함
둘레	오목점
넓이	대칭성
매끄러움	프랙탈 차원

지도학습이므로 정답이 있는 라벨 컬럼이 있는데 이 라벨컬럼이 diagnosis입니다.

라벨: 양성(B), 악성(M)

■ 2단계 : 데이터 탐색(시각화)

- 정답에 해당하는 라벨 컬럼의 데이터 분포를 원형그래프로 시각화
r()

숫자를 선택하세요 ~ 종료하려면 0번을 누르세요

- 1: 막대 그래프
- 2: 원형 그래프
- 3: 라인 그래프
- 4: 히스토그램 그래프
- 5: 박스 그래프
- 6: 테이블과 통계정보
- 7: 산포도 그래프
- 8: 범주형 원형 그래프

선택: 8

양성이 60%, 악성이 40% 비율로 구성되어있음.

- 이상치가 얼마나 있는지 확인

게시글 326. 이상치 확인 함수

```

library(outliers)

grubbs.flag <- function(x) {
  outliers <- NULL
  test <- x
  grubbs.result <- grubbs.test(test)
  pv <- grubbs.result$p.value
  while(pv < 0.05) {
    outliers <- c(outliers,as.numeric(strsplit(grubbs.result$alternative," ")[[1]][3]))
    test <- x[!x %in% outliers]
    grubbs.result <- grubbs.test(test)
    pv <- grubbs.result$p.value
  }
  return(data.frame(X=x,Outlier=(x %in% outliers)))
}

```

```

wisc <- read.csv("wisc_bc_data.csv")
colnames(wisc)

grubbs.flag(wisc$radius_mean)

```

```
library(outliers)
```

```

grubbs.flag <- function(x) {
  outliers <- NULL
  test <- x
  grubbs.result <- grubbs.test(test)
  pv <- grubbs.result$p.value
  while(pv < 0.05) {
    outliers <- c(outliers,as.numeric(strsplit(grubbs.result$alternative," ")[[1]][3]))
    test <- x[!x %in% outliers]
    grubbs.result <- grubbs.test(test)
    pv <- grubbs.result$p.value
  }
  return(data.frame(X=x,Outlier=(x %in% outliers)))
}

```

```
wisc <- read.csv("wisc_bc_data.csv")
```

```

for (i in 3:length(colnames(wisc))){
  a = grubbs.flag(wisc[,colnames(wisc)[i]])
  b = a[a$Outlier==TRUE,"Outlier"]
  print ( paste( colnames(wisc)[i] , '--> ', length(b) ) )
}
```

```
}
```

일단 지금은 이상치 값이 얼마나 있는지 정도만 확인하고 나중에 모델의 정확도가 너무 안 나오면 이상치 데이터를 삭제하고 학습시킬것을 고려해야 한다.

3. 결측치가 많은 컬럼이 무엇이 있는지 확인합니다.

```
colSums( is.na(wisc) )
```

설명 : 유방암 데이터에는 결측치가 없다. 만약 결측치가 많은 데이터가 있다면 결측치를 다른값으로 치환하거나 삭제를 고려해야 한다.

■ 3단계 : 데이터로 모델 훈련 ---> knn 알고리즘(유클리드 거리계산 공식)으로 모델 생성

1. 데이터를 로드한다.

```
wbcd <- read.csv("wisc_bc_data.csv", header=True, stringAsFactors=FALSE)
```

2. diagnosis (정답컬럼)을 factor로 변환한다.

```
wbcd$diagnosis <- factor( wbcd$diagnosis,  
                           levels= c("B", "M"),  
                           labels=c("Benign", "Malignant") )
```

```
str(wbcd)
```

3. 데이터를 shuffle 시킵니다.

sample(10) # 1부터 10까지의 숫자를 섞어서 출력하는 코드

```
wbcd_shuffle <- wbcd[ sample( nrow(wbcd) ), ]
```

설명 : wbcd[행, 열]

```
wbcd_shuffle
```

설명 : 왜 데이터를 섞어야 하는가?

개와 고양이를 분류하는 인공지능 모델을 만든다면 이 모델을 학습시킬 때 개 사진 10000장, 고양이 사진 10000장이 있다면 개 사진을 먼저 10000장을 학습시키고 나중에 고양이 사진 10000장을 학습 시키는 것보다 섞어서 학습 시키는게 학습이 더 잘된다.

4. 데이터에서 환자번호 id 컬럼을 제외 시킨다.

```
wbcd2 <- wbcd_shuffle[ , -1]  
str(wbcd2)
```

5. 데이터를 정규화 한다.

정규화를 왜 해야하는가? 몸무게, 키와 같이 서로 단위가 다른 데이터를 전부 0~1사이로 데이터를 맞춰 줌.

단위가 서로 다르면 몸무게는 80, 키 172 이런식이라 키 데이터의 영향도가 커짐.

모든 컬럼을 0~1사이의 데이터로 변경해 주어야 함

정규화 방법 2가지: 1. scale 함수 사용

2. min/max 함수 사용(머신러닝시에 학습 잘 되는 함수)

```
normalize <- function(x) {  
  return ( (x-min(x)) / ( max(x) - min(x) ) )  
}
```

```
wbcd_n <- as.data.frame( lapply( wbcd2[ , 2:31], normalize) )
```

설명 : shuffle 시킨 wbcd2 데이터의 2번 컬럼부터 31번째 컬럼까지의 데이터를 normalize 함수에 적용시켜서 데이터를 변환하고 데이터 프레임으로 구성해라~

```
summary(wbcd_n) # 전부 0~1사이의 데이터로 변환 되었는지 확인
```

6. 훈련 데이터와 테스트 데이터로 wbcd_n 데이터를 9대 1로 나눈다.

```
nrow( wbcd_n )  
train_num <- round( 0.9* nrow(wbcd_n), 0 )  
train_num #512
```

```
wbcd_train <- wbcd_n[ 1:train_num, ]  
wbcd_test <- wbcd_n[ (train_num + 1) : nrow(wbcd_n), ]  
nrow(wbcd_test) #57
```

7. 훈련데이터에 라벨(정답)을 생성하고 테스트 데이터의 라벨(정답)을 생성한다.

```
wbcd_train_label <- wbcd2[ 1:train_num, 1]  
wbcd_test_label <- wbcd2[ (train_num+1): nrow(wbcd_n), 1]  
wbcd_test_label
```

8. knn 모델로 훈련시켜서 모델을 만들고 바로 그 모델에 test 데이터를 넣어서 정확도를 확인한다

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

result1 <- knn(train=wbcn_train, test=wbcn_test, cl=wbcn_train_label, k=21)
```

설명 : train = 훈련데이터, test = 테스트 데이터, cl= 훈련데이터의 라벨
result1에 테스트 데이터에 대해서 머신러닝 모델이 알아낸 정답이 들어간다.
k를 21로 줘서 만들었는데 이 숫자의 제일 좋은 숫자는 우리가 알아내야 함

```
result1
```

실제 정답은 wbcn_test_label이다. 아래의 코드 2개를 서로 비교한다.

```
data.frame( result1, wbcn_test_label )

sum( result1 == wbcn_test_label )
```

■ R 수업내용 복습

1. R을 왜 배워야 하는가? (머신러닝을 이용한 분류, 예측) / 빅데이터 기사 자격증, 포트폴리오 2개
 1. 리눅스와 하둡
 2. 케글 도전
2. R 기본 문법 (SQL과 비교)
3. R에서 그래프 그리는 방법 (자동화 스크립트)
4. 머신러닝 책 1장 기계학습이란 무엇인가?
 - **지도학습** : 정답이 있는 데이터로 학습
 - 종류: knn, 나이브베이즈, 의사결정트리(랜덤포레스트), 인공신경망, 서포트 벡터 머신, 연관분석, 회귀분석
 - **비지도학습** : 정답이 없는 데이터로 학습
 - 종류: k-means
 - **강화학습** : 에이전트가 환경에 적응해 나가는 빅데이터를 생성하는 학습방법
5. 머신러닝 책 2장 기계학습을 배우기 위해 기본적으로 알아야하는 내용
6. 머신러닝 책 3장 knn 알고리즘

유클리드 거리공식을 이용해서 가장 가까운 이웃을 자기 이웃으로 보고 분류하는 머신러닝 알고리즘

유방암 환자의 종양을 시약 검사하지 않고 사진만 보고 악성종양인지 양성종양인지 판단하는 머신러닝 모델을 개발

1단계. 데이터 수집 : 위스콘신 대학교에서 제공하는 유방암 환자 데이터 569개

2단계. 데이터 탐색 :

- 정답 라벨 데이터 분포를 확인 (원형그래프)
- 이상치가 존재 확인 : 이상치 제거 고려
- 결측치 존재 확인 : 다른 데이터로 치환을 고려
- 문자형 데이터인지, 숫자형 데이터인지, 둘 다 섞여 있는지 확인 : 어떤 머신러닝 알고리즘을 선택할지를 결정해야 하기 때문에 확인

3단계. 데이터 모델 훈련

1. 데이터를 로드한다. (stringsAsFactors=FALSE)

2. 라벨(정답)만 팩터로 변환한다.

3. 데이터를 shuffle 시킨다.

설명 : 데이터를 shuffle 시키는 이유는 무엇인가?

- 섞어야 학습이 더 잘된다. sample(숫자) 함수를 이용해 shuffle 시켰다.

예: sample(5)

설명: 사실 createPartition 함수를 이용하면 **shuffle**과 **데이터 분할**을 동시에 해주기 때문에 shuffle을 따로 하지 않아도 된다.

어떤 자리에서든 동일하게 shuffle되게 하려면 어떻게 해야하는가?

```
set.seed(1)  
sample(5)  
[1] 1 4 3 5 2
```

```
set.seed(2)  
sample(5)  
[1] 5 3 2 4 1
```

1. 학습에 도움이 안되는 데이터를 제외한다. (ex : 환자번호)
2. 데이터를 정규화 한다. (1. scale 함수, 2. minmax 함수)
3. 데이터를 훈련 데이터와 테스트 데이터로 나눈다.

설명 : 데이터 분할을 하는 이유는? (수제비 책 3-19)

모델(모형)이 주어진 데이터에 대해서만 높은 성능을 보이는 과대적합의 문제를 예방하여 2종 오류인 잘못된 귀무가설을 채택하는 오류를 방지하는데 목적이 있음

데이터 분할 3가지 ? 1. 훈련데이터

2. 검증용 데이터 (훈련 데이터의 일부)
3. 테스트용 데이터

검증용 데이터 ?

모형의 학습 과정에서 모형이 제대로 학습되었는지 중간에 검증을 실시하고, 과대적합과 과소 적합의 발생여부를 확인하여 모형의 튜닝에도 사용한다.

7. 훈련 데이터를 가지고 knn 모델을 생성한다.

예: result1 <- knn(train=wbc_train, test=wbc_test, cl=wbc_train_label, k=21)

knn의 k값을 우리가 직접 알아내야 한다. 이 유방암 데이터를 잘 분류하는 적절한 k 값을 우

리가 실험으로 알아내야 한다.

이러한 k 값을 뭐라고 하는가?

----> "하이퍼 파라미터"라고 한다.

• 빅데이터 기사 시험 (3-12)

- 머신러닝 모델을 학습시킬 때 필요한 파라미터 2가지?

1. **파라미터** : 모델 내부에서 확인이 가능한 변수로 데이터를 통해서 산출이 가능한 값, 학습이 되어지면 만들어지는 숫자값

예 : 인공신경망에서의 가중치, 서포트 벡터 머신에서의 서포트 벡터, 선형회귀나 로지스틱 회귀분석에서의 결정계수

2. **하이퍼 파라미터** : 모델에서 외적인 요소로 데이터 분석을 통해서 얻어지는 값이 아니라 사용자가 직접 설정해 주는 값. 경험에 의해서 결정 가능한 값.

예: knn에서의 k 값, 신경망에서의 학습률, 의사결정트리에서의 깊이 (문제 185)

■ 데이터를 잘 분할해주는 R 패키지 caret 소개 (수제비책 3-20)

```
install.packages(caret)
library(caret)
wbcn <- read.csv("wbcn_dc_data.csv")
nrow(wbcn) #569
ncol(wbcn) #32
train_num <- createDataPartition( wbcn$diagnosis, p=0.9, list=F)
```

설명 : wbcn\$diagnosis가 양성과 악성이 있는데 훈련 데이터와 테스트 데이터를 p=0.9로 했기 때문에 9대 1로 나누는데 wbcn\$diagnosis가 양성과 악성이 있으므로 훈련 데이터에 양성이나 악성이 모여 있지 않고 골고루 분포될 수 있도록 나눠주는 것이다.

훈련(90%) , 테스트(10%)

양성	양성
악성	악성

```
train_data <- wbcn[ train_num, ]
test_data <- wbcn[-train_num, ]
nrow(train_data) #513
nrow(test_data) #56
table( train_data$diagnosis )
B   M
322 191
```

```
table( test_data$diagnosis )
B   M
```

```
> prop.table(table( train_data$diagnosis ))
  B      M
0.6276803 0.3723197
```

```
> prop.table(table( test_data$diagnosis ) )
```

```
  B      M
0.625 0.375
```

4단계. 모델 성능 평가

5단계. 모델 성능 개선

■ Set Seed test 스크립트:

1. 데이터를 로드합니다.

```
wbcd <- read.csv("wisc_bc_data.csv", header=T, stringsAsFactors=FALSE)
```

```
wbcd$diagnosis <- factor(wbcd$diagnosis,
  levels =c("B","M"),
  labels = c("Benign","Malignant"))
```

2. 데이터를 shuffle 합니다.

```
set.seed(1)
wbcn_shuffle <- wbcd[sample(nrow(wbcd)), ]
```

#3. 환자번호를 제외합니다.

```
wbcd2 <- wbcn_shuffle[-1]
```

4. 데이터를 min/max 정규화 합니다. (0~1사이의 숫자로 다 변환합니다.)

```
normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x)) ) }
```

```
wbcd_n <- as.data.frame(lapply(wbcd2[2:31],normalize))
```

5. 훈련데이터와 테스트 데이터를 9대 1로 나눕니다.

```
train_num<-round(0.9*nrow(wbcd_n),0)
wbcn_train<-wbcd_n[1:train_num,]
wbcn_test<-wbcd_n[(train_num+1):nrow(wbcd_n),]
```

```
wbcd_train_label <- wbcd2[1:train_num,1]
wbcd_test_label <- wbcd2[(train_num+1):nrow(wbcd_n),1]

wbcd_test_label
```

6. 모델을 훈련시켜 테스트 데이터의 라벨을 예측합니다.

```
library(class)
set.seed(1)
result1 <- knn(train=wbcd_train, test=wbcd_test, cl=wbcd_train_label, k=21)
```

#7. 테스트 데이터에 대한 실제값과 예측값을 확인합니다.

```
x <- data.frame('실제'=wbcd_test_label, '예측'=result1)
table(x)
```

k = 2 일때 :

		예측	
실제	Benign	Maliganant	
Benign	33	1	
Maliganant	0	23	

k = 21일때 :

		예측	
실제	Benign	Maliganant	
Benign	34	0	
Maliganant	0	23	

설명 : seed값과 k 값이 결국 무엇일까? 하이퍼 파라미터 // seed 값은 사용자가 알아내야 함

■ Knn 머신러닝 알고리즘을 사용해서 머신러닝 모델을 만드는 순서

- 데이터를 수집한다.
- 데이터를 탐색한다.
- 데이터로 모델을 생성한다.
- 모델의 성능을 평가한다.

```
x <- data.frame('실제'=wbcd_test_label, '예측'=result1)
table(x)
```

- 이원 교차표로 확인하기
- ```
library(gmodels)
g2 <- Crosstable(x=wbcd_test_label, y=result1)
print(g2$prop.tb[1] + g2$prop.tb[4]) #정확도
```

- 모델의 성능을 개선한다.
- 적절한 k값과 seed값을 알아낸다.

## ■ 다른 데이터를 knn 모델에 학습 시키기 (iris 데이터셋)

head( iris )

(문제 189 참조)

아이리스를 분류할 수 있는 모델을 생성하고 농림수산부에 가서 구현을 하고 싶다면 가져가야 하는 산출물?

1. iris 데이터 : 머신러닝을 학습시키기에 가장 적합한 데이터 생성
2. seed 값
3. k 값
4. 구현 스크립트

주식에 관련한 회사들 : raw데이터를 가지고 주가를 예측하기 좋게 가공한 데이터를 네이버나 다음에 판매하는 회사

## ■ R에 이미 내장된 data 셋 확인하는 방법

data()

\* 농림수산부에 머신러닝 구현을 해주려 간다면 필요한 산출물

1. 데이터
2. seed 값 (62)
3. k 값 (7)
4. 스크립트 전체

사용한 데이터: 유방암 데이터, 아이리스 데이터, 와인 데이터

## ■ 4장. 나이브 베이즈 알고리즘

### ■ 나이브 베이즈 알고리즘이 사용 되는 분야 :

1. 스팸 이메일 필터링과 같은 텍스트 분류
2. 컴퓨터 네트워크에서 침입이나 비정상적인 행위를 탐지
3. 일련의 관찰된 증상에 따른 의학적 질병 진단

나이브베이즈 이론 ppt 설명 ---> R 목차

예제: 비아그라가 포함되어져 있는 메일이 스팸메일일 확률을 구하시오 !

$$\begin{array}{ccc}
 & \text{우도} & \text{사전확률} \\
 & \downarrow & \downarrow \\
 P(\text{비아그라} | \text{스팸}) * P(\text{스팸}) & & \\
 P(\text{스팸} | \text{비아그라}) = & \cdots\cdots\cdots & \\
 & \uparrow & \\
 & \text{P}(\text{비아그라}) & \\
 & \uparrow & \\
 \text{사후확률} & & \\
 & \uparrow & \\
 & \text{주변우도} & 
 \end{array}$$

우도  $P(\text{비아그라} | \text{스팸})$ ? 이전 스팸메세지에서 단어 비아그라가 사용된 확률

베이즈 이론을 적용해서 메세지가 스팸이 될 확률을 측정한 후 사후확률을 계산해서 사후확률이 50% 보다 크다면 이 메세지는 햄보다는 스팸일 될 가능성이 좀더 크다.

빈도표  $\cdots\cdots\cdots \rightarrow$  우도표

| 비아그라 |   |     | 비아그라 |    |       |        |    |
|------|---|-----|------|----|-------|--------|----|
|      | 예 | 아니오 | 총계   |    | 예     | 아니오    | 총계 |
| 스팸   | 4 | 16  | 20   | 스팸 | 4/20  | 16/20  | 20 |
| 햄    | 1 | 79  | 80   | 햄  | 1/80  | 79/80  | 80 |
|      | 5 | 95  | 100  |    | 5/100 | 95/100 |    |

$$P(\text{비아그라}=예 | \text{스팸}) * P(\text{스팸}) = 4/20 * 20/100$$

$$P(\text{스팸} | \text{비아그라}) = \frac{P(\text{비아그라})}{P(\text{비아그라})} = \frac{5/100}{5/100}$$

$$= 0.8$$

### ■ 비아그라 뿐만 아니라 다른 단어들도 여려개 있는 경우의 스팸일 확률?

$P(\text{스팸} | \text{비아그라} \cap \text{돈} \cap \text{식료품} \cap \text{주소삭제}) ?$

우도표

비아그라 단어 하나만 가지고 스팸메일인지를 분류하면 정확하게 분류가 안될 수 있으니 다른 단어들도 같이 포함시켜야한다.

수학기호 :  $\neg$  : 존재하지 않는다.(부정)  
 $\exists$  : 존재한다. (긍정)

$P(\text{스팸} | \text{비아그라} \cap \neg\text{돈} \cap \neg\text{식료품} \cap \text{구독취소}) = ?$

비아그라와 구독취소는 포함되어져 있는데 돈과 식료품은 포함되지 않는 메일이 스팸일 확률?

$$\begin{aligned}
 P(B|A) * P(A) &= P(\text{비아그라} \cap \neg\text{돈} \cap \neg\text{식료품} \cap \text{구독취소} | \text{스팸}) * P(\text{스팸}) \\
 P(A|B) = \frac{P(A)}{P(B)} &= \frac{P(\text{비아그라} \cap \neg\text{돈} \cap \neg\text{식료품} \cap \text{구독취소})}{P(\text{비아그라} \cap \neg\text{돈} \cap \neg\text{식료품} \cap \text{구독취소} | \text{스팸}) * P(\text{스팸})} \\
 \\ 
 &= \frac{P(\text{비아그라} | \text{스팸}) * P(\neg\text{돈} | \text{스팸}) * P(\neg\text{식료품} | \text{스팸}) * P(\text{구독취소} | \text{스팸}) * P(\text{스팸})}{P(\text{비아그라} \cap \neg\text{돈} \cap \neg\text{식료품} \cap \text{구독취소})}
 \end{aligned}$$

=

### 분모무시

| 비아그라     |   |    | 비아그라     |    |               |
|----------|---|----|----------|----|---------------|
| 예 아니오 총계 |   |    | 예 아니오 총계 |    |               |
| 스팸       | 4 | 16 | 20       | 스팸 | 4/20 16/20 20 |
| 햄        | 1 | 79 | 80       | 햄  | 1/80 79/80 80 |
|          | 5 | 95 | 100      |    | 5/100 95/100  |

$$\begin{aligned}
 &= P(\text{비아그라} | \text{스팸}) * P(\neg\text{돈} | \text{스팸}) * P(\neg\text{식료품} | \text{스팸}) * P(\text{구독취소} | \text{스팸}) * P(\text{스팸}) \\
 &\quad 4/20 \qquad \qquad \qquad 10/20 \qquad \qquad \qquad 20/20 \qquad \qquad \qquad 12/20 \qquad \qquad \qquad 20/100 \\
 &= 0.012
 \end{aligned}$$

스팸의 전체 우도 ? 0.012

햄의 전체 우도 ? 0.002

$$\begin{aligned}
 &P(\text{비아그라} | \text{햄}) * P(\neg\text{돈} | \text{햄}) * P(\neg\text{식료품} | \text{햄}) * P(\text{구독취소} | \text{햄}) * P(\text{햄}) \\
 &\quad 1/80 \qquad * \qquad 66/80 \qquad * \qquad 71/80 \qquad * \qquad 23/80 \qquad * \qquad 80/100
 \end{aligned}$$

0.012 / 0.002 가 6이므로 이 메세지가 스팸일 가능성성이 햄일 가능성보다 6배가 높다고 할 수 있다.

$$\begin{aligned}
 &0.012 \\
 &\text{스팸일 확률} ? \frac{0.012}{0.012 + 0.002} = 0.857
 \end{aligned}$$

$$\begin{aligned}
 &0.002 \\
 &\text{햄일 확률} ? \frac{0.002}{0.012 + 0.002} = 0.143
 \end{aligned}$$

이 메세지에 있는 단어 패턴에 대해 85.7% 의 확률로 메세지가 스팸이고 14.3% 의 확률로 햄이라고 예상을 한다.

■ 라플라스 추정기 P159

$P(\text{비아그라} | \text{스팸}) * P(\text{돈} | \text{스팸}) * P(\text{식료품} | \text{스팸}) * P(\text{구독취소} | \text{스팸}) * P(\text{스팸})$ 

4/20

10/20

0/20

12/20

\* 20/100

 $= 0$ 

스팸의 우도 ?

 $P(\text{비아그라} | \text{스팸}) * P(\text{돈} | \text{스팸}) * P(\text{식료품} | \text{스팸}) * P(\text{구독취소} | \text{스팸}) * P(\text{스팸})$ 

5/24

11/24

1/24

13/24

\* 20/100

햄의 우도 ?

 $P(\text{비아그라} | \text{햄}) * P(\text{돈} | \text{햄}) * P(\text{식료품} | \text{햄}) * P(\text{구독취소} | \text{햄}) * P(\text{햄})$ 

2/84

15/84

9/84

24/84

80/100

라플라스값을 주어서 나이브베이즈 모델의 성능을 올리는데 사용됨

설명: knn에서는 k값이 하이퍼 파라미터였는데 나이브베이즈에서는 라플라스 값이 하이퍼 파라미터입니다.

스팸일 확률 ?

햄일 확률 ?

## ■ 독버섯을 분류하는 나이브 베이즈 알고리즘

<https://cafe.daum.net/oracleoracle/SZTZ/2002>

### 1. 버섯 데이터를 R로 로드한다.

```
mushroom <- read.csv("mushrooms.csv", header=T, stringsAsFactors=TRUE)
```

```
View(mushroom)
```

```
str(mushroom) #문자형이 다 factor로 바뀐것을 알 수 있음
```

설명: stringAsFactors=TRUE로 설정해서 문자형 데이터를 전부 factor로 변환한다.

데이터를 살펴보니 전부 문자형이어서 knn 알고리즘으로 분류할 수 없고 나이브 베이즈 알고리즘으로 분류 해야함.

\* 맨 앞에 있는 type이 라벨이다.

```
unique(mushroom$type)
```

결과: poisonous edible

\* 독버섯과 정상버섯의 비율이 어떻게 되는지 확인하시오!

```
prop.table(table(mushroom$type))
```

```
edible poisonous
0.5179714 0.4820286
```

두개가 딱 절반이어서 독버섯도 잘 학습할 수 있고 정상버섯도 잘 학습할 수 있게 되었다. 위의 라벨 컬럼의 비율을 확인하는것이 중요한 이유가 비율이 위와같이 균등해야 오버피팅이 발생할 가능성이 많이 줄어들기 때문.

## 2. 8123 독버섯 데이터만 따로 빼서 mush\_test.csv 로 저장한다.

```
mush_test <- mushroom[8123,]
mush_test

write.csv(mush_test, "mush_test.csv", row.names=FALSE)
```

설명: write.csv로 mush\_test 데이터를 mush\_test.csv로 저장합니다.

이 한건의 데이터를 따로 뺀 이유는 나중에 학습 다 시키고 만든 모델이 데이터가 독버섯인지 정상버섯인지 잘 맞추는지 확인하기 위함.

## 3. 8123 독버섯 데이터를 훈련 데이터에서 제외 시키시오 !

```
nrow(mushroom)
mushrooms <- mushroom[-8123,]
nrow(mushrooms)
```

## 4. mushrooms 데이터를 훈련 데이터와 테스트 데이터로 나눈다

( 훈련 데이터는 75%, 테스트 데이터는 25% )

```
set.seed(1)
dim(mushrooms)
결과:
[1] 8123 23
```

설명 : 8123개의 행에 23개의 열로 구성되어 있다는 것을 확인할 수 있다.

```
dim(mushrooms)[1] #8123
dim(mushrooms)[2] #23
```

```
train_cnt <- round(0.75*dim(mushrooms)[1])
```

설명: 8123의 75%에 해당하는 숫자를 train\_cnt에 담음.

```
train_cnt
설명: 1번부터 8123번까지의 숫자중에서 6092개의 숫자를 랜덤으로 추출한다.
replace=F를 써서 비복원 추출합니다.
```

```
train_index <- sample(1:dim(mushrooms)[1], train_cnt, replace=F)
```

```
mushrooms_train <- mushrooms[train_index,] #훈련데이터 구성
```

```
mushrooms_test <- mushrooms[- train_index,] #테스트데이터 구성
```

```
nrow(mushrooms_train) # 6092
nrow(mushrooms_test) # 2031

str(mushrooms_train)
```

## 5. 나이브 베이즈 알고리즘으로 독버섯과 일반 버섯을 분류하는 모델을 생성한다.

```
install.packages("e1071")
library(e1071)
모든 컬럼들(.)
↓
```

```
model1 <- naiveBayes(type ~ ., data=mushrooms_train)
↑
라벨 컬럼명
```

model1 #버섯 데이터로 빈도표를 만들고 그 빈도표로 우도표를 생성했다.

## 6. 위에서 만든 모델과 테스트 데이터를 가지고 독버섯과 일반버섯을 잘 분류하는지 예측해본다. mushrooms\_test[ , -1] 는 정답을 뺀 테스트 데이터이다.

```
result1 <- predict(model1, mushrooms_test[, -1])

result1
```

## 7. 이원 교차표를 그려서 최종 분류 결과를 확인한다.

```
library(gmodels)
CrossTable(mushrooms_test[, 1], result1)
↑ ↑
실제 예측
g2 <- CrossTable(mushrooms_test[, 1], result1)
g2$prop.tbl
print(g2$prop.tb[1] + g2$prop.tb[4]) # 정확도
```

## 8. 위의 모델의 성능을 올리시오 !

```
model2 <- naiveBayes(type ~ ., data=mushrooms_train, laplace=0.0004)
```

설명 : 나이브베이즈 함수의 하이퍼 파라미터인 라플라스 값에 아주 작은 값을 임의로 줘서 성능을 올리면 된다. 식료품이 스팸메일에 0으로 있었던 것 처럼 0이 있으면 나이브 베이즈 함수의 성능이 현저히 떨어진다. 아주 작은 값인 라플라스 값을 줘서 0 때문에 계산되지 않게 하는 것을 방지하는데 이 값에 따라 성능이 많이 달라진다.

```
result2 <- predict(model2, mushrooms_test[, -1])
```

```
CrossTable(mushrooms_test[, 1], result2)
```

```
g3 <- CrossTable(mushrooms_test[,1], result2)

g3$prop.tbl
print(g3$prop.tb[1] + g2$prop.tb[4]) # 정확도
```

결과: (정확도)  
0.9916297

위의 모델에 별도로 구분해 놓은 테스트 데이터 한개(독버섯) 8123 번 데이터를 넣어서 독버섯인지 정상인지 확인하시오 !

```
result3 <- predict(model2, mush_test)
```

## ■ R수업 복습

1. R을 왜 배워야 하는지
2. R 기본 문법
3. R을 활용한 머신러닝 1장 ( 기계학습이 무엇인지? )
4. 머신러닝을 배우기 위해서 미리 알아야 할 내용
5. knn 알고리즘 ( 3장 ) ---> 수치형 데이터 분석해서 분류  
( 유방암 환자 데이터셋: 양성, 악성)
6. 나이브 베이즈 알고리즘 ( 4장 ) --> 명목형 데이터 분석해서 분류  
( 독버섯 데이터셋: 독버섯, 정상버섯)

## ■ 세원씨가 루프문 돌려서 알아낸 정확도 100퍼센트 독버섯 분류 코드

( 시드: 6, 라플라스: 0.000001 )

### #1. 독버섯 데이터를 R로 로드.

```
mushroom <- read.csv("mushrooms.csv", header=T, stringsAsFactors=TRUE)
View(mushroom)
str(mushroom)
```

### #2. 독버섯 데이터를 하나 뽑아서 mush\_test.csv로 저장

```
mush_test <- mushroom[8123,]
mush_test
write.csv(mush_test, "mush_test.csv", row.names=FALSE)

nrow(mushroom)
mushrooms <- mushroom[-8123,]
nrow(mushrooms)
```

# 설명: 8123만 가지고 나이브베이즈 머신러닝 모델을 학습 시킴. 또다른 테스트 데이터라고 보면 됨.  
# 모델을 활용하기 위해 하나 빼고 학습 시킴

### #3. 데이터를 shuffle 시키면서 훈련 데이터를 75%, 테스트 데이터를 25%로 구성

```
set.seed(6)

dim(mushrooms)[1] #8123
dim(mushrooms)[2] #23

train_cnt <- round(0.75*dim(mushrooms)[1])
train_cnt #6092

train_index <- sample(1:dim(mushrooms)[1], train_cnt, replace=F)
설명: 8123개 중에 6092개를 비보원 추출한 행번호(index)를 train_index에 담는다.
```

```
mushrooms_train <- mushrooms[train_index,] # 훈련데이터 구성 : 75%
mushrooms_test <- mushrooms[- train_index,] # 테스트 데이터 구성 : 25%
```

```
nrow(mushrooms_train) # 6092
nrow(mushrooms_test) # 2031
```

```
str(mushrooms_train)
```

#### #4. 나이브 베이즈 머신러닝 알고리즘 모델을 훈련데이터로 생성

```
library(e1071)
```

```
model1 <- naiveBayes(type ~ ., data=mushrooms_train)
설명: naiveBayes(정답컬럼~ 모든 컬럼들, data = 훈련 데이터 프레임명), . 은 SQL에서 *로 보면됨
```

```
model1
설명: P(식용버섯 | 버섯향 = 아몬드향) = 확률 / P(독버섯 | 버섯향 = 아몬드향) = 확률
나이브베이즈 공식에 대입해서 위의 확률들을 모두 계산해 우도표를 생성했다.
```

$$P(\text{식용버섯} \mid \text{버섯향=아몬드향} \cap \text{버섯모자의 색깔=갈색} \cap \dots) = ?$$

# 5. 위에서 생성한 머신러닝 모델을 이용해서 테스트 데이터 2031개의 라벨을 예측함

```
result1 <- predict(model1, mushrooms_test[, -1])
설명 : predict(모델명, 테스트 데이터)
```

result1

## #6. 이원 교차표 확인

```
library(gmodels)
```

```
g2 <- CrossTable(mushrooms_test[,1], result1)
```

```
print(q2$prop.tb[1] + q2$prop.tb[4]) # 정확도
```

## #7. 위의 모델의 성능 개선

```
model2 <- naiveBayes(type ~ ., data=mushrooms_train, laplace=0.000001)
```

```
result2 <- predict(model2, mushrooms_test[, -1])
```

```
g3<- CrossTable(mushrooms_test[,1], result2)
```

“是的，CrossTable(x=“国歌”，y=“军歌”)

**print( gs\$prop.tb[1] + gs\$prop.tb[4] )** # 경록호

```
설명 : 정확도 100%를 보이는 훌륭한 머신러닝 모델이 나왔다.
```

```
> g3$prop.tb[1]
[1] 0.5051699
> g3$prop.tb[2]
[1] 0
> g3$prop.tb[3]
[1] 0
> g3$prop.tb[4]
[1] 0.4948301

빼 놓은 데이터 불러오기
mush_test <- read.csv("mush_test.csv", stringsAsFactors=TRUE)
mush_test

예측값
result7 <- predict(model2, mush_test[, -1])
result7

실제값
mush_test[1]
```

## ■ 나이브 베이즈 머신러닝 실습2 ( 영화 장르 예측 )

### # 1. 워킹 디렉토리에 영화(movie.csv)를 가져다 둔다.

데이터 소개: 나이, 성별, 직업, 결혼여부, 이성친구, 장르로 구성되어 있으며 장르가 정답(라벨)이 되는 컬럼

```
movie <- read.csv("movie.csv", header=T, stringsAsFactors=TRUE)
설명 : stringsAsFactors=TRUE를 써서 문자형 데이터를 팩터로 변환해야
머신러닝이 학습을 할 수 있는 데이터가 됩니다.
```

```
nrow(movie) #39
View(movie)
```

### #2. 한글명인 컬럼을 영어로 변환한다.

```
colnames(movie) <- c("age", "gender", "job", "marry", "friend", "m_type")
```

### #3. 39개의 행밖에 안되므로 38개의 데이터로 학습을 시키고 39번째 하나로 테스트한다.

```
train_data <- movie[1:38,]
test_data<- movie[39,]

View(test_data)
```

### #4. 훈련 데이터를 가지고 나이브 베이즈 모델을 생성한다.

```
library(e1071)
```

```
View(train_data[, 1:5]) # 훈련 데이터의 첫번째 컬럼부터 5번째 컬럼까지의 데이터만 학습시킴
6번째 컬럼이 정답인데 정답은 따로 제공함
```

```
model2 <- naiveBayes(train_data[, 1:5], train_data$m_type, laplace=0)
#설명 : naiveBayes(정답 없는 훈련데이터, 정답, 라플라스 값)
```

## #5. 테스트 데이터 test\_data를 가지고 예측을 해봄

```
result2 <- predict(model2, test_data[, 1:5])
result2
```

## ■ 나이브 베이즈 머신러닝 실습3 ( 독감 환자인지 아닌지 분류 )

# 1. flu.csv를 R로 로드하시오!

```
flu <- read.csv("flu.csv", header=T, stringsAsFactors=TRUE)
View(flu)
```

#데이터 설명:

patient\_id : 환자 번호

chills : 오한

runny\_nose : 콧물

headache : 두통

fever : 열

flue : 독감환자 여부 (정답 라벨 컬럼)

## ■ 5장. 의사결정트리

데이터들이 가진 속성들로부터 분할 기준 속성을 판별하고, 분할 기준 속성에 따라 트리 형태로 모델링하는 분류 예측 모델을 의사결정트리 모델이라고 한다.

( 빅데이터 기사 수제비 책 3-36 )

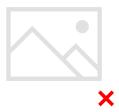
회귀분석, 의사결정트리는 현업에서 고객들과 데이터 분석가들이 선호하는 머신러닝 알고리즘이다. 신경망이 정확도는 훨씬 훌륭한데 신경망 내부가 블랙박스다 보니 설명이 안돼서 고객들이 이해를 못하는 경우가 많다. 의사결정트리와 회귀분석의 경우, 왜 이렇게 예측하고 분류했는지 설명이 잘 돼서 현업에서 선호하는 분류 모델이다.

예 : 빅데이터 기사 수제비 ( p3-37)

물고기인지 아닌지를 분류하는 의사결정트리 모델을 생성

의사결정 나무 기법은 분석의 대상을 분류함수를 활용하여 의사 결정 규칙으로 이루어진 나무 모양으로 그리는 기법입니다.

|   | 수중  | 지느러미 | 물고기 |
|---|-----|------|-----|
| 1 | YES | YES  | YES |
| 2 | NO  | NO   | NO  |
| 3 | NO  | YES  | NO  |
| 4 | NO  | NO   | NO  |



### ※ 의사결정트리 나무를 어떻게 만드는가?

부모마디의 순수도에 비해서 자식 마디들의 순수도가 증가하도록 자식 마디를 형성해 나가면서 만든다.

#### - 순수도란?

목표변수의 특정범주에 개체들이 포함되어있는 정도를 의미합니다.

의사결정트리 설명 PPT 참고 ( 카페 )

## ■ 화장품 구매에 영향을 크게 미치는 변수(컬럼)가 무엇인지 정보획득량을 구하시오!

```
skin <- read.csv("skin.csv", header=T, stringsAsFactors=T)
View(skin)

install.packages("FSelector")
library(FSelector)

wg <- information.gain(cupon_react ~., skin, unit='log2')
#설명: information.gain(라벨컬럼 ~ 모든컬럼, 데이터 프레임명, unit='log2')
#엔트로피 구할 수 : log2 쓸지 log10쓸지 정하기
print(wg)
```

결과:

|         | attr_importance |
|---------|-----------------|
| cust_no | 0.00000000      |
| gender  | 0.06798089      |
| age     | 0.00000000      |
| job     | 0.03600662      |
| marry   | 0.18350551      |
| car     | 0.02487770      |

설명 : 확인해보면 결혼유무가 가장 정보획득량이 높은것으로 확인된다.

## ■ 불순도를 측정하는 척도 3가지 ? (빅데이터 기사 시험 수제비 3-41)

1. 엔트로피 지수 : 열역학에서 쓰는 개념으로 무질서 정도에 대한 측도이고 엔트로피 지수의 값이 클수록 순수도가 낮다고 볼 수 있음.
2. 카이제곱 통계량 : 데이터의 분포와 사용자가 선택한 기대 또는 가정된 분포 사이의 차이를 나타내는 측정값

3. 지니지수 : 노드의 불순도를 나타내는 값

## ■ (의사결정트리 실습 1) 백화점 화장품 고객중에 구매가 예상이 되는 고객이 누구인가 ?

### 1. 의사결정 패키지인 C50 패키지를 설치한다.

```
install.packages("C50")
library(C50)
```

### ※ 의사결정나무 알고리즘 4가지 ( 빅데이터기사 3-43 )

1. CART : 나무의 분리기준 - 지니지수
2. C4.5 와 C5.0 : 나무의 분리기준 - 엔트로피 지수
3. CHAID : 나무의 분리기준 - 카이제곱 통계량과 F검정
4. QUEST : 나무의 분리기준이 카이제곱 통계량과 F검정

### 2. 백화점 화장품 고객 데이터를 로드하고 shuffle 한다.

```
skin <- read.csv("skin.csv", header=T ,stringsAsFactors = TRUE)
nrow(skin)
```

```
skin_real_test_cust <- skin[30,] #테스트용으로 따로 분리한다.
skin2 <- skin[1:29,] #29개의 데이터로 학습시켜서 의사결정나무를 생성
nrow(skin2)
```

```
skin_real_test_cust
```

결과 :

```
cust_no gender age job marry car cupon_react
30 30 female 40 YES YES NO YES
```

```
skin2 <- skin2[, -1] # 고객번호를 제외시킨다.
```

```
set.seed(11)
skin2_shuffle <- skin2[sample(nrow(skin2)),] # shuffle 시킴
```

### 3. 화장품 고객 데이터를 7대 3로 train 과 test 로 나눈다.

```
train_num <- round(0.7 * nrow(skin2_shuffle), 0)
skin2_train <- skin2_shuffle[1:train_num,]
skin2_test <- skin2_shuffle[(train_num+1) : nrow(skin2_shuffle),]
```

```
nrow(skin2_train) # 20
nrow(skin2_test) # 9
```

### 4. C50 패키지를 이용해서 분류 모델을 생성한다.

```
library(C50)
```

```
skin_model <- C5.0(skin2_train[, -6], skin2_train$cupon_react)
 ↑ ↑
```

라벨을 뺀 train 전체 data train 데이터의 라벨

> skin\_model

결과:

Call:

C5.0.default(x = skin2\_train[, -6], y = skin2\_train\$cupon\_react)

Classification Tree

Number of samples: 20

Number of predictors: 5 #스무고개의 질문이 5개

Tree size: 1

## 5. 위에서 만든 skin\_model 를 이용해서 테스테 데이터의 라벨을 예측하시오!

skin2\_result <- predict( skin\_model , skin2\_test[ , -6])

↑

라벨을 뺀 테스트 데이터 전체

## 6. 이원 교차표로 결과를 확인하시오 !

library(gmodels)

CrossTable( skin2\_test[ , 6], skin2\_result )

결과:

|                         |       | skin2_result |           |
|-------------------------|-------|--------------|-----------|
| skin2_test[, 6]         |       | NO           | Row Total |
| ----- ----- ----- ----- |       |              |           |
| NO                      | 2     | 2            |           |
|                         | 0.222 |              |           |
| ----- ----- ----- ----- |       |              |           |
| YES                     | 7     | 7            |           |
|                         | 0.778 |              |           |
| ----- ----- ----- ----- |       |              |           |
| Column Total            | 9     | 9            |           |
| ----- ----- ----- ----- |       |              |           |

설명 : CrossTable( x = 실제값, y = 예측값 )

전체 9개중에 겨우 2개 맞췄다. 정확도가 0.222밖에 안된다.

## ■ ( 의사결정트리 실습2 ) 아이리스 데이터를 가지고 의사결정트리 실습

c50패키지말고 ctree 함수를 사용할 수 있는 party 패키지를 설치함

이 실습은 빅데이터 기사 수제비의 3-44페이지의 내용을 좀 더 보강한 내용임.

### #1. party 패키지를 설치한다.

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

#2. 아이리스 데이터를 불러온다.

```
data(iris)
```

#3. 아이리스 데이터의 전체행수가 어떻게 되는지 확인

```
nrow(iris)
```

#4. 아이리스 데이터의 통계정보 확인

```
summary(iris)
```

#5. 아이리스 데이터의 라벨 컬럼 종류와 건수를 확인

```
table(iris$Species)
```

결과:

|        |            |           |
|--------|------------|-----------|
| setosa | versicolor | virginica |
| 50     | 50         | 50        |

설명: 꽃의 종류가 3가지고 각각 50개씩 구성되어 있음

#6. sample함수를 사용해서 shuffle도 하면서 데이터를 7대 3으로 나누고 replace=T를 사용해 복원추출 함

```
set.seed(11)
```

```
ind <- sample(2,nrow(iris), replace=T, prob=c(0.7,0.3))
```

#설명: 숫자 2는 데이터를 2개로 나누겠다는 것.

# nrow(iris)로 전체 행의 개수를 적어줌.

# replace=T로 복원추출 함

prob=c(0.7, 0.3)으로 7대 3으로 나누겠다고 지정

```
ind==1
```

```
ind==2 확인해보기
```

#7. 위에서 만든 ind==1 과 ind==2를 이용해 훈련 데이터와 테스트 데이터를 만듦. iris[ 행, 열 ]

```
traindata <- iris[ind==1,] #70%에 해당하는 데이터를 traindata로 구성
```

```
testdata <- iris[ind==2,] #70%에 해당하는 데이터를 testdata로 구성
```

```
nrow(traindata) #120
```

```
nrow(testdata) #30
```

#8. 의사결정트리 모델(나무) 생성

```
myformula <- Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
```

# 설명 : 변수 <- 라벨컬럼 ~ 독립컬럼1 + 독립컬럼2 + 독립컬럼 3 + ...

# 또는 myformula <- Species ~.으로 해도 됨

```
iris_ctree <- ctree(myformula,data=traindata)
```

설명: party 패키지의 ctree 함수를 이용해서 의사결정 모델 생성

#9. 예측결과를 확인

```
predict(iris_ctree)
```

#10. 예측결과와 실제 테스트 데이터의 정답과 비교해 봄

```
table(predict(iris_ctree),traindata$Species)
```

|            | setosa | versicolor | virginica |
|------------|--------|------------|-----------|
| setosa     | 42     | 0          | 0         |
| versicolor | 0      | 40         | 5         |
| virginica  | 0      | 1          | 32        |

설명 : 훈련데이터 120개를 가지고 예측한 결과를 보니 120중에 6개를 틀리고 다 맞췄다. 우리가 knn과 나이브 베이즈 실습할 때는 모델 생성 후에 그 모델로 테스트 데이터를 잘 맞추는지 확인했었는데 지금 이 실습은 훈련데이터부터 먼저 확인을 하고 있다. 지금 만든 의사결정트리 모델이 훈련 데이터를 잘 분류하는 모델이니 먼저 확인하고 있다.

### #11. 모델을 print 해서 스무고개의 질문이 어떻게 되는지 확인

```
print(iris_ctree)
```

결과:

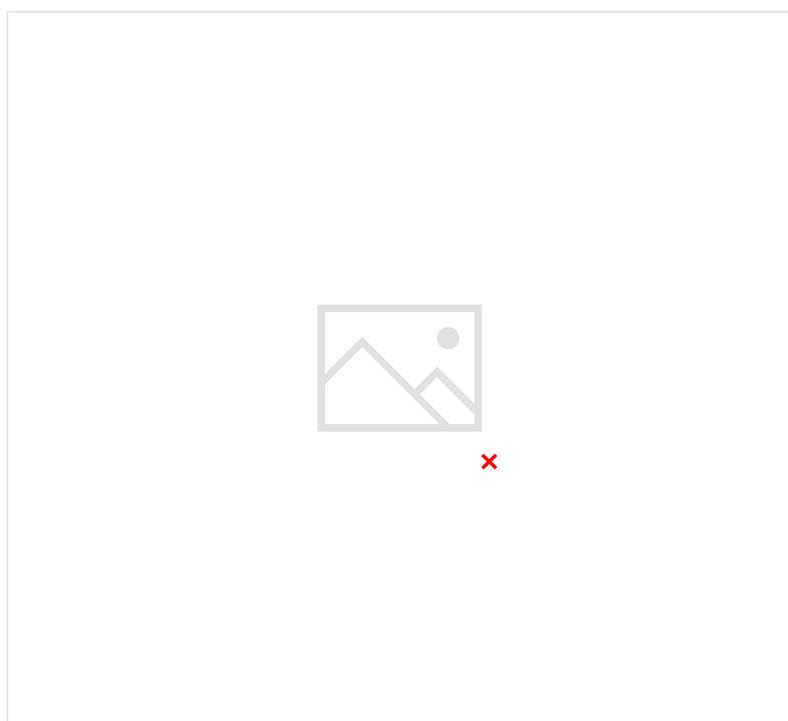
```
Number of observations: 120 --->
```

- 1) Petal.Length <= 1.9; criterion = 1, statistic = 112.117
- 2)\* weights = 42
- 1) Petal.Length > 1.9
- 3) Petal.Width <= 1.7; criterion = 1, statistic = 51.573
- 4) Petal.Length <= 4.8; criterion = 0.998, statistic = 12.024
- 5)\* weights = 37
- 4) Petal.Length > 4.8
- 6)\* weights = 8
- 3) Petal.Width > 1.7
- 7)\* weights = 33

### #12. 아이리스 데이터의 의사결정트리모델을 시각화 함.

```
plot(iris_ctree)
```

```
plot(iris_ctree,type="simple")
```



### #13. 테스트 데이터 30개를 예측하고 잘 맞췄는지 확인

```
testpred <- predict(iris_ctree, newdata=testdata)
```

```
table(testpred,testdata$Species)
```

```
testpred setosa versicolor virginica
```

```
setosa 8 0 0
```

```
versicolor 0 9 0
```

```
virginica 0 0 13
```

## ■ R 수업 복습

### 1. R을 왜 배워야하는가 ? 데이터 분석가가 되기 위해서

대기업 데이터 분석가

- 중소기업 / 스타트업 3년 경력
- 대학원 석/박사

포트폴리오, 빅데이터 기사 시험 준비

### 2. R 기본 문법 ( 데이터 검색 ) :

- R의 자료구조?
  - 1. vector : c(1,2,3)
  - 2. matrix : matrix() 함수
  - 3. array : array() 함수
  - 4. data frame : data.frame() 함수  
예 : emp[ 행, 열 ]
  - 5. list : list() 함수
- factor ( 팩터 ) ?  
vector + 순서

머신러닝 학습시 문자형 데이터를 factor로 변환해서 제공해줘야 에러가 나지 않음

**면접질문** : 머신러닝 학습시 문자형 데이터를 factor로 제공해줘야 하는 이유?

**답변** : 사람이 학습할 때도 뭐가 좋고 나쁜지 배우면서 학습 하듯이 ( 어떻게 해야 맛을 수 있고 틀릴 수 있는지) 기계에게도 데이터의 서열을 부여해서 제공함으로서 서열을 학습할 수 있게 하기 위해서입니다.

### 3. R을 활용한 머신러닝 1장 ( 기계학습 )

1. 지도학습 : knn, naivebayes, decision tree, 신경망, 회귀분석, 서포트 벡터 머신, 연관분석
2. 비지도학습: k-means
3. 강화학습

#### 4. R을 활용한 머신러닝 2장 ( 머신러닝 분석에 필요한 도구들 )

1. 수치형 데이터를 살펴보는 방법들 :
  - i. summary : 수치형 데이터의 통계 정보 확인
  - ii. histogram : 데이터의 분포가 정규분포 형태인지 확인
  - iii. box : 이상치에 민감한 데이터를 분석할 때 유용한데 50%에 해당하는 데이터에만 집중할 수 있도록 함
  - iv. 산포도 : 두개의 수치형 데이터의 상관성을 확인

#### 2. 명목형 데이터를 살펴보는 방법들

- i. 이원교차표 : 두개의 명목형 데이터의 유의성을 알아보는 툴
  - 1) 카이제곱 검정 값:  
이원 교차표를 볼 때 두개의 명목형 데이터의 연관성을 수치로 나타내는 통계분석

#### ※ 범주형 자료 분석 ( 빅데이터 기사 수제비 책 3-64 )

범주형 자료 분석은 종속변수가 하나이고 범주형인 데이터를 분석하여 모형의 유의성과 독립변수의 유의성을 알아보는 분석 방법이다.

범주형 자료분석은 독립변수와 종속변수의 척도에 따라 분석기법이 다르다.

| 독립변수 | 종속변수 | 분석방법               |
|------|------|--------------------|
| 범주형  | 범주형  | 이원교차표분석 ( 카이제곱검정 ) |
| 수치형  | 범주형  | 로지스틱 회귀분석          |

예 : 유방암 데이터 머신러닝 분석결과 확인시

머신러닝이 예측한 라벨 : B B M M M B M .....

테스트 데이터의 라벨 : B B B M B M .....

#### 5. R을 활용한 머신러닝 3장 ( knn 알고리즘 )

분류를 위해 사용하는 수학 공식 : 유clidean 거리공식

빅데이터 기사시험 수제비 책 3-59



#### 6. R을 활용한 머신러닝 4장 ( naiveBays 알고리즘 )

분류를 위해 사용하는 수학 공식 : 빅데이터 기사시험 수제비책 3-82 ~ 3-83

$$P(\text{스팸} | \text{비아그라}) = \frac{P(\text{비아그라} | \text{스팸}) * P(\text{스팸})}{P(\text{비아그라})}$$

↑ ↑  
 사후확률 주변우도

## 7. R을 활용한 머신러닝 5장 ( Decision tree 알고리즘 )

## 분류를 위해 사용하는 수학공식 : 1. 엔트로피 지수

2. 지니지수
  3. 카이제곱

## ■ sample 함수에 대한 설명

## 1. sample 함수로 데이터를 랜덤 추출하는 방법

```
set.seed(1)
sample(c(1,2), size=3, replace=TRUE)
결과: [1] 1 2 1
```

설명 : 데이터가 1과 2 두개가 있는 상태에서 replace=TRUE를 써서 복원추출 하는데 3번 하겠다는 뜻

■ wine 데이터를 의사결정 알고리즘을 이용해서 와인 데이터의 품질(type)을 분류하는 실험

## #1. party 패키지를 불러옴

```
library(party) #의사결정트리 알고리즘이 구현된 패키지
```

# 의사결정트리 시각화를 편하게 할 수 있는 패키지

`library(qmodels)` : 이원교차표를 보기 위한 패키지

## # 2. 데이터를 불러옴

```
wine<-read.csv("wine.csv")
View(wine)
종속 변수가 첫번째 컬럼인 type이고 범주형
그 외 모든 데이터가 연속형임을 확인
```

## # 3. 데이터에 대한 정보를 확인

```
nrow(wine) #178
ncol(wine) #14
summary(wine)
```

## # 4. 와인 데이터의 종속변수 컬럼의 종류와 건수를 확인

```
table(wine$Type)
세가지의 타입이 존재하며 t2의 빈도수가 가장 많음
```

## # 5. Type 컬럼을 펙터로 만들어줌 (chr 타입)

```
wine$Type <- factor(wine$Type,
 level=c("t1", "t2", "t3"),
 labels=c("T1", "T2", "T3"))
```

설명 : factor로 변환하려면 순서를 지정해야 하기 때문에 level이 필요한 것이고 labels는 데이터의 이름을 변경하는거라서 있어도 되고 없어도 됩니다.

## # 6. sample 함수를 사용하여 데이터 분할 및 무작위 정렬

( 가급적 7대3 비율이 되게끔 하려고 확률을 70%와 30%로 줌 )

```
set.seed(9)
ind <- sample(2, nrow(wine), replace=T, prob=c(0.7,0.3))
```

설명 : 위의 코드는 아래와 같다.

```
sample(c(1,2), 178, replace=T, prob=c(0.7, 0.3))
```

숫자 1과 2를 178번 복원 추출하는데 숫자 1을 70%의 확률로 추출하고 숫자 2는 30%의 확률로 추출

## # 7. 위에서 만든 ind로 훈련데이터와 테스트 데이터를 만들

```
train_data<- wine[ind==1,] # 대략 70%에 해당하는 데이터
test_data<-wine[ind==2,] # 대략 30%에 해당하는 데이터
```

```
nrow(train_data) # 123
nrow(test_data) # 55
```

설명 : 훈련데이터 123개를 가지고 기계가 학습을 할 것이고 테스트 데이터 55개를 가지고 기계가 시험을 볼 것임 그래서 데이터를 훈련 데이터와 테스트 데이터로 나눈 것임.  
평가를 해봐야 학습을 잘 했는지 확인할 수 있기 때문

## # 8. 의사결정트리 모델을 생성

```
myformula <- Type ~ .
wine_ctree <- ctree(myformula, data=train_data)
```

ctree( 종속변수 ~., data= 데이터 프레임 )

설명: ctree는 기계에게 의사결정트리 알고리즘으로 데이터를 분류하게끔 구현된 코드가 있는 R 함수( 어느 외국인 개발자가 만듦 ), 개발자가 함수를 만들 때 어떻게 사용하라고 메뉴얼도 같이 만들어서 cran이라는 사이트에 올려 놓음

?ctree 해보기

#### # 9. 모델이 예측한 결과를 확인

```
predict(wine_ctree)
```

#### # 10. 예측 결과와 훈련 데이터의 정답과 비교

```
table(predict(wine_ctree), train_data$Type)
```

#### # 11. 어떠한 질문을 기준으로 나뉘는지 확인

```
print(wine_ctree)
print(wine_ctree, type='simple')
```

#### # 12. 검증용 데이터를 예측하고 잘 맞췄는지 확인

```
test_pred <- predict(wine_ctree, newdata=test_data)
table(test_pred, test_data$Type)
```

#### # 13. 교차검정표와 정확도 계산

```
g2 <- CrossTable(test_data$Type, test_pred)
```

설명 : CrossTable( 테스트 데이터의 진짜 정답, 테스트 데이터를 기계가 예측한 결과)

```
sum(g2$prop.tbl*diag(3)) # 정확도: 0.8363636
```

설명: diag(3)은 대각선이 1인 3x3행렬을 출력하는 코드인데 g2\$prop.tbl 와 diag(3)을 곱하면 대각선 데이터만 남게 됩니다. 남은 3개의 데이터를 sum을 사용해서 모두 더했습니다.

```
plot(wine_ctree)
```

## ■ (의사결정트리 실습 3) 은행대출채무를 불이행할 것 같은 고객이 누구인가?

의사결정트리의 활용 :

은행에 대출 신청을 했는데 대출 거부를 당했으면 은행에서는 당사자에게 설명을 해줘야 함

1단계 : 데이터 수집

```
credit <- read.csv("credit.csv", string stringsAsFactor=TRUE)
str(credit)
```

데이터 설명 : 독일의 한 신용기관에서 얻은 대출정보가 들어있다.

신용 데이터셋은 100개의 대출 예시와 대출금액, 대출 신청자의 특성을 나타내는 일련의 수치 특징과 명목특징을 포함하고 있다.

라벨( 정답 )컬럼 : default -----> yes : 대출금 상환 안함  
no : 대출금 상환

```
prop.table(table(credit$default))
```

```
no yes
0.7 0.3
```

대출금을 상환한 고객이 70%이고 안한 고객이 30%를 이루고 있다.

checking\_balance 컬럼 : 예금계좌

saving\_balance 컬럼 : 적금계좌

amount 컬럼 : 대출금액 ( 250 마르크 ~ 18424 마르크 )

100 마르크 : 우리나라 돈 6~7만원

**예제1. amount 컬럼의 데이터를 히스토그램 그래프로 그리시오!**

```
r()
```

**예제2. amount 컬럼에 대한 통계정보를 확인하시오!**

```
summary(credit$amount)
```

결과:

| Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max.  |
|------|---------|--------|------|---------|-------|
| 250  | 1366    | 2320   | 3271 | 3972    | 18424 |

설명 : 최소 대출금액은 250마르크(1750만원)이고 최대 18424 마르크까지 구성되어 있으며 평균이 3271마르크 입니다.

**은행의 목표** : 과거의 데이터를 분석해보니 대출금 상환 불이행자가 30%나 되어서 앞으로는 30% 이내로 떨어뜨리게끔 하는게 은행의 목표다. 거기에 맞는 머신러닝 모델을 생성해야 함.

## 2단계 : 데이터 탐색

- \* 데이터가 명목형 데이터인지 수치형 데이터인지 확인
- ```
str( credit )
View( credit )
```

설명 : 전부 수치형으로만 되어있다면 knn을 사용하면 되고 전부 명목형 데이터로만 구성되어 있다면 naivebayes를 쓰면 되는데 위의 데이터는 문자와 숫자가 섞여 있으므로 decision tree 를 써서 기계학습을 시킨다.

* 데이터를 shuffle 시킨다.

```
set.seed(123)
credit_shuffle <- credit[ sample( nrow( credit ) ), ]
train_num <- 0.9 * 1000
credit_train <- credit_shuffle[1:train_num, ] #1번부터 900번째 행까지
```

#901번부터 1000번까지는 테스트 데이터로 생성

```
credit_test <- credit_shuffle[ (train_num + 1) : nrow(credit_shuffle), ]
```

```
nrow(credit_train) #900
nrow(credit_test) #100
```

* 데이터를 9대 1로 나눈다. (훈련데이터 9 , 테스트 데이터 1)

3단계 : 모델 생성

```
library( C50 ) #엔트로피지수를 이용해서 순수도를 구하고 분류하는 패키지
str( credit_train ) #라벨컬럼이 몇번째 컬럼인지 확인
ncol( credit_train ) # 컬럼개수 확인, 총 17개의 컬럼이 있고 17번째가 라벨이다.
```

문법 : credit_model <- C5.0(라벨을 뺀 나머지 데이터, 라벨 컬럼 데이터)

```
credit_model <- C5.0( credit_train[ , -17], credit_train[ , 17] )
또는
credit_model <- C5.0( credit_train[ , -17], credit$default )
```

설명 : 900개의 데이터로 학습한 모델을 생성하였다.

4단계 : 모델 평가

* 위에서 만든 모델을 이용해 테스트 데이터 100개를 예측한다.

```
예측결과변수 <- predict( 모델, 라벨을 뺀 테스트 데이터 )
```

```
credit_result <- predict( credit_model, credit_test[ , -17] )
table( credit_result )
```

결과:

```
no yes  
75 25
```

설명 : 머신러닝이 예측한 결과는 75명이 채무이행을 했고 25명이 불이행했다고 나온다. 그렇다면 정답은 어떻게 되었을까?

```
table(credit_test[, 17] )
```

결과:

```
no yes  
65 35
```

설명 : 실제 정답과 예측이 10명의 차이를 보이고 있다.

* 머신러닝이 예측한 결과와 실제 정답을 서로 비교한다.

```
table( credit_test[, 17], credit_result )
```

결과:

```
credit_result no yes  
no 55 10  
yes 20 15
```

설명 : 총 100명의 고객들 중에서

채무이행할거라 예측했는데 채무 이행 고객이 55명 --> o

채무이행할거라 예측했지만 불이행한 고객이 10명 --> x

채무 불이행할거라 예측했지만 채무 이행한 고객이 20명 ---> x

채무 불이행할거라 예측했는데 채무 불이행한 고객이 15명 ---> o

채무이행할거라 예측했는데 불이행한 고객이 10명이나 되므로 우리는 이 부분을 0으로 만들 수 있도록 모델의 성능을 개선해야합니다.

```
library(gmodels)  
x <- CrossTable( x= credit_test[, 17], y=credit_result )
```

설명 : CrossTable(x = 실제값, y=예측값)

정확도 확인

```
print( sum(x$prop.tbl * diag(2)) ) #0.7
```

5단계 : 모델 성능 개선

의사결정트리의 하이퍼 파라미터를 조정한다.

하이퍼 파라미터 2개 :

1. trials : trials는 의사결정 나무의 개수를 결정하는 파라미터 (0~ 최대 100개)

- seed 값 : shuffle할 때 섞는 규칙을 결정하는 파라미터
정수형 범위안에서 다 줄 수 있음

```
credit_model2 <- C5.0( credit_train[ , -17], credit_train[ , 17], trials=100 )
credit_result2 <- predict( credit_model2, credit_test[ , -17] )
x <- CrossTable( x= credit_test[ , 17], y=credit_result2 )
print( sum(x$prop.tbl * diag(2)) ) #0.72
```

정확도를 90%이상 올려야하므로 의사결정트리 말고 다른 분류 알고리즘을 사용해야 한다.

knn, naivebayes, decision tree, 신경망, 서포트 벡터 머신, 랜덤 포레스트

■ 최적의 seed값을 루프문으로 돌려서 알아내는 방법

```
library(party)
```

#wine을 데이터를 불러온다.
wine<-read.csv('wine.csv',stringsAsFactors = T)

#와인 데이터 라벨 컬럼 종류와 건수를 확인합니다.
summary(wine\$Type)

결과 :

```
t1 t2 t3
59 71 48
```

#repeat문 돌려서 seed값찾기

```
i <- 1 # 파이썬 코드로 치면 cnt=1 이란 코드
```

#repeat 안에 있는 코드를 계속 반복시키겠다. break를 만날 때 까지

```
repeat {
  set.seed(i)
  i <- i+1
  ind <- sample(2,nrow(wine),replace=T,prob=c(0.7,0.3))
```

ind==1과 ind==2를 이용해서 훈련데이터와 테스트 데이터를 만든다.
traindata <- wine[ind==1,] #70%의 데이터로 훈련데이터
testdata <- wine[ind==2,] #30%의 데이터로 테스트데이터

```
# 의사결정트리 모델(나무)를 생성한다.  
myformula <- Type ~ .  
  
# party패키지의 ctree함수를 이용해서 의사결정모델을 생성한다.  
wine_ctree <- ctree(myformula,data=traindata)  
  
# 예측결과를 확인한다.  
#predict(wine_ctree)  
  
# 예측결과와 실제 테스트 데이터의 정답과 비교한다.  
#table(predict(wine_ctree),traindata$Type)  
  
# test데이터를 예측하고 확인한다.  
testpred <- predict(wine_ctree, newdata=testdata)  
library(gmodels)  
g3<-CrossTable(testdata$Type, testpred)  
x<- (g3$prop.tb[1] + g3$prop.tb[5]+g3$prop.tb[9])  
  
if(x==1) break  
print(i) }  
print(i)
```

■ 5장. oneR 알고리즘 (p. 227)

규칙 기반 알고리즘 : 1. oneR 알고리즘
2. Riper 알고리즘

■ 1R 알고리즘 :

"하나의 사실(조건) 만 가지고 간단하게 분류하는 알고리즘"

하나의 사실만 가지고 분류하다보니 간단하지만 오류가 많다.

예 : 가슴통증의 유무에 따라 심장질환이 있는지 분류하고자 한다면 가슴통증 하나만 보고 심장질환이 있다고 분류하기에는 오류가 많아진다.
왜냐하면 식도염, 폐질환도 가슴통증이기 때문이다.

■ Riper 알고리즘 :

"복수개의 사실(조건)을 가지고 분류하는 알고리즘"

예 : 가슴통증이 있으면서 호흡곤란이 있으면 심장질환이다.

알고리즘이 데이터를 보고 패턴(조건)을 발견한다.

예 : 독버섯 분류 조건을 발견한 알고리즘 : p. 245

■ 독버섯 데이터와 식용버섯 데이터를 oneR 알고리즘으로 분류하는 실습 (p. 237)

1. 버섯 데이터를 R로 로드한다.

```
setwd("d:\\data")
mushroom <- read.csv("mushrooms.csv", stringsAsFactors=T)
```

데이터 설명 : UCI 머신러닝 저장소에서 제공하는 데이터이며 23종의 버섯과 8124개의 버섯샘플에 대한 정보가 포함되어 있다. 버섯샘플 22개의 특징은 갓모양, 갓색깔, 냄새, 주름크기와 색, 줄기모양, 서식지와 같은 특징이 있다.

2. mushroom 데이터를 훈련 데이터와 테스트 데이터로 나눈다 (훈련 데이터 75%, 테스트 데이터 25%)

```
set.seed(11)
```

```
dim(mushroom)
```

```
train_cnt <- round( 0.75 * dim(mushroom)[1])
train_index <- sample(1:dim(mushroom)[1], train_cnt, replace=F)
# 설명: 1~8024까지의 숫자를 6093번 비복원추출합니다.
```

```
mushroom_train <- mushroom[train_index, ] #6092
mushroom_test <- mushroom[-train_index, ] #2031
```

3. 규칙기반 알고리즘인 **oneR** 을 이용해서 독버섯과 일반버섯을 분류하는 모델을 생성한다.

```
install.packages("OneR")
library(OneR) # 한가지 조건만 가지고 분류하는 알고리즘
# 장점 : 간단하다.
# 단점 : 정확하게 분류하지 못한다.
```

```
model1 <- OneR(type~, data=mushroom_train)
# 설명 : oneR( 라벨컬럼~ 모든컬럼, data = 데이터 프레임명 )
```

```
model1
```

결과 :

Rules:

```
If odor = almond then type = edible
If odor = anise then type = edible
If odor = creosote then type = poisonous
If odor = fishy then type = poisonous
If odor = foul then type = poisonous
If odor = musty then type = poisonous
If odor = none then type = edible
If odor = pungent then type = poisonous
If odor = spicy then type = poisonous
```

```
summary(model1)
```

```
# 설명 : 버섯 냄새 하나만 가지고 버섯은 분류하고 있음
```

```
# 책 p.242 : 독버섯의 냄새는 물고기냄새, 악취, 쿠키한, 톡쏘는, 매운, 크레오소트 같은 냄새가 나며
식용버섯은 냄새가 없거나 아몬드 냄새가 나는것을 알고리즘 데이터를 보고 발견했다.
```

4. 위에서 생성한 모델을 가지고 테스트 데이터로 결과를 확인한다.

```
result1 <- predict( model1, mushroom_test[ , -1] )
# 설명 : -1을 사용해서 테스트 데이터에서 정답컬럼을 제외했다.
```

```
library(gmodels)
```

```
CrossTable( mushroom_test[ , 1], result1)
```

■ oneR 알고리즘이 어떻게 냄새로 버섯을 식용과 독버섯을 분류하는가?

" 독버섯 데이터에서 정보획득량이 가장 높은것이 odor(버섯향) 이어서 odor로 분류하였다 "

예제1. 독버섯 데이터의 컬럼들의 정보획득량을 확인하세요!

```
library( Fselector )
w1 <- information.gain( type ~ ., mushroom, unit="log2" )
w1
```

■ 규칙기반 알고리즘 실습(Ripper 알고리즘) p. 243

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

JRip 함수는 oneR 패키지처럼 한가지 조건만 가지고 분류하는 규칙기반이 아닌 여러개의 조건을
가지고 분류하는 분류 알고리즈다.

```
model2 <- JRip(type~ ., data=mushroom_train)
model2 # R책 p. 245 참조
```

(odor = foul)

설명 : 냄새가 약취이면 이 버섯 종은 독이 있다.

(gill_size = narrow) and (gill_color = buff)

설명 : 주름크기가 좁고 주름색이 담황색이면 이 버섯 종은 독이 있다.

(gill_size = narrow) and (odor = pungent)

설명 : 주름크기가 좁고 냄새가 톡 쏘면 이 버섯 종은 독이 있다.

(odor = creosote)

설명 : 냄새가 크레소토이면 이 버섯 종은 독이 있다.

(spore_print_color = green)

설명 : spore_print_color가 녹색이면 이 버섯 종은 독이 있다.

(stalk_surface_below_ring = scaly) and (stalk_surface_above_ring = silky)

(habitat = leaves) and (cap_surface = scaly) and (population = clustered)

(cap_surface = grooves)

위의 7가지 조건 외에 나머지는 다 식용이다.

-> type=edible

설명 : 데이터를 보고 위와 같은 규칙을 컴퓨터가 알아냈다는 것만으로 사람이 할 일을 많이 줄여 줄 수 있다.

summary(model2)

작은 이원교차표가 하나 보임

a	b	<-- classified as
3125	0	a = edible
0	2968	b = poisonous

훈련 데이터에 대한 정확도가 100%에 해당하는 결과를 보이고 있다.

result2 <- predict(model2, mushroom_test[, -1])

```
library(gmodels)
CrossTable( mushroom_test[, 1], result2)
```

		result2		Row Total
mushroom_test[, 1]		edible	poisonous	
edible	1083	0	1083	1083
	442.493	505.507		
	1.000	0.000	0.533	
	1.000	0.000		
	0.533	0.000		
poisonous	0	948	948	948
	505.507	577.493		
	0.000	1.000	0.467	
	0.000	1.000		
	0.000	0.467		
Column Total	1083	948	2031	2031
	0.533	0.467		

■ 6장. 회귀분석

회귀분석은 하나의 변수가 나머지 다른 변수들과의 선형관계를 갖는가의 여부를 분석하는 방법으로 하나의 종속변수(예측하고자 하는 값)와 독립변수 사이의 관계를 명시하는 것을 말한다.

예 : 집값에 가장 영향을 주는 요소가 무엇인가?

- 독립변수 : 종속변수에 영향을 주는 변수(평수, 역세권, 학군, ...)
- 종속변수 : 서로 관계를 가지고 있는 변수들 중에서 다른 변수에 영향을 받는 변수(집값)

■ 최소 제곱 추정법(p 257)

최적의 a(기울기)와 b(절편)를 결정하기 위해서는 최소제곱으로 알려진 추정기법을 사용한다.
실제값과 예측값 사이의 수직 직선이 오차(잔차)를 제곱해서 구한 총합을 알아야 한다.

■ 회귀분석 유형 (빅데이터 기사시험 수제비책 3-29)

1. 단순회귀 : 독립변수가 1개이며, 종속변수와의 관계가 직선
2. 다중회귀 : 독립변수가 k개이며, 종속변수와의 관계가 선형 (1차함수)
3. 다행회귀 : 독립변수와 종속변수와의 관계가 1차함수 이상인 관계
4. 곡선회귀 : 독립변수가 1개이며, 종속변수와의 관계가 곡선
5. 로지스틱 회귀 : 종속변수가 범주형 (2진변수)인 경우
6. 비선형 회귀 : 회귀식의 모양이 미지의 모수들의 선형관계로 이루어져 있지 않은 모형

$$y = ax + b \rightarrow \text{회귀계수 : } a \text{와 } b \text{ (기울기와 절편)}$$

회귀계수를 최소제곱법을 사용해서 추정한다.

■ R함수 lm을 이용한 단순회귀분석 실습1

" 탄닌 함유량과 애벌레 성장간의 관계에 대한 회귀식 도출하기 "

#1. 데이터를 로드한다.

```
reg <- read.table("regression.txt", header = T )
reg
```

#2. 데이터를 시각화한다.

```
attach(reg)
plot( growth ~ tannin, data=reg, pch=21, col='blue', bg='blue' )
#설명 : plot( y ~ x, data=데이터프레임 )
```



#3. 회귀분석을 통해 회귀계수인 기울기와 절편을 구한다.

```
model <- lm( growth ~ tannin, data=reg )  
# lm : 회귀분석함수, growth = 종속변수, tannin = 독립변수  
model
```

결과 :

Coefficients:

(Intercept)	tannin
11.756	-1.217

#4. 2번에서 시각화한 산포도 그래프에 회귀직선을 겹쳐서 그린다.

```
abline( model, col='red' )
```



#5. 그래프 제목을 회귀 직선의 방정식으로 출력되게 한다.

```
model$coefficients[2]  
model$coefficients[1]  
  
title( paste( '=', model$coefficients[2], 'x 탄닌 + ', model$coefficients[1] ) )
```

단순회귀분석 -----> 상관관계 -----> 다중회귀분석



■ 다중공선성 (variance inflation factor)

회귀분석에서 사용된 모형의 일부 설명변수(독립변수) 가 다른 독립변수와의 상관정도가 높아 데이터 분석시 부정적인 영향을 미치는 현상을 말한다.

두 독립변수들끼기 서로에게 영향을 주고 있다면 둘 중 하나의 영향력을 검증할때 다른 하나의 영향력을 완전히 통제할 수 없게 된다.

아파트가격, 평수, 역과의 거리

예: 학업성취도, 일평균음주량, 혈중 알코올 농도

↑
종속변수

음주가 학업성취도에 미치는 영향을 알아보려고 회귀분석을 하려고한다. 일평균 음주량과 혈중 알코올 농도는 서로 아주 강한 상관정도를 보인다.

실제로 x_1 과 x_2 의 값이 증가 또는 감소 할수록 y 값이 증가 또는 감소할 것인데 이중 하나는 굉장히 불안정한 계수값을 보이게 된다.

공선성은 두개의 독립변수들 간의 관계를 의미하는데 예를들어 두개의 독립변수들 간의 상관관계 계수가 1이면 완전한 공선성을 보인다고 하고, 계수가 0 이면 전혀 공선성이 없음을 의미한다.

특히 3개 이상의 변수들간의 관계를 다중 공선성이라한다. 한 독립변수가 종속변수에 대한 설명력이 높더라도 (다중) 공선성이 높으면 설명력이 낮은것처럼 나타난다.

다중공선성을 알아보기 위한 가장 간단한 방법은 독립변수들간의 상관관계를 조사하는 것이다. 독립 변수들 간의 높은 상관관계(일반적으로 0.9이상)은 공선성을 판단하는 지표이다.

공선성을 보다 엄격하게 점검하려면 팽창계수(VIF)를 확인하면 된다.

현업기준 : 팽창계수(VIF)가 보통 10보다 큰것을 골라내고 엄격하게는 5보다 큰 것을 골라낸다.
느슨하게 하려면 15또는 20으로 주로 골라낸다.

다중공선성 확인 실습 :

```
install.packages( "car" )
library(car)
data(Boston, package="MASS" )
Boston
```

데이터 설명 : <https://cafe.daum.net/oracleoracle/SDMs/68>

```
model <- lm( medv ~. , data=Boston )
```

```
vif(model)
vif(model) > 10 #다중공선성을 보이는 변수들 확인
```

결과 :

```
crim      zn    indus   chas    nox     rm     age     dis     rad     tax   ptratio   black   lstat
 FALSE    FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE   FALSE
 FALSE
```

설명 : 전부 FALSE여서 다중 공선성을 보이는 컬럼은 없다.

■ 나이브베이즈 수학식을 복습하며 빅데이터 기사 시험 대비하기

면접문제

전력 사용량 예측 ----> 회귀분석으로 예측을 한다.

단순회귀 분석 ---> 상관관계 분석 ---> 다중회귀 분석

■ 상관관계 데이터 분석 p. 260

두 변수 간의 상관관계는 변수들의 관계가 직선에 가깝게 따르는 정도를 나타내는 숫자이다. 상관관계는 -1에서 +1 사이의 범위에 있다.

극값은 완벽한 선형관계를 나타내는 반면, 0에 가까운 상관관계는 선형관계가 없음을 나타낸다.

피어슨 상관관계 : 두 변수의 공분산을 표준편차의 곱으로 나눈 값으로 상관계수를 구한다.

※ 빅데이터 시험에서의 상관계수 이론 (수제비 책의 3-72)

1. **피어슨 상관계수** : - 등간척도나 비례척도의 데이터에서 두 변수의 공분산을 두 변수의 표준편차의 곱으로 나눈 값이다.
 - 두 변수간의 선형관계의 크기를 측정하는 값으로 비선형적인 상관관계를 나타내 못한다.
 - 피어슨 상관계수는 모집단, 표본에 적용할 수 있다.
2. **스피어만의 상관계수** : - 스피어만의 상관계수는 두 변수간의 비선형적인 관계도 나타낼 수 있는 값이다.
 - 두 변수를 모두 순위로 변환시킨 후 두 순위 사이의 스피어만 상관계수를 구한다.

■ 다중선형회귀분석 p. 262

단순 선형회귀 분석의 목적이 하나의 독립변수만을 가지고 종속변수를 예측하기 위한 회귀모형을 만들기 위한 것이었다면 다중 회귀분석의 목적은 여러개의 독립변수들을 가지고 종속변수를 예측하기 위한 회귀모형을 만드는 것이다.

예 : 집값에 영향을 미치는 요소가 단순히 평수만 있는게 아니다.

집값 <---- 평수, 교통, 학군, 범죄율, 총수, 방의개수, 한강뷰 ...

질문1 : 집값에 영향을 미치는 요소가 위의 여러가지 독립변수들 중에서 무엇인가?

질문2 : 집값을 예측하기 위한 다중회귀식은 어떻게 되는가?

단순선형 회귀식 : $y = ax + b$

예 : 태양열전지 만드는데 있어서 전력효율을 최대화 할 수 있는 가장 좋은 태양열 전지 재료 조합이 어떻게 되는가?

$$\text{전력량} = a_1 \times \text{재료1} + a_2 \times \text{재료2} + \dots + b$$

■ 책 264페이지에 나오는 다중선형 회귀식의 회귀계수를 도출하는 방법

다중선형 회귀식을 이해하기 위해 사전에 알아야하는 내용?

1. 전치행렬
 2. 단위행렬
 3. 역행렬
 4. 의사역행렬

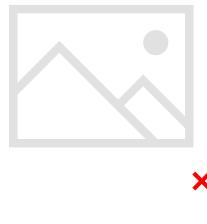
■ 1. 전치행렬 ? 행과 열을 교환하여 얻은 행렬



■ 2. 단위행렬 ? 주 대각선의 원소가 모두 1이며 나머지 원소는 모두 0인 정사각 행렬



■ 3. 역행렬 ? 원래 행렬과 곱했을 때 단위행렬이 되는 행렬



■ 4. 의사 역행렬 (Pseudo inverse) ?

기본적으로 역행렬은 정방행렬일 때만 구할 수 있다.

직사각형 행렬이면 역행렬을 구할 수 없다.

직사각형 행렬의 역행렬을 구하려면 의사역행렬을 구해야만 한다.

#4.1 정방행렬일 때?



```
a <- matrix( c(1,2,3,4), nrow=2, ncol=2, byrow=T )
solve(a)
```

#4.2 정방행렬이 아닐때?



■ 분류할 때는 데이터를 정규화를 하고 모델을 학습 시켰는데 회귀분석에서도 정규화를 해야하는가?

미국 보험회사의 미국 국민 의료비를 가지고 보험 비용을 산정하는 예

1. 정규화를 하는 경우 :

보험비용에 가장 영향을 크게 미치는 변수가 무엇인지 확인할 때
종속변수에 대한 독립변수의 영향도를 확인하고 싶을 때

2. 정규화를 안 해야하는 경우 :

나이가 한살 늘어날 때 보험료가 얼마나 인상되어야 하는지 예측해야 할 때
부양가족이 한명이 더 늘어날 때마다 보험료가 얼마나 인상되어야 하는지 예측해야 할 때

■ 표준화와 정규화의 차이

1. 표준화 : 평균이 0이고 표준편차가 1인 데이터 분포로 구성하는 것

예 : scale 함수

2. 정규화 : min/max 정규화인데 데이터를 0~1사이의 숫자로 변환하는 것 입니다.

예 : minmax 함수

머신러닝에서는 표준화보다 정규화가 더 좋은 결과가 나와서 정규화를 더 선호한다.

■ 다중 회귀분석 실습(미국 국민의 의료비 데이터를 예측) p 268

■ 보험회사의 보험료 산정에 미치는 요소 분석

주제 : 보험회사에서 보험료 산정에 도움이 될 수 있도록 미국민의 의료비를 예측하는 회귀모델 생성하기

1. 보험 데이터를 R로 로드한다.

```
insurance <- read.csv("insurance.csv")
```

```
head(insurance)
```

데이터 소개: 미국 환자의 가상 의료비가 들어있는 모의 데이터셋입니다.

이 데이터는 미국 통계국의 인구 통계를 이용해 생성되었으며 대개 실제 질병을 반영합니다. 현재 의료보험에 등록된 1,338명의 수의자 예시가 들어있으며 각 예시는 환자의 특성과 해당 연도에 의료보험에 청구된 전체 의료비를 나타내는 특징으로 구성되어 있습니다.

age: 나이

sex : 성별

bmi : 체질량 비만 지수

children : 부양가족수

smoker : 흡연여부

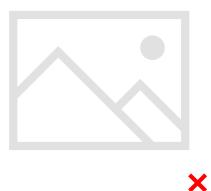
region : 사는 지역 (북동, 남동, 북서, 남서)

※ 결측치가 있는지 확인합니다.

```
colSums ( is.na(insurance) )
```

※ 종속변수 데이터가 정규성을 띠는지 히스토그램으로 확인해본다.

```
hist( insurance$expenses )
```



오른쪽으로 꼬리가 긴 분포를 보여준다. 대다수의 사람들의 의료비는 0~\$15,000 달러 사이에 있다. 이 분포는 선형회귀에서는 이상적이지 않지만 미리 알고 있으면 나중에 모델을 설계할 때 도움이 된다.

#2. 정규화 작업을 수행한다.

```
normalize <- function(x) {  
  return ( (x-min(x)) / (max(x) - min(x)) ) }
```

```
insurance_n <- as.data.frame(lapply(insurance[,c(1,3,4,7)], normalize))  
summary(insurance_n)
```

결과:

age	bmi	children	expenses
Min. :0.0000	Min. :0.0000	Min. :0.000	Min. :0.00000
1st Qu.:0.1957	1st Qu.:0.2776	1st Qu.:0.000	1st Qu.:0.05776
Median :0.4565	Median :0.3881	Median :0.200	Median :0.13185
Mean :0.4610	Mean :0.3953	Mean :0.219	Mean :0.19392
3rd Qu.:0.7174	3rd Qu.:0.5040	3rd Qu.:0.400	3rd Qu.:0.24770
Max. :1.0000	Max. :1.0000	Max. :1.000	Max. :1.00000

#모두 다 0과 1사이

설명 : 종속변수에 어떤 독립변수가 영향이 큰지 확인하기 위해서 정규화를 한다.

#3. 독립변수와 종속변수간의 상관관계 분석

```
cor( insurance[, c("age","bmi","children","expenses")] )
```

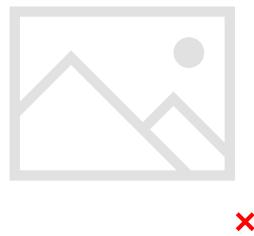
설명 : 회귀분석하기 전에 상관관계를 확인해야하는 이유는 독립변수들간의 강한 상관관계를 보이게 되는 다중공선성 여부를 확인해야 회귀분석 결과에 가장 중요한 결정계수에 결과를 신임할 수 있기 때문

	age	bmi	children	expenses
age	1.0000000	0.10934101	0.04246900	0.29900819
bmi	0.1093410	1.00000000	0.01264471	0.19857626
children	0.0424690	0.01264471	1.00000000	0.06799823
expenses	0.2990082	0.19857626	0.06799823	1.00000000

설명 : 눈에띄게 아주 강한 상관관계를 보이는 것은 없지만 일부 눈에 띄는 연관성이 있다. 예를 들어 age와 bmi는 약한 양의 상관관계가 있어서 나이가 들수록 몸무게가 증가하는 경향이 있다. age와 expenses를 보면 양의 상관관계를 보이고 있어서 나이가 들수록 의료비가 증가하는 경향이 있다.

※ 상관관계를 시각화하기

```
library( psych )
pairs.panels( insurance[ , c('age', 'bmi', 'children', 'expenses' ) ] )
```



산포도에 있는 달걀모양의 객체는 상관관계 타원형으로 상관관계의 강도를 시각화 한것이다.
타원형 중심에 있는 점은 x와 y축 변수에 대한 평균값 지점을 나타낸다.

타원이 늘어질수록 강한 상관관계

타원이 완벽한 둥근 달걀모양에 가까울수록 약한 상관관계

* 독립변수들의 상관관계를 통해서 알 수 있는 점?

1. 나이가 많을 수록 의료비가 더 많이든다.
2. 나이가 많을수록 비만지수가 더 높았다.
3. 중년무렵부터 부양 가족수가 최고점이 된다.

#4. 회귀함수인 lm 을 이용해서 독립변수들의 회귀모수를 확인한다 (기울기)

```
m5 <- lm( expenses ~ age + children + bmi +smoker +region, data=insurance)
```

또는

```
attach(insurance)
```

```
lm(expenses ~ ., data= insurance)
```

결과 (정규화 전 데이터 사용):

Coefficients:

(Intercept)	age	sexmale	bmi	children	smokeryes	regionnortheast
-11941.6	256.8	-131.4	339.3	475.7	23847.5	-352.8
regionsoutheast	regionsouthwest					
-1035.6	-959.3					

- 설명 : 1. 나이가 일년씩 더해질 때마다 평균적으로 의료비가 256.8 달러가 증가될 것으로 예상됨
2. 자녀가 한명씩 추가될 때마다 475.7 달러가 추가될 것으로 예상된다.
3. 비만지수(bmi)의 단위가 증가할 때마다 연간 의료비가 평균 339.3달러 증가될 것으로 예상됨
4. 더미변수를 자동으로 추가해서 변수 값의 상대적 추정을 하고 있다.

sexmale -131.4 ---> 남성은 여성에 비해 매년 의료비가 131.4달러가 적게 든다고 예상
smokeryes 23847.5 ---> 흡연자는 비흡연자보다 매년 평균 의료비가 23,847.5달러가 더 듈다

northeast에 비해 northwest는 의료비가 연간 평균 352.8달러 덜 들고
southeast는 의료비가 연간 평균 1035.6달러 덜 들고
southwest는 의료비가 연간 평균 959.3 달러 덜든다.

■ 머신러닝 데이터 분석 5단계

1. 데이터 수집 및 데이터 설명
2. 데이터 탐색 및 분석
3. 머신러닝 모델 훈련
4. 모델 성능 평가
5. 모델 성능 개선

■ 4. 모델 성능 평가

앞에서 분류를 할 때 모델의 성능 평가는 "정확도"로 했습니다.

회귀분석일때는 회귀모델의 성능평가를 무엇으로 하나요 ? "결정계수"

```
model2 <- lm(expenses ~ ., data= insurance)
```

```
summary(model2)
```

▣ 결과 해석1

Call :

```
lm( formula = expenses ~ ., data=insurance )
```

Residuals : 잔차? 표본에서 나온 관측값의 회귀선과 비교해 볼 때 나타나는 차이

Residuals:

Min	1Q	Median	3Q	Max
-11302.7	-2850.9	-979.6	1383.9	29981.7

예측에서 오차에 대한 요약 통계를 제공합니다.

결과해석 : 1. 모델이 최소 하나의 관측에 대해 거의 30000 달러의 비용을 낮게 예측했다.
2. 오차의 50%는 -2850달러 ~1383 달러 사이에 있다.

오차를 최소화 할 수 있는 적절한 회귀 직선을 알아내야 한다.

▣ 결과 해석2 (두번째 부분 해설)

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-11941.6	987.8	-12.089	< 2e-16 ***
age	256.8	11.9	21.586	< 2e-16 ***
sexmale	-131.3	332.9	-0.395	0.693255
bmi	339.3	28.6	11.864	< 2e-16 ***
children	475.7	137.8	3.452	0.000574 ***
smokeryes	23847.5	413.1	57.723	< 2e-16 ***
regionnorthwest	-352.8	476.3	-0.741	0.458976
regionsoutheast	-1035.6	478.7	-2.163	0.030685 *
regionsouthwest	-959.3	477.9	-2.007	0.044921 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

설명 : 추정된 회귀계수별로 표시된 p값은 추정된 계수가 실제 0일 확률 추정치이다.

p값이 작은 경우 실제 계수가 0이 아닐 가능성이 높다는 것을 말하며 특징이 종속변수와 관계가 없을 가능성이 아주 낮다는 것을 의미한다.

유의수준 보다 낮은 p값은 통계적으로 유의한 것으로 간주된다.

p 값이 0.05 미만으로 나온 독립변수가 유의한 변수들이다.

* p-value 값 ? 귀무가설에서 얻는 검정통계량의 값 이상으로 대립가설에서 유리한 데이터를 얻을 수 있는 데이터를 얻을 수 있는 확률

p값 > 유의수준 : 귀무가설을 기각할 수 없다.

p값 < 유의수준 : 대립가설 채택

옆에 나온 (*)이 추정치로 충족되는 유의수준을 나타내는 각주에 해당된다.

▣ 결과 해석3 (세번째 부분 해설)

Residual standard error: 6062 on 1329 degrees of freedom

Multiple R-squared: 0.7509, Adjusted R-squared: 0.7494

F-statistic: 500.9 on 8 and 1329 DF, p-value: < 2.2e-16

결과해석 : 결정계수(R-squared)란 회귀모형의 데이터에 대한 설명력을 나타내는 척도이다.
(수제비 책 4-4)

좋은 회귀모형에는 두가지 조건이 있다.

1. 데이터를 잘 설명한다.
2. 간단하다.

독립변수가 많은 회귀모형의 경우에는 위의 첫번째 조건을 만족한다. 그러나 두번째 조건은 탈락이라 볼 수 있다. 아무리 설명력이 좋아도 복잡하다면 그다지 좋은 모형은 아니다.

독립변수가 많으면 결정계수가 높아지는데 결정계수가 모형의 설명력을 측정하기 좋은 척도라는 것은 사실이지만 위의 단점을 보완하기 위해 보안된 척도가 있는데 그게 바로 " 조정된 결정계수(Adjusted R-squared) " 이다.

이 척도는 다행스럽게도 독립변수의 숫자가 증가한다고 해서 무작정 커지지 않는다.

Multiple R-squared: 0.7509, Adjusted R-squared: 0.7494

독립변수가 종속변수를 75% 설명하고 있다는 뜻이다.

위의 둘의 차이가 크면 불필요한 변수가 있을것이라고 예상할 수 있다.

■ 5단계. 모델 성능 개선

분류할 때 모델의 성능을 개선하려고 했던 일 ? 하이퍼 파라미터 조절

1. knn의 k값

2. 나이브베이즈 : laplace 값

3. decision tree: trials 값

회귀분석에서 성능개선을 위해 필요한 일 ? 파생변수 추가



종속변수(의료비)에 영향을 주는, 종속변수를 잘 설명할 수 있는 새로운 데이터



개인의 창의성이라 여기서부터 캐글의 순위가 결정됩니다.

6장 : 단순회귀분석 (lm) ---> 상관관계분석(cor) ---> 다중회귀분석(lm, reg 함수)

1. knn : 유클리드 거리
2. naivebayes : 나이브베이즈 공식
3. decision tree : 정보획득량 공식(분할 전 엔트로피 - 분할 후 엔트로피)
4. regression : 아래의 다중회귀 수학식

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

↴ ↓ →
 회귀계수 독립변수들의 행렬 종속변수
 (가운기, 절편)

* 좋은 파생변수를 생성해야 상위권 도전 가능

- R NCS 평가방법 : 1. 케글 순위 제출 2. 시각화와 분석이 들어있는 R 포트폴리오 제출

상관관계 분석 : 독립변수와 종속변수와의 상관관계를 살펴본다.

(예 : 나이가 들수록 의료비가 많이든다, 부양가족이 많을 수록 의료비가 많이든다.)

다중회귀분석 : 나이가 한살 더 먹을 수록 의료비가 정확하게 얼마나 드는지, 부양가족 한명이 늘어날수록 의료비가 정확하게 얼마나 드는지 확인

→ 독립변수와 독립변수들끼리의 상관관계가 높은지를 확인

■ 왜 독립변수들끼리의 상관관계를 확인해야하는가?

다중회귀분석을 하고 결과를 봤더니 유의한 변수들을 발견할 수 없었다고 한다면 다중공선성을 의심해봐야 한다.

설명 : 다중회귀분석을 했는데 결과에서 유의한 변수들이 보이지 않는다면 독립변수들끼리의 상관관계가 아주 높은지 의심을 해봐야한다.

만약 독립변수들끼리 상관관계가 매우 강하여 절대값 1에 가까워지면, 최소자승법 적용자체가 매우 심각한 국면을 맞이하게 된다.

이 때 나타나는 현상을 다중 공선성이라고 한다.

■ 다중 공선성 실험

1. 다중공선성의 VIF(팽창계수)를 확인할 수 있는 패키지 설치

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

설명 : 다중공선성을 보이는 변수들은 팽창계수가 10이상인 변수들이다.

2. 데이터를 로드한다.

```
test <- read.csv("test_vif1.csv")
test
```

3. 독립변수들끼리 상관관계를 확인한다.

```
cor( test[ , c("아이큐", "공부시간") ] )
```

	아이큐	공부시간
아이큐	1.0000000	0.7710712
공부시간	0.7710712	1.0000000

설명: 두 독립변수의 상관관계가 1에 가까운 강한 양의 상관관계를 보이고 있다.

4. 회귀분석 모델을 생성한다.

```
test <- test[ , -1] #학생번호는 제외한다.
m1 <- lm( test$시험점수 ~. , data=test )
summary(m1)
```

5. 다중 공선성을 보이는지 확인한다.

vif(m1)

아이큐 공부시간
2.466401 2.466401

현업기준 : 팽창계수(vif)가 **보통 10보다 큰것**으로 골라내고 **엄격하게 하려면 5보다 큰 것을** 골라낸다.
느슨하게 하려면 15~20으로 주로 골라낸다.

vif(m1)>10
아이큐 공부시간
FALSE FALSE

설명 : 위의 예제는 팽창계수가 10보다 크지 않으므로 다중공성선을 보이는 변수는 없다.

■ 회귀 트리 p 289

1. 회귀트리란? 수치를 예측하는 트리(tree)

의사결정트리



회귀트리



특정 숫자를 예측하는데 다중 회귀분석을 하지 않고 왜 회귀트리를 사용하는가?

* 수치예측

예 :

집값 <--- 평수, 학군, ...

의료비 <--- 부양가족수, 성별, 흡연여부, 나이 ...

수치예측 작업을 할 때는 일반적으로 전통적인 회귀분석 방법을 가장 먼저 선택하지만 경우에 따라서는 수치 의사결정트리가 분명한 이점을 제공한다. 의사결정트리의 장점을 수치예측에 활용할 수 있다.

의사결정트리는 작업의 특징이 많거나 특징과 결과간의 **매우 복잡하고 비선형적인 관계를 가질 때 잘 맞는다.** 반면 회귀분석은 이럴 때 어려움이 있다.

회귀 분석의 경우에는 독립변수의 개수가 많으면 독립변수들간의 상관관계를 유심히 조사해야한다. 왜냐하면 다중공선성 문제가 생기면 회귀분석결과에 신뢰도가 떨어지기 때문이다.

회귀트리 : 회귀 + 의사결정트리의 장점

회귀트리에서 사용하는 수학식? SDR(표준편차축소)을 사용

관련 ppt :



Machine
Learning...

정리 : x값을 고려하지 않고 y값만 가지고 데이터를 분할하는데 표준편차축소 값이 가장 높은 값을 기준으로 y값을 분할하고 분할된 각각의 영역의 평균값을 예측값으로 지정한다.
분할한 y값의 기준이 되는 x값을 가지고 ppt에 나온 것처럼 $x \leq 7$ 라는 조건을 가지고 x값이 7이하면 예상되는 y값은 1.99이고 x 값이 7보다 크면 예상되는 y값은 6.45라고 예측하는 것이 회귀트리이다.

■ 와인 데이터의 등급(수치)을 예측하는 회귀트리 모델을 생성하는 실습

데이터 : winetree.csv

#1. 데이터를 로드한다.

```
#첨부파일 whitewines.csv  
wine <- read.csv("whitewines.csv")
```

```
#fixed.acidity      : 고정 산도  
#volatile.acidity   : 휘발성 산도  
#citric.acid       : 시트르산  
#residual.sugar    : 잔류 설탕  
#chlorides          : 염화물  
#free.sulfur.dioxide : 자유 이산화황  
#total.sulfur.dioxide: 총 이산화황  
#density            : 밀도  
#pH                 : pH  
#sulphates          : 황산염  
#alcohol             : 알코올  
#quality             : 품질 <----- 얘만 종속변수
```

2. 와인데이터의 종속변수인 quality 데이터가 정규분포에 속하는 안정적인 데이터인지 확인

```
hist(wine$quality)
```



×

설명 : 어느 한쪽으로 데이터가 치우치지 않은 안정적인 모양을 보이고 있다.

3. wine 데이터를 train 데이터와 test 데이터로 나눈다.

```
wine_train <- wine[1:3750, ] #3750, 77%
wine_test <- wine[3751:4898, ] #1148, 23%
```

4. train 데이터를 가지고 model 을 생성한다.

설명 : 회귀트리 모델을 구현할 수 있는 rpart 패키지를 설치한다.

```
install.packages("rpart")
library(rpart)
model <- rpart( quality ~ . , data=wine_train)
model

node), split, n, deviance, yval
 * denotes terminal node

1) root 3750 2945.53200 5.870933
  2) alcohol< 10.85 2372 1418.86100 5.604975
    4) volatile.acidity>=0.2275 1611  821.30730 5.432030
      8) volatile.acidity>=0.3025 688  278.97670 5.255814 *
      9) volatile.acidity< 0.3025 923  505.04230 5.563380 *
```

```
5) volatile.acidity< 0.2275 761 447.36400 5.971091 *
3) alcohol>=10.85 1378 1070.08200 6.328737
6) free.sulfur.dioxide< 10.5 84 95.55952 5.369048 *
7) free.sulfur.dioxide>=10.5 1294 892.13600 6.391036
14) alcohol< 11.76667 629 430.11130 6.173291
28) volatile.acidity>=0.465 11 10.72727 4.545455 *
29) volatile.acidity< 0.465 618 389.71680 6.202265 *
15) alcohol>=11.76667 665 403.99400 6.596992 *
```

설명 : * 표시가 있는 노드는 앞노드로 노드에서 예측이 이루어진다는 것을 의미한다.
와인데이터의 예측등급이다.

5.97이라는 등급으로 예를들면 alcohol < 10.85이고

volatile.acidity < 0.2275 이면 모든 와인 샘플의 품질값은 5.97로 예상된다. 등급이 3~9사이로 구성되어 있다.

5. 위에서 나온 모델로 트리를 시각화 하시오 !

```
install.packages("rpart.plot")
library(rpart.plot)
rpart.plot( model, digits=3)
```



설명 : digits= 3은 소수점 세번째까지 허용하겠다는 뜻

```
rpart.plot(model, digits=3, fallen.leaves=T, type=3, extra=101)
```



6. 위에서 만든 모델로 테스트 데이터의 라벨을 예측하시오 !

(위에서 따로 분리했던 23%의 데이터로 테스트 한다.)

```
result <- predict(model, wine_test)
```

7. 테스트 데이터의 실제 라벨(품질) 과 예측결과(품질) 을 비교한다

```
cbind( round(result), wine_test$quality)
```

8. 테스트 데이터의 라벨과 예측 결과와 상관관계가 어떻게 되는지 확인한다.

```
cor(result, wine_test$quality) # 0.5369525
```

설명 : 이전에 knn, naivebayes, decision tree의 경우에는 이원 교차표를 그려서 정확도를 확인했는데 이번에는 수치 예측으로 상관관계와 오차율을 살펴보며 모델의 성능을 확인해야 한다.

9. 두 데이터간의 오차율을 확인

```
MAE <- function( actual, predicted) {  
  mean( abs( actual - predicted) )  
}
```

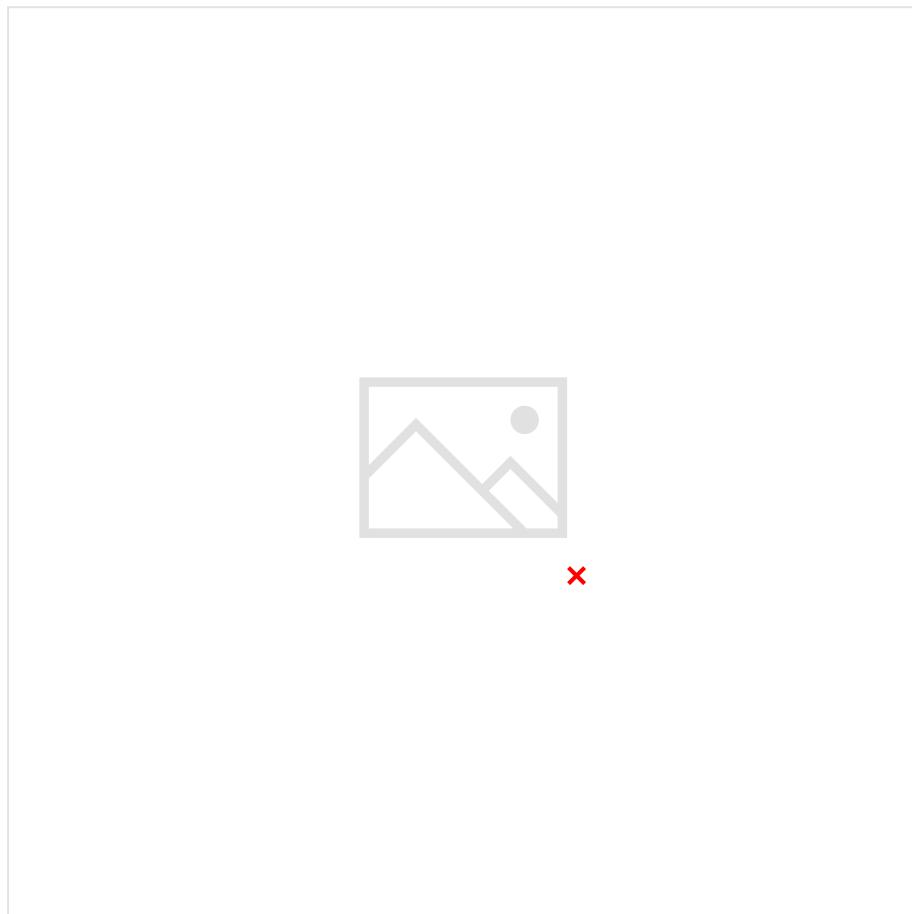
```
MAE( result, wine_test$quality) # 0.5872652
```

위의 경우에는 상관관계는 1에 가까울수록 좋고 오차는 0에 가까울수록 좋다.

설명 : 이 모델의 경우 다른 모델인 서포트 벡터머신에서는 오차가 0.45인데 0.58 이면 상대적으로 좀 큰 오차이므로 개선의 여지가 필요해 보인다.

개선방법 : 회귀트리 -----> 모델트리

■ 모델트리



기존 회귀트리 모델 + 다중회귀를 추가한 모델

회귀트리는 무조건 분할한 y (종속변수의 값들) 값들의 평균값으로만 예측을 했는데 모델트리는 분할한 x 값과 y 값들에 대한 회귀식을 통해서 y 값을 예측을 한다.

#1. 회귀트리 실습의 1번 부터 3번 실습 까지 반복한다.

#1.1. 데이터를 로드한다.

첨부파일 whitewines.csv

```
wine <- read.csv("whitewines.csv")
```

#fixed.acidity : 고정 산도

#volatile.acidity : 휘발성 산도

#citric.acid : 시트르산

```
#residual.sugar      : 잔류 설탕  
#chlorides         : 염화물  
#free.sulfur.dioxide : 자유 이산화황  
#total.sulfur.dioxide: 총 이산화황  
#density           : 밀도  
#pH                : pH  
#sulphates        : 황산염  
#alcohol          : 알코올  
#quality          : 품질
```

1.2. 와인의 quality 데이터가 정규분포에 속하는 안정적인 데이터인지 확인

```
hist(wine$quality)
```

1.3. wine 데이터를 train 데이터와 test 데이터로 나눈다.

```
wine_train <- wine[1:3750, ]  
wine_test  <- wine[3751:4898, ]
```

2. 모델트리를 구현하기 위한 패키지 설치

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

3. 와인의 품질을 예측하는 모델을 생성한다.

```
m.cubist <- cubist(x= wine_train[-12], y=wine_train$quality)  
                                ↑  
                                ↑  
                    독립변수들          종속변수  
m.cubist
```

4. 만든 모델과 테스트 데이터로 예측을 한다.

```
p.cubist <- predict( m.cubist, wine_test)  
p.cubist
```

5. 예측값(p.m5p) 과 테스트 데이터의 라벨간의 상관관계를 확인한다

```
cor( p.cubist , wine_test$quality ) #0.6201015
```

6. 예측값(p.m5p) 과 테스트 데이터의 라벨간의 평균절대오차를 확인 한다.

```
MAE( wine_test$quality, p.cubist) #0.5339725
```

설명 : 회귀트리일때는 오차가 0.58이었는데 모델트리는 오차가 0.53으로 좀 더 개선이 됨

6장 정리 : 다중 회귀분석에서는 미국 국민 의료비 데이터로 미국민의 의료비를 예측하는 회귀모델을 생성했다. 회귀모델의 성능을 높이기 위해서 파생변수를 생성했었다. 회귀트리와 모델트리는 분류를 하는데 회귀의 개념이 섞인 분류이다.

머신러닝의 종류 3가지?

1. 지도학습 : 분류 - knn, naivebayes, decision tree, oneR, riper, regression tree, model tree
회귀 - 단순회귀분석, 다중회귀분석
2. 비지도학습
3. 강화학습

■ 오라클에서 바로 SQL로 회귀분석 구현하는 방법

1. 회사의 중요한 데이터는 다 오라클과 같은 RDBMS에 들어있다.
2. 오라클과 파이썬 또는 R과 연동해서 회귀분석을 하는 경우가 많은데 연동하지 않고 바로 오라클의 회귀분 패키지를 이용해서 분석 하게되면 오라클의 자동 SQL튜닝을 이용할 수 있다는 장점이 있다. 오라클은 대용량 데이터를 빠르게 처리하는 기능이 이미 내장되어 있어서 그 기능을 이용할 수 있다. 그런데 파이썬이나 R과 연동하면 대용량 데이터인 경우 모래시계가 뜨면서 분석시간이 상당히 오래 걸리거나 메모리 부족 오류가 나면서 분석을 못 할 수도 있다.
3. 케글 상위권에 도전할 때 오라클의 머신러닝 패키지를 이용하면 좀 더 편하게 상위권에 들어갈 수 있다.
하이퍼 파라미터를 사용자가 직접 지정하지 않아도 최적의 하이퍼 파라미터를 오라클이 알아서 찾아낸다.
(자동으로 수행하는 기능을 사용할 때)

■ 보스톤의 집값 데이터 예측을 오라클 SQL을 이용해서 회귀분석 하기

1. 보스톤 하우징 데이터를 테이블로 구성한다.

```
drop table BOSTON_HOUSING;
```

```
CREATE TABLE BOSTON_HOUSING
```

```
(  
    ID      NUMBER,  
    CRIM    NUMBER,  
    ZN      NUMBER,  
    INDUS   NUMBER,  
    CHAS    NUMBER,  
    NOX     NUMBER,  
    RM      NUMBER,  
    AGE     NUMBER,  
    DIS     NUMBER,  
    RAD     NUMBER,  
    TAX     NUMBER,  
    PTRATIO NUMBER,  
    BLACK   NUMBER,  
    LSTAT   NUMBER,  
    MEDV   NUMBER  
);
```

테이블이 생성되었으면 훈련데이터(train.csv)를 이 테이블에 입력한다.
생성 되었으면 sqldeveloper를 이용해 수행한다.

boston-housing 압축파일을 풀업하면 3가지 파일이 있다. (캐글 데이터)

1. train.csv : 정답이 있는 훈련 데이터
2. test.csv : 정답이 없는 테스트 데이터
3. submission_example.csv : 예측한 정답을 만들어낸 제출용 샘플 파일

위의 예측한 submission_example.csv를 캐글에 제출하면 바로 전세계에서 내가 몇등인지 등수를 알려준다.

2. 머신러닝 구현

SETTINGS_GLM이라는 이름으로 머신러닝 환경구성 테이블을 생성한다.
회귀분석을 할 거라고 이 환경 구성 테이블에 알려준다.

```
DROP TABLE SETTINGS_GLM;
```

```
CREATE TABLE SETTINGS_GLM  
AS  
SELECT *  
FROM TABLE (DBMS_DATA_MINING.GET_DEFAULT_SETTINGS)
```

```

WHERE SETTING_NAME LIKE '%GLM%';

BEGIN

INSERT INTO SETTINGS_GLM
VALUES (DBMS_DATA_MINING.ALGO_NAME, 'ALGO_GENERALIZED_LINEAR_MODEL');
-- 회귀분석 코드

INSERT INTO SETTINGS_GLM
VALUES (DBMS_DATA_MINING.PREP_AUTO, 'ON');

INSERT INTO SETTINGS_GLM
VALUES (
    DBMS_DATA_MINING.GLMS_RIDGE_REGRESSION,
    'GLMS_RIDGE_REG_DISABLE');

INSERT INTO SETTINGS_GLM
VALUES (
    DBMS_DATA_MINING.ODMS_MISSING_VALUE_TREATMENT,
    'ODMS_MISSING_VALUE_MEAN_MODE');

--결측치를 그 데이터의 평균값으로 치환하라는 코드

COMMIT;

END;
/

```

3. 모델 생성

```

CREATE OR REPLACE VIEW VW_BOSTON_HOUSING
AS SELECT * FROM BOSTON_HOUSING;

```

```

BEGIN

DBMS_DATA_MINING.CREATE_MODEL(
    model_name      => 'MD_GLM_MODEL', --모델명
    mining_function => DBMS_DATA_MINING.REGRESSION, --회귀분석하겠다.

```

```
data_table_name      => 'VW_BOSTON_HOUSING', --회귀분석 할 테이블명
case_id_column_name => 'ID', --보스톤 하우징의 집의 id 번호
target_column_name   => 'MEDV', --라벨컬럼
settings_table_name  => 'SETTINGS_GLM'); --환경셋팅 테이블 이름

END;

/
```

4. 모델 확인

모델 확인하는 SQL은 SQLdeveloper에서 보기

```
SELECT MODEL_NAME,
       ALGORITHM,
       COMMENTS,
       CREATION_DATE,
       MINING_FUNCTION,
       MODEL_SIZE
  FROM ALL_MINING_MODELS
 WHERE MODEL_NAME = 'MD_GLM_MODEL';
```

-- 만든 머신러닝 환경정보 확인

```
SELECT SETTING_NAME, SETTING_VALUE
  FROM ALL_MINING_MODEL_SETTINGS
 WHERE MODEL_NAME = 'MD_GLM_MODEL';
```

--R에서 회귀분석 결과를 본 내용이 아래의 SQL이다.

```
SELECT * FROM
TABLE (DBMS_DATA_MINING.GET_MODEL_DETAILS_GLM ('MD_GLM_MODEL'));
```

5. 테스트 데이터를 담을 테이블을 생성하고 test.csv를 로드한다.

```
CREATE TABLE BOSTON_HOUSING_TEST
```

```
(  
    ID      NUMBER,  
    CRIM   NUMBER,  
    ZN     NUMBER,  
    INDUS  NUMBER,  
    CHAS   NUMBER,  
    NOX    NUMBER,  
    RM     NUMBER,  
    AGE    NUMBER,  
    DIS    NUMBER,  
    RAD    NUMBER,  
    TAX    NUMBER,  
    PTRATIO NUMBER,  
    BLACK  NUMBER,  
    LSTAT  NUMBER,  
    MEDV   NUMBER  
);
```

```
SELECT * FROM BOSTON_HOUSING_TEST;
```

6. 테스트 데이터의 집값을 위에서 만든 회귀 모델로 예측한다.

```
SELECT id,  
       PREDICTION (MD_GLM_MODEL USING *) MODEL_PREDICT_RESPONSE  
  FROM BOSTON_HOUSING_TEST T;
```

위의 id와 예측한 집값 173개를 kaggle에 올려서 자신의 순위를 확인한다.

■ 미국 보스톤 집값 예측을 회귀분석으로 분석해서 결과물을 캐글에 올리고 자신의 점수가 어떻게 되는지 확인하는 방법

1. 훈련 데이터와 테스트를 나누고 훈련 데이터로 회귀모델을 생성
(이미 캐글에서 훈련데이터와 테스트 데이터를 나눠서 제공)
2. 정답(집값)이 있는 훈련 데이터로 회귀모델 생성
3. 만든 모델로 정답이 없는 테스트 데이터의 집값을 예측
4. 테스트 데이터의 정답과 내가 만든 예측값과의 오차가 0에 가까울수록 잘 만든 회귀모델

정답은 캐글만이 알고 있기 때문에 오차를 확인하려면 캐글에 결과물을 올려서 확인을 해야 함

■ 7장. 신경망

머신러닝의 종류 3가지 ?

1. 지도학습 : 분류 - knn, naivebayes, decision tree, oneR, riper, regression tree, model tree,
신경망
회귀 - 단순회귀와 다중회귀, 신경망
2. 비지도 학습
3. 강화학습

* 인공신경망을 만들어서 분류작업을 수행

인공신경망을 만들면서부터 육체노동을 대신했던 기계(컴퓨터)에게 지능을 부여하기 시작

↓

인공지능 신경망의 시초는 퍼셉트론이다.

퍼셉트론은 1958년 로센블래트라는 분에 의해 제안된 알고리즘

사람의 뇌 ----> 컴퓨터를 이용한 지능처리

2014년 딥마인드라는 회사를 구글이 4000억원을 주고 인수하면서 인공신경망이 더 많이 알려졌고
딥마인드가 2015년 2월에 딥러닝 2.0버전을 소개하면서 DQN(깊은 보상학습)을 알렸다
---> 이게 바로 알파고

■ 단층 퍼셉트론과 이후 딥러닝의 역사

17세기의 라이프니츠 수학자 ---> 19세기 조지 볼 ---> 엘런튜링 수학자
(이진법) (0과 1로 논리연산자) (논리연산자 / 컴퓨터로 구현)

---> 로젠틀라트 1957 퍼셉트론(단층) ---20년 후 ---> 오차역전파(다층) ---> 딥러닝
1. and게이트 xor게이트
2. or 게이트
3. not and 게이트

■ 컴퓨터로 퍼셉트론 구현하기 (R코드)

문제251~260번

■ 활성화 함수의 종류 p318

1. 계단함수 : 입력신호의 총합이 임계치를 넘느냐 안넘느냐를 숫자 0과 1로 리턴하는 함수
예 : $f(0.3) = 1, f(-0.2)=0$

2. 시그모이드 함수 : 계단함수는 무조건 0아니면 1을 리턴하지만 시그모이드는 0~1사이의 연속적인 실수값을 리턴한다.

단층 : 입력층 -----> 출력층

다층 : 입력층 -----> 은닉층 -----> 출력층

단층이 아니라 다층 신경망을 사용하려면 활성화 함수를 시그모이드(sigmoid) 함수를 사용해야 한다.

1

시그모이드 함수 공식 : $f(k) = \frac{1}{1 + \exp(-k)}$ (비선형함수)

$$f(1.0) = 0.731$$

$$f(2.0) = 0.880$$

3. 렐루 함수 : relu 함수

↓

Rectified Linear unit

정류된 -----> 전기회로용어

시그모이드 함수의 단점이 전파가 역전파 될때 기울기 소실로 전파가 앞층까지 안된다는 단점이 있어서 나오게 된 함수이다.

예 : 구글 검색으로 gradient vanishing 해보기

$$f(x) = \begin{cases} x & (x > 0) \\ 0 & (x \leq 0) \end{cases}$$

입력값이 0보다 크면 그 값을 그대로 출력하고 0보다 작거나 같으면 0으로 출력하는 함수



1. 계단함수

2. 시그모이드

3. 렐루함수 : 현업에서 많이 사용

4. leaky relu 함수 : 렐루함수의 음수 부분의 기울기가 0이어서 역전파할때 기울기가 소실되므로 기울기를 0이 아니게 만들어 준 함수

5. tanh 함수

인공신경망의 기초가 된 알고리즘은 인공신경세포 하나를 컴퓨터로 구현한 퍼셉트론이다.

퍼셉트론에서 사용되는 활성화함수는 위와 같이 5가지 함수이다.

인공신경망 학습의 목표는 학습이 잘 된 가중치를 생성하는 것이다.

■ 신경망 실습1 (콘크리트 강도를 예측하는 인공신경망 만들기)

" 어떻게 재료를 조합해야 강도가 높은 콘크리트를 만들 수 있는가? "

자갈, 모래, 시멘트등을 몇대 몇 비율로 섞었을때 어느정도 강도가 나오는지 예측하는 신경망
신경망으로 분류도 할 수 있고 회귀분석도 할 수 있다.

1. 콘크리트 데이터 소개

* 콘크리트 데이터

1. mount of cement : 콘크리트의 총량
2. slag : 시멘트
3. ash : 분 (시멘트)
4. water : 물
5. superplasticizer : 고성능 감수재(콘크리트 강도를 높이는 첨가제)
6. coarse aggregate : 굵은 자갈
7. fine aggregate : 잔 자갈
8. aging time : 숙성시간

2. 콘크리트 데이터를 R로 로드한다.

- 머신러닝 데이터 116번

```
concrete <- read.csv("concrete.csv")  
str(concrete)
```

3. 정규화 함수로 데이터를 정규화 작업

```
normalize <- function(x) {  
  return ( (x-min(x)) / (max(x) - min(x)) )  
}
```

```
concrete_norm <- as.data.frame(lapply(concrete,normalize) )
```

설명 : strength가 종속변수이고 연속형 데이터로 구성되어 있습니다.

4. 0~1사이로 데이터가 잘 변경되었는지 확인

```
summary( concrete_norm$strength)
```

본래 데이터의 최소값, 최대값과 비교

```
summary( concrete$strength)
```

설명 : 콘크리트 강도는 82가 가장 크고 2가 가장 작습니다.

※ 현업에 리얼 데이터 또는 캐글의 데이터로 분석할 때 추가작업

1. 결측치 확인 :

```
colSums( is.na(concrete) )
```

2. 이상치 확인 :

```
library(outliers)
```

```
grubbs.flag <- function(x) {  
  outliers <- NULL  
  test <- x  
  grubbs.result <- grubbs.test(test)  
  pv <- grubbs.result$p.value  
  while(pv < 0.05) {  
    outliers <- c(outliers,as.numeric(strsplit(grubbs.result$alternative," ")[[1]][3]))  
    test <- x[!x %in% outliers]  
    grubbs.result <- grubbs.test(test)  
    pv <- grubbs.result$p.value  
  }  
  return(data.frame(X=x,Outlier=(x %in% outliers)))  
}
```

```
wisc <- read.csv("concrete.csv")
```

```
for (i in 3:length(colnames(wisc))){  
  
  a = grubbs.flag(wisc[,colnames(wisc)[i]])  
  b = a[a$Outlier==TRUE,"Outlier"]  
  print ( paste( colnames(wisc)[i] , '-->' , length(b) ) )  
  
}
```

결과 :

```
[1] "ash --> 0"  
[1] "water --> 0"  
[1] "superplastic --> 5"  
[1] "coarseagg --> 0"  
[1] "fineagg --> 0"  
[1] "age --> 59"  
[1] "strength --> 0"
```

5. 훈련 데이터, 테스트 데이터를 나눈다 (8:2)

```
concrete_train <- concrete_norm[1:773, ] #773  
concrete_test <- concrete_norm[774:1030, ] #257
```

6. neuralnet 패키지를 설치한다.

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

7. neuralnet 패키지에 콘크리트 훈련 데이터를 넣어서 모델을 생성한다.

```
concrete_model <- neuralnet(formula=strength ~ cement + slag + ash + water + superplastic +  
coarseagg + fineagg + age, data =concrete_train)
```

9. 모델(신경망) 을 시각화

```
plot(concrete_model )
```



✗

10. 만든 모델로 테스트 데이터를 가지고 테스트 한다.

```
model_results <- compute(concrete_model, concrete_test[1:8])
```

```
predicted_strength <- model_results$net.result
```

11. 예측값과 실제값간의 상관관계를 확인

```
cor(predicted_strength, concrete_test$strength) #0.8062537
```

설명 : 예측결과가 콘크리트 강도이기 때문에 이원교차표로 정확도를 확인해서 신경망 모델의 성능을 확인할 수는 없고 상관관계를 구해서 성능을 확인해야한다.

12. 모델의 성능 개선

```
concrete_model2 <- neuralnet(formula=strength ~ cement + slag + ash +  
water +superplastic + coarseagg + fineagg + age, data =concrete_train, hidden=c(5,2) )
```

설명 : hidden= c(5, 2)

↑ ↑

은닉1층의 뉴런 수 은닉2층의 뉴런 수

원래는 2층 신경망이었는데 3층으로 늘리면서 뉴런도 7개로 만들었다.

뉴런의 개수

1. 사람 : 850억개
2. 고양이 : 10억개
3. 쥐 : 7천 5백만개
4. 바퀴벌레 : 몇백만개
5. 하루살이 : 지금까지 나온 인공지능의 뉴런 수 보다 많다.

```
plot(concrete_model2)
```



입력층 은닉1층 은닉2층 활성층

13. 위에서 만든 모델로 테스트를 수행한다.

```
model_results <- compute(concrete_model2, concrete_test[1:8])  
predicted_strength2 <- model_results$net.result
```

14. 예측값과 실제값간의 상관관계를 확인

```
cor(predicted_strength2, concrete_test$strength) #0.9333894
```

■ 정규화된 데이터를 역정규화 하는 실습

카페 게시글 653 denormalize data

#1. 테스트 데이터를 만든다.

```
dd <- data.frame(x=1:5,y=6:10)
dd
```

2. 위의 테스트 데이터를 min/max 정규화 한다.

```
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}
```

```
ddnorm <- as.data.frame(lapply(dd,normalize))
ddnorm
```

3. 정규화 된 데이터를 다시 역정규화 한다.

3.1 정규화 하기전 데이터에서 최소 벡터와 최대 벡터를 찾는다.

```
minvec <- sapply(dd,min)
maxvec <- sapply(dd,max)
minvec
```

```
denormalize <- function(x,minval,maxval) {
  x*(maxval-minval) + minval
}
```

```
as.data.frame(Map(denormalize,ddnorm,minvec,maxvec))
# 설명 : Map( 역정규화 시키는 함수명, 정규화된 데이터 프레임명, 정규화 시키기 전 최소벡터, 정규화 시키기 전 최대 벡터 )
```

■ 역정규화 방법을 변경한 코드

결론 : 방금전에 설명한 코드와의 차이는?

-> 캐글에 id와 테스트 데이터에 대한 예측집값을 올릴 때 정규화된 테스트 데이터의 예측 집값을 역 정규화 시킬 때 훈련 데이터의 집값 중 최댓값인 45와 최솟값인 5를 이용해서 집값만 역정규화 시킴

1. 데이터를 불러온다.

```
getwd()

normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x) ) )
}

boston_train<- read.csv('train.csv', head=T)
boston_test<-read.csv("test.csv", head=T)
str(boston_test)

head(boston_train)
head(boston_test)
```

2. 정규화를 시킨다.

```
boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

summary(boston_train_norm)
summary(boston_test_norm)
```

3. 신경망 모델을 생성한다.

```
#install.packages("neuralnet")
library(neuralnet)

boston_model <- neuralnet(formula=medv ~ crim + zn + indus + chas + nox + rm + age + dis +
rad + tax + ptratio + black + lstat ,
                           data=boston_train_norm ,hidden = c(2,5) , learningrate =0.02)
```

#4. 테스트 데이터에 대한 예측값을 출력한다.

```
model_results <- compute(boston_model, boston_test_norm)
```

```
pred_medv <- model_results$net.result
pred_medv
```

설명 : 정규화된 상태로 집값이 예측된다.

#5. 역정규화 시키는데 테스트 데이터의 집값만 역정규화 시킨다.

훈련 데이터의 집값 중 최댓값이 45이고 최솟값이 5이므로 아래와 같이 역정규화 시킨다.

```
denormalize <- function(x) { return ( x*(50-5+5) } #max(train_data$medv) : 45
                                #min(train_data$medv) : 5
pred_medv_un <- denormalize(pred_medv)
```

#6. 캐글에 올릴 제출 포맷으로 생성한다.

```
sample <- cbind(boston_test$ID, pred_medv_un)
head(sample)
```

```
colnames(sample) <- c("id", "medv")
```

```
write.csv(sample, "Submission_sample.csv", row.names=FALSE)
```

- 위에서 만든 모델의 가중치를 확인하는 방법 :
summary(boston_model)

■ 파생변수를 추가해 보스톤 하우징 머신러닝 모델의 성능을 높이는 방법

1. 집값에 영향을 주는 독립변수가 무엇인지 확인 (상관관계로 확인)

1.1 책 p. 275에서 사용했던 pairs.panel() 함수 이용

```
library( psych )
pairs.panels( boston_train[ , -1 ] )
```

1.2 corrplot 패키지를 이용해서 상관관계 확인

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



```
boston_cor <- cor(boston_train[, -1])
boston_cor
```

```
corrplot(boston_cor, method="circle")
corrplot(boston_cor, method="number")
```



✗

설명 : 남색과 빨간색으로 시각화하는데 남색에 가까울수록 양의 상관관계가 높고 빨간색으로 진해지면 음의 상관관계가 높다. 원의 크기가 클수록 상관관계가 크다.



✗

결과 분석 :

- 결과1. 방의 개수(rm)이 많을수록 집값(medv)이 올라간다.
- 결과2. 하위계층의 비율(lstat)가 높을수록 집값이 떨어진다.
- 결과3. 오래된 집(age)일수록 직업센터의 접근성지수(dis)가 떨어진다.
- 결과4. 공장지대(indus)일 수록 일산화질소(nox)가 높다.

2. 히스토그램 그래프를 그려서 데이터 분포 확인

2.1 종속변수가 정규성을 띠는지 확인한다.

```
hist( boston_train$medv )
```



✗

2.2 방의 개수(rm)가 정규성을 띠는지 확인한다.

```
hist( boston_train$rm )
```



✗

2.3 하위계층의 비율(lstat)이 정규성을 띠는지 확인한다.

```
hist( boston_train$lstat )
```



2.4 결측치가 있는지 확인한다.

```
colSums( is.na( boston_train ) )
colSums( is.na( boston_test ) )
```

2.5 이상치가 있는지 확인한다.

```
str(boston_train)
library(outliers)
```

```
grubbs.flag <- function(x) {
  outliers <- NULL
  test <- x
  grubbs.result <- grubbs.test(test)
  pv <- grubbs.result$p.value
  while(pv < 0.05) {
    outliers <- c(outliers,as.numeric(strsplit(grubbs.result$alternative," ")[[1]][3]))
    test <- x[!x %in% outliers]
    grubbs.result <- grubbs.test(test)
    pv <- grubbs.result$p.value
  }
  return(data.frame(X=x,Outlier=(x %in% outliers)))
}
```

```
wisc <- read.csv("train.csv")
```

```
for (i in c(1,2,3,4,6,7,8,9,10,11,12,13,14,15)){
```

```
  a = grubbs.flag(wisc[,colnames(wisc)[i]])
  b = a$a$Outlier==TRUE,"Outlier"]
```

```
print ( paste( colnames(wisc)[i] , '-->' , length(b) ) )  
}
```

결과:

```
[1] "ID --> 0"  
[1] "crim --> 26"  
[1] "zn --> 34"  
[1] "indus --> 0"  
[1] "nox --> 0"  
[1] "rm --> 1"  
[1] "age --> 0"  
[1] "dis --> 0"  
[1] "rad --> 0"  
[1] "tax --> 0"  
[1] "ptratio --> 0"  
[1] "black --> 64"  
[1] "lstat --> 1"  
[1] "medv --> 0"
```

설명 : 결측치가 있다면 다른 값으로 치환해야 한다. 그 다른값은 다음과 같다.

- a. 결측치가 있는 컬럼의 다른 데이터의 평균값
- b. 결측치가 있는 컬럼의 다른 데이터의 최빈값
- c. 결측치가 있는 컬럼의 다른 데이터의 회귀식의 y 값

이상치가 너무 터무니없이 큰 이상치 값이라면 그 값을 포함한 행을 다 지운다.

3. 파생변수를 추가한 데이터를 학습시켜서 모델 재생성

결과1. 방의 개수(rm)이 많을수록 집값(medv)이 올라간다.

결과2. 하위계층의 비율(lstat)이 높을수록 집값이 떨어진다.

결과3. 오래된 집(age)일수록 직업센터의 접근성지수(dis)가 떨어진다.

결과4. 공장지대(indus)일 수록 일산화질소(nox)가 높다.

"비만이면서 담배를 피면 의료비가 많이든다" 라는것을 보고 비만이면서 담배를 피면 1 아니면 0으로 더미변수를 생성해주면 머신러닝 모델이 더 잘 학습할 수 있었던 것처럼 위의 결과를 보고 관련된 더미변수를 생성해줘야합니다.

어떠한 조건하에 ~이면 집값이 더 비싸다?

건물이 최근에 지어졌으면서 (노후화 x) 직업센터와의 접근성 지수가 높으면 집값이 더 비싸다.

예제1. age와 medv와의 plot 그래프를 확인하시오!

```
attach( boston_train )
```

```
plot( age, medv, col="blue" )
cor(age, medv) # -0.3588883
```

설명 : 오래된 집일수록 집값이 떨어지는 것을 확인할 수 있습니다.

예제2. 강남에 인접할 때 집값이 더 높은지 plot그래프로 확인해보시오!

```
plot( dis, medv )
cor( dis, medv ) # 0.2494223
```

설명 : 상업지구에 대한 접근지수가 높을 수록 집값이 높다.

최근에 지은 집이면서 접근 지수가 높은 조건은 ?

```
range(age) # 6 ~ 100
hist( age )
```

상업지구의 접근지수가 높은 조건은 ?

```
range( dis ) #1 ~ 10
hist( dis )
```

예제3. 훈련 데이터에서 아래의 조건에 해당되는 건수가 어떻게 되는지 확인하시오!

```
boston_train[ ( age < 20 & dis >= 8 ), ]
boston_train[ ( age < 30 & dis >= 5 ), "medv" ]
```

예제4. age<30이면서 dis >=5인 데이터는 1로 하고 아니면 0으로 하는 파생변수를 age_dis라는 컬럼으로 만드시오!

```
boston_train$age_dis <- ifelse( ( age < 30 & dis >= 5 ) 1, 0 )
table( boston_train$age_dis )
```

```
#####
getwd()

normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x)) )
```

```
}  
  
boston_train<- read.csv('train.csv', head=T)
boston_test<-read.csv("test.csv", head=T)
```

```
boston_train$age_dis <- ifelse( ( boston_train$age < 30 & boston_train$indus <= 10) , 1, 0 )
boston_test$age_dis <- ifelse( ( boston_test$age < 30 & boston_test$indus <= 10) , 1, 0 )
```

```
head(boston_train)
head(boston_test)
```

```
boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize))
```

```
summary(boston_train_norm)
summary(boston_test_norm)
```

###강남접근 파생변수 추가 최종 코드

```
#install.packages("neuralnet")
library(neuralnet)

set.seed(123)
boston_model <- neuralnet(formula=medv ~ crim + zn + indus + chas + nox + rm + age + dis +
rad + tax + ptratio + black + lstat + age_dis ,
                           data=boston_train_norm ,hidden = c(6,3,2) , learningrate =0.02)

model_results <- compute(boston_model, boston_test_norm)

pred_medv <- model_results$net.result
pred_medv

denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data$medv)
pred_medv_un <- denormalize(pred_medv)

sample <- cbind(boston_test$ID, pred_medv_un)
head(sample)

colnames(sample) <- c("id", "medv")

write.csv(sample, "Submission_sample2.csv", row.names=FALSE)
```

5. 테스트 데이터를 예측하고 캐글에 결과물 올림

■ 다중 회귀 분석으로 보스톤 집값을 예측하는 모델을 만들어서 캐글 상위권에 도전하기

```
getwd()

normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x) ) )
}

boston_train<- read.csv('train.csv', head=T)
boston_test<-read.csv("test.csv", head=T)

#boston_train$age_dis <- ifelse( ( boston_train$age < 30 & boston_train$indus <= 10) , 1, 0 )
#boston_test$age_dis <- ifelse( ( boston_test$age < 30 & boston_test$indus <= 10) , 1, 0 )

head(boston_train)
```

```
head(boston_test)

boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

summary(boston_train_norm)
summary(boston_test_norm)

boston_reg_model <- lm(medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad + tax + ptratio + black +
lstat ,
data=boston_train_norm )

summary(boston_reg_model)

model_results <- predict(boston_reg_model, boston_test_norm)

model_results

pred_medv <- model_results
pred_medv

denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data$medv)
pred_medv_un <- denormalize(pred_medv)

sample <- cbind(boston_test$ID, pred_medv_un)
head(sample)

colnames(sample) <- c("id", "medv")

write.csv(sample, "Submission_sample16.csv", row.names=FALSE)
```

■ 8장. 연관규칙

머신러닝의 종류 3가지 ?

1. 지도학습 : 정답이 있다.

분류 : knn(3장), naivebayes(4장), decision tree (5장), 신경망(7장)

회귀 : 회귀분석(6장)

정답을 가지고 있는 데이터를 가지고 학습시켜서 분류와 예측(회귀)를 잘하는 모형을 생성

- a. 독버섯과 식용버섯을 분류하는 머신러닝 모형 (naivebayes)
- b. 유방암의 악성과 양성을 분류하는 머신러닝 모형 (knn)
- c. 콘크리트 강도를 예측하는 머신러닝 모형 (신경망)
- d. 미국민의 의료비를 예측하는 머신러닝 모형(회귀분석)
- e. 보스톤 지역의 집값예측(신경망, 회귀분석)

위의 5가지 모형은 전부 데이터에 정답이 있어서 정답을 가지고 컴퓨터를 학습 시켰다.

2. 비지도학습 : 정답이 없다.

apriori 알고리즘(연관규칙 / 8장) : 교보문고, 쿠팡, 아마존 등의 연관 상품 추천

k-means(9장) : 통신사에 기지국을 세울 때 어디에 기지국을 세우는게 가장 효과가 좋은지 찾을 때

3. 강화학습

■ 연관규칙이 필요한 이유 ?

연관규칙이란 데이터 내부에 존재하는 항목간의 상호관계 혹은 종속관계를 찾아내는 분석기법이다. 데이터 간의 관계에서 조건과 반응을 연결하는 분석으로 장바구니 분석 또는 서열 분석이라고 한다

(수제비 책 p. 3-55)

연관규칙을 활용한 쿠팡의 사례

SSG광고 (신세계 온라인 쇼핑)

■ Apriori 알고리즘

간단한 성능 측정치를 이용해서 거대한 데이터에서 데이터 간의 연관성을 찾는 알고리즘

■ Apriori 알고리즘은 어떤 데이터의 패턴을 찾을 때 유용한가?

1. 암 데이터에서 빈번히 발생하는 DNA 패턴과 단백질의 서열을 검사할 때
2. 사기성 신용카드 및 사기성 보험료 청구시에 패턴 발견
3. 유통업에서는 장바구니 분석을 통해 상품 추천 뿐만 아니라 상품진열, 홈쇼핑의 경우에는 방송순서 등에도 적용 가능

■ 연관규칙에 관련한 중요 용어 3가지 ?

1. 지지도 : 전체 거래 중 항목 A와 B를 동시에 포함하는 거래의 비율

1.1 X 아이템의 지지도 ?

$$n(X) \quad \leftarrow \text{아이템 } X\text{의 거래건수}$$

표기식 : support(X) = -----

N

← 전체 거래건수

n(커피 교 빵)

n(커피)

1.2 두 아이템 X와 Y의 지지도 ?

n(X ∩ Y) ← 아이템 X와 Y를 포함하는 거래건수

표기식 : support(X , Y) = -----

N

← 전체 거래건수

2. 신뢰도 : A 상품을 샀을 때 B 상품을 살 조건부 확률에 대한 척도

" 두 아이템의 연관규칙이 유용한 규칙일 가능성의 척도 "

n (X ∩ Y) ← 아이템 X와 Y가 동시에 발생하는 건수

표기식 : confidence(X → Y) = -----

n(X)

← 전체 거래건수에서 아이템 X의 건수

아이템 X를 포함하는 거래 중에서 아이템 Y도 포함하는 거래비율(조건부 확률)을 말한다.

신뢰도가 높을수록 유용한 규칙일 가능성이 높다고 할 수 있다.

3. 향상도 : 규칙이 우연에 의해 발생한 것인지를 판단하기 위해 연관성 정도를 측정하는 척도

" 두 아이템의 연관규칙이 우연인지 아닌지를 나타내는 척도 "

confidence(X → Y)

표기식 : lift (X → Y) = -----

support(Y)

P(B | A) ← p (A ∩ B)

향상식 : lift (A → B) = ----- = -----
P(B) P(A) P(B)

A와 B를 포함하는 거래건수 x 전체 거래건수

= -----
A를 포함하는 거래건수 x B를 포함하는 거래건수

아이템 X가 주어지지 않았을 때의 아이템 Y의 확률대비

아이템 x가 주어졌을때의 아이템 Y의 확률 증가 비율을 나타냄

향상도	설명	예시
향상도 = 1	서로 독립적 관계	과자와 후추
향상도 > 1	양(+)의 상관관계	빵과 버터
향상도 < 1	음(-)의 상관관계	설사약과 변비약

거래번호 거래 아이템

1	우유, 버터, 시리얼
2	맥주, 기저귀
3	동요패드, 쌀과자
:	:
:	:
99	동요패드, 쌀과자
:	
10000	

■ 연관 규칙을 컴퓨터에게 시켜야하는 이유?

아이템의 집합을 아이템의 개수만큼 만들려면 아이템의 개수를 k라고 하면 2의 k승개의 아이템의 집합이 생성되는데 아이템이 100개이면 2의 100승의 아이템의 집합이 생기므로 사람이 그 많은 데이터의 지지도, 신뢰도, 향상도를 다 구해서 그 중에 가장 값이 큰게 어떤건지 분석하기는 어렵다. 그래서 컴퓨터에게 시켜야한다.

컴퓨터에게 계산을 시켰을 때 컴퓨터에게 이렇게 많은 연산을 무턱대고 시키는게 아니라 효율적으로 하게끔 시켜야 한다.

■ 컴퓨터가 어떻게 연관 규칙을 찾는지 샘플로 알아보기

거래번호	아이템 목록
1	A, B, D
2	B, C
3	A, B, C, E
4	B, C, E

쿠팡의 물류센터에서 동요패드와 쌀과자의 조합을 알아냈듯이 위의 거래들을 보고 가장 적절한 아이템 목록의 조합을 알아내는게 우리의 목적이다.

첫번째 단계 : 위의 아이템들에 대해서 아이템 1개에 대한 지지도를 구한다.

원래 지지도는 구매건수/ 전체구매건수 인데 단순하게 아이템의 개수로 처리한다.

아이템 지지도

A	2
B	4
C	3
D	1
E	2

위의 결과에서 지지도가 1보다 큰 것만 추출해서 다시 정리

아이템 지지도

A	2
B	4
C	3
E	2

두번째 단계 : 이제 아이템들 간의 연관규칙을 알아야하므로 다시 아이템들간의 조합으로 구성하고 지지도를 다시 구한다. (두가지 조합)

아이템 지지도

A B	2
A C	1
A E	1
B C	3
B E	2
C E	2

위의 A B C E로 만들 수 있는 3개의 조합으로 지지도를 구하면 ?

아이템 지지도

A B C	1
A B E	1
A C E	1
B C E	2

지지도가 1보다 큰 것으로 다시 정리하면 B C E의 조합으로 물건을 추천해주면 된다.

■ 아프리오리 알고리즘 예제1. 맥주와 기저귀

금요일 밤에 남자들이 기저귀를 사러가면 맥주를 같이 사는 Rule 발견하기

1. 데이터를 로드한다.

```
x <- data.frame(  
beer=c(0,1,1,1,0),  
bread=c(1,1,0,1,1),  
cola=c(0,0,1,0,1),  
diapers=c(0,1,1,1,1),  
eggs=c(0,1,0,0,0),  
milk=c(1,0,1,1,1) )
```

x

결과 :

beer bread cola diapers eggs milk

1	0	1	0	0	0	1
2	1	1	0	1	1	0
3	1	0	1	1	0	1
4	1	1	0	1	0	1
5	0	1	1	1	0	1

(거래) (구매건수)

물건사이의 연관 규칙을 찾아냄 : 연관규칙

사람사이의 규칙을 찾아냄 : k-means

2. arules 패키지를 설치한다.

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

설명 : arules - 아프리오리 알고리즘을 사용할 수 있는 패키지

```
trans <- as.matrix( x, "Transaction")  
trans
```

Transaction 이란 옵션을 줘서 데이터 프레임을 행렬로 변경

3. apriori 함수를 이용해서 연관관계를 분석한다.

설명 : 지지도가 0.2 이상이고 신뢰도가 0.6 이상인 rule을 발견하시오!

```
rules1 <- apriori(trans, parameter=list(supp=0.2, conf=0.6, target="rules"))  
rules1
```

set of 49 rules # 49개의 rule이 발견이 되었다.

설명 : 규칙(rule)을 확인한다.

```
inspect(sort(rules1))
```

결과 :

lhs	rhs	support	confidence	coverage	lift	count
[1] {}	=> {milk}	0.8	0.8000000	1.0	1.0000000	4
[2] {}	=> {bread}	0.8	0.8000000	1.0	1.0000000	4
[3] {}	=> {diapers}	0.8	0.8000000	1.0	1.0000000	4
[4] {}	=> {beer}	0.6	0.6000000	1.0	1.0000000	3

[5] {beer} => {diapers}	0.6	1.0000000	0.6	1.2500000	3
[6] {diapers} => {beer}	0.6	0.7500000	0.8	1.2500000	3
[7] {milk} => {bread}	0.6	0.7500000	0.8	0.9375000	3
[8] {bread} => {milk}	0.6	0.7500000	0.8	0.9375000	3
[9] {milk} => {diapers}	0.6	0.7500000	0.8	0.9375000	3
[10] {diapers} => {milk}	0.6	0.7500000	0.8	0.9375000	3
[11] {bread} => {diapers}	0.6	0.7500000	0.8	0.9375000	3
[12] {diapers} => {bread}	0.6	0.7500000	0.8	0.9375000	3
[13] {cola} => {milk}	0.4	1.0000000	0.4	1.2500000	2
[14] {cola} => {diapers}	0.4	1.0000000	0.4	1.2500000	2
[15] {beer} => {milk}	0.4	0.6666667	0.6	0.8333333	2
[16] {beer} => {bread}	0.4	0.6666667	0.6	0.8333333	2
[17] {cola, milk} => {diapers}	0.4	1.0000000	0.4	1.2500000	2
[18] {cola, diapers} => {milk}	0.4	1.0000000	0.4	1.2500000	2
[19] {diapers, milk} => {cola}	0.4	0.6666667	0.6	1.6666667	2
[20] {beer, milk} => {diapers}	0.4	1.0000000	0.4	1.2500000	2
[21] {beer, diapers} => {milk}	0.4	0.6666667	0.6	0.8333333	2
[22] {diapers, milk} => {beer}	0.4	0.6666667	0.6	1.1111111	2
[23] {beer, bread} => {diapers}	0.4	1.0000000	0.4	1.2500000	2
[24] {beer, diapers} => {bread}	0.4	0.6666667	0.6	0.8333333	2
[25] {bread, diapers} => {beer}	0.4	0.6666667	0.6	1.1111111	2
[26] {bread, milk} => {diapers}	0.4	0.6666667	0.6	0.8333333	2
[27] {diapers, milk} => {bread}	0.4	0.6666667	0.6	0.8333333	2
[28] {bread, diapers} => {milk}	0.4	0.6666667	0.6	0.8333333	2
[29] {eggs} => {beer}	0.2	1.0000000	0.2	1.6666667	1
[30] {eggs} => {bread}	0.2	1.0000000	0.2	1.2500000	1
[31] {eggs} => {diapers}	0.2	1.0000000	0.2	1.2500000	1
[32] {beer, eggs} => {bread}	0.2	1.0000000	0.2	1.2500000	1
[33] {bread, eggs} => {beer}	0.2	1.0000000	0.2	1.6666667	1
[34] {beer, eggs} => {diapers}	0.2	1.0000000	0.2	1.2500000	1
[35] {diapers, eggs} => {beer}	0.2	1.0000000	0.2	1.6666667	1
[36] {bread, eggs} => {diapers}	0.2	1.0000000	0.2	1.2500000	1
[37] {diapers, eggs} => {bread}	0.2	1.0000000	0.2	1.2500000	1
[38] {beer, cola} => {milk}	0.2	1.0000000	0.2	1.2500000	1
[39] {beer, cola} => {diapers}	0.2	1.0000000	0.2	1.2500000	1
[40] {bread, cola} => {milk}	0.2	1.0000000	0.2	1.2500000	1
[41] {bread, cola} => {diapers}	0.2	1.0000000	0.2	1.2500000	1
[42] {beer, bread, eggs} => {diapers}	0.2	1.0000000	0.2	1.2500000	1
[43] {beer, diapers, eggs} => {bread}	0.2	1.0000000	0.2	1.2500000	1
[44] {bread, diapers, eggs} => {beer}	0.2	1.0000000	0.2	1.6666667	1
[45] {beer, cola, milk} => {diapers}	0.2	1.0000000	0.2	1.2500000	1
[46] {beer, cola, diapers} => {milk}	0.2	1.0000000	0.2	1.2500000	1

```
[47] {bread,
      cola,
      milk} => {diapers} 0.2 1.0000000 0.2 1.2500000 1
[48] {bread,
      cola,
      diapers} => {milk} 0.2 1.0000000 0.2 1.2500000 1
[49] {beer,
      bread,
      milk} => {diapers} 0.2 1.0000000 0.2 1.2500000 1
```

4. 위의 맥주와 기저귀 연관 관계를 시각화 하기

```
install.packages("sna")
install.packages("rgl")
library(sna)
library(rgl)

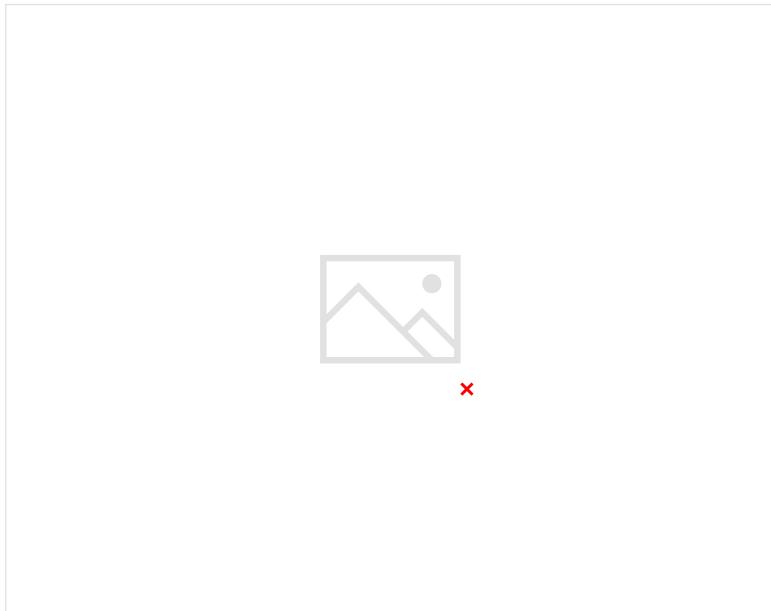
b2 <- t(as.matrix(trans)) %*% as.matrix(trans) # 희소행렬을 만든다.
b2.w <- b2 - diag(diag(b2)) # diag : 대각선이 1인 행렬, 위에서 만든 희소행렬에서 대각선에 해당하는
# 데이터를 빼서 대각선 데이터를 0이 되게 한다.
```

결과:

```
> b2
      beer bread cola diapers eggs milk
beer    3     2     1     3     1     2
bread   2     4     1     3     1     3
cola    1     1     2     2     0     2
diapers 3     3     2     4     1     3
eggs    1     1     0     1     1     0
milk    2     3     2     3     0     4
```

```
> b2.w
      beer bread cola diapers eggs milk
beer    0     2     1     3     1     2
bread   2     0     1     3     1     3
cola    1     1     0     2     0     2
diapers 3     3     2     0     1     3
eggs    1     1     0     1     0     0
milk    2     3     2     3     0     0
```

```
gplot(b2.w , displaylabel=T , vertex.cex=sqrt(diag(b2)) , vertex.col = "green" , edge.col="blue" ,
boxed.labels=F , arrowhead.cex = .3 , label.pos = 3 , edge.lwd = b2.w*2)
```



■ apriori 알고리즘 예제2 (보습학원과 연관된 업종은 ?)

건물상가에 서로 연관이 있는 업종이 무엇인가?
건물에 병원이 있으면 약국이 있는가?

보습학원이 있는 건물에는 어떤 업종의 매장이 연관되어 있는지 분석하는 실습

1. 데이터를 로드합니다.

```
build <- read.csv("building.csv" , header = T)
nrow(build) #20
```

2. na 를 0 으로 변경합니다.

```
build[is.na(build)] <- 0
```

3. 필요한 변수만 선별합니다.

```
build <- build[-1] # 건물 번호를 제외시킨다.
build
```

4. 연관규칙 패키지를 다운로드 받습니다.

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

5. 연관규칙 모델을 생성합니다.

```
trans <- as.matrix(build , "Transaction")
# trans_no <- as.matrix( build )

transactions class : 1과 0으로 이루어져 있는 데이터에서 0이 훨씬 많을 때
(spase format - 희소 형태의 데이터).
```

즉, 의미 없는 정보가 많고 크기가 커서 데이터를 처리하기 힘들 때 transactions class로 처리

설명 : 지지도 0.2 이상이고 신뢰도 0.6 이상인 규칙 만들기

```
rules1 <- apriori(trans , parameter = list(supp=0.2 , conf = 0.6 , target = "rules"))
rules1
```

6. 연관규칙을 확인합니다.

```
inspect(sort(rules1))
```

7. 시각화를 합니다.

여러 규칙들 중에서 보습학원 부분만 따로 검색

```
rules2 <- subset(rules1 , subset = lhs %pin% '보습학원' & confidence > 0.7)
```

설명 : subset 함수는 전체 규칙에서 일부 규칙만 검색하는 함수이다.

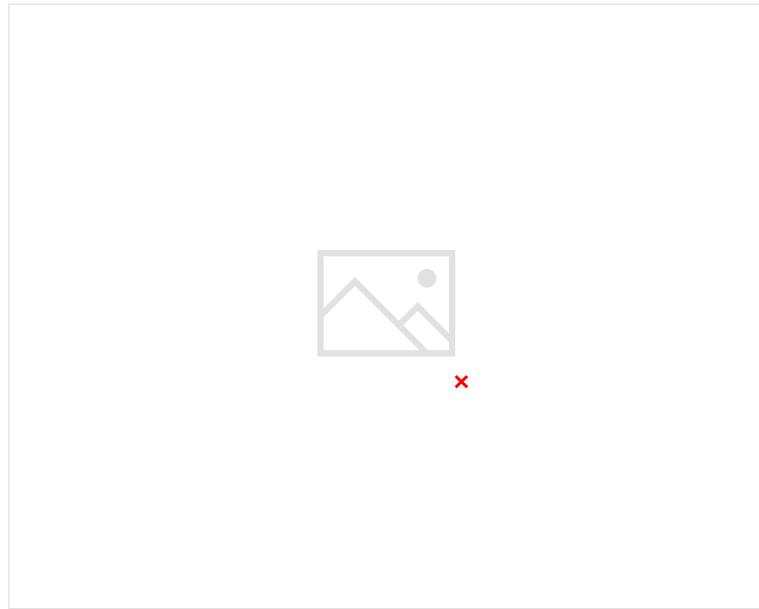
```
inspect(sort(rules2))
```

여러 규칙들 중에서 편의점에 연관된 부분만 따로 검색

```
rules3 <- subset(rules1 , subset = rhs %pin% '편의점' & confidence > 0.7)
rules3
inspect(sort(rules3))
```

#visualization

```
b2 <- t(as.matrix(build)) %*% as.matrix(build) #회소행렬로 변경  
  
install.packages("sna")  
install.packages("rgl")  
library(sna)  
library(rgl)  
  
b2.w <- b2 - diag(diag(b2)) # 회소행렬의 대각선을 0으로 변경  
  
#rownames(b2.w)  
#colnames(b2.w)  
  
gplot(b2.w , displaylabel=T , vertex.cex=sqrt(diag(b2)) , vertex.col = "green" , edge.col="blue" ,  
boxed.labels=F , arrowhead.cex = .3 , label.pos = 3 , edge.lwd = b2.w*2)
```



많이 언급되면 원이 크고, 지지도와 신뢰도가 강할수록 선이 굵어진다.

■ 연관규칙과 연관된 데이터 분석(12장)

" 사회적 연결망 데이터 분석 " → 사회학, 통계물리학

사회적 연결망 데이터 분석 ?

영어로는 [social network analysis](#)라고 하며, 줄여서 [SNA](#)라고 함.

사회적 연결망 분석은 개인과 집단간의 관계를 노드와 링크로서 모델링 해 그것의 구조나 확산 및 진화과정을 계량적으로 분석하는 방법론이다.

이 때 노드란 분석하고자하는 객체로서, 사람이나 사물 등을 말하는 것이다.

이 노드간의 관계를 나타내는 연결을 링크라고 한다.

노드에 연결된 링크의 수를 Degree라고 한다.

예제 : 독서 시간이 많은 사람은 친구가 없는가?

1. 이번주에 친구를 만난 횟수를 사회행렬로 구성함

```
paper <- read.csv("paper1.csv" , header = T)  
paper[is.na(paper)] <- 0  
paper
```

```
rownames(paper) <- paper[,1]  
paper <- paper[-1]
```

```
paper2 <- as.matrix(paper) # 행렬로 변환  
paper2
```

2. 이번주에 개별적으로 책을 읽은 시간 데이터를 로드한다.

```
book <- read.csv("book_hour.csv" , header = T)  
paper2  
book
```

결과:

```
name book  
1 지현 5  
2 지애 1  
3 덕채 20  
4 주경 0  
5 재환 18  
6 승리 2  
7 현주 3  
8 광희 5  
9 연수 1  
10 영진 14  
11 한상 2  
12 여원 4  
13 호상 1  
14 정훈 1  
15 영준 0  
16 우찬 0  
17 민정 10  
18 새미 4  
19 용석 7
```

```
library(sna)  
x11() #창을 새로 띄움
```

```
gplot(paper2 , displaylabels = T, boxed.labels = F , vertex.cex = sqrt(book[,2]) , vertex.col = "blue" ,  
vertex.sides = 20 ,  
edge.lwd = paper2*2 , edge.col = "green" , label.pos = 3)
```



×

■ 9장. k-means

머신러닝의 종류 3가지 ?

1. 지도학습 : knn, naivebayes, decision tree, 신경망, 회귀분석, 서포트벡터 머신
2. 비지도 학습 : 아프리오리 알고리즘, k-means 알고리즘
 - 지지도
 - 신뢰도
 - 향상도
3. 강화학습

* k-means (k개의 평균들) 군집화 이론이란 ?

k-means 알고리즘은 주어진 데이터를 [k개의 클러스터로 묶는 알고리즘](#)으로, 각 클러스터와 거리차이의 분산을 최소화하는 방식으로 동작한다.

이 알고리즘은 자율학습의 일종으로 [레이블\(정답 \)이 달려있지 않은 입력 데이터에 레이블\(정답 \)을 달아주는 역할](#)로도 활용되고 있다.

예 : emp 데이터프레임의 라벨이 될만한 컬럼은 무엇인가 ?

이 세상의 데이터는 정답이 없는 데이터가 많아서 라벨링 작업(노가다)을 사람이 일일이 해줘야 좋은 지도학습 모형을 만들 수 있다.
그래서 정답이 없는 데이터 속에서 어떤 패턴을 찾고 싶을 때 비지도 학습의 k-means 군집화 머신러닝 알고리즘을 활용한다.

k-means 현업사례 : 4기의 dba 출신 선배가 s* telecom에서 k-means를 이용해 어디에 기지국을 세우는 것이 가장 효율적인가를 분석한 사례



k-means 현
업 사례

오라클의 머신러닝 패키지를 이용해서 만든 사례

1. 우리나라에서 잘 활용하고 있는 사례

- a. 통신사에서 기지국을 세울 때 사용
- b. 병원에서 암 판별 머신러닝 모델을 만드는데 라벨링이 있는 지도학습 데이터를 훈련 시킬 때 비지도 학습을 같이 사용해서 모형의 정확도를 올리는데 참조

데이터가 악성인지 양성인지 라벨링을 하는 작업을 사람이 일일이 하면 실수를 할 가능성이 있다. 실수를 했는지 안했는지 빠르게 판단할 때 k-means를 활용한다.

- c. 마케팅 쪽 세그멘테이션(segmentation)에 활용이 된다. 고객들을 특성이 맞는 사람들끼리 군집화 한다.
- d. 보험회사에서 고객들을 segmentation해서 맞춤형 보험 상품 개발 및 광고를 진행한다.
- e. 미국의 유명한 사례 순찰 지역을 정하는데 범죄율을 사용해서 영화 마이너리 리포트처럼 범죄를 미리 예방하는데 사용을 했다.

→ "빅데이터 세상을 바꾸다"라는 kbs 다큐멘터리 참고

<https://www.youtube.com/watch?v=Hm9JPcHq8nQ>

■ k-means 실습1

1. 기본 데이터셋을 만든다.

```
c <- c(3,4,1,5,7,9,5,4,6,8,4,5,9,8,7,8,6,7,2,1)
row <- c("A","B","C","D","E","F","G","H","I","J")
col <- c("X","Y")
data <- matrix( c, nrow= 10, ncol=2, byrow=TRUE, dimnames=list(row,col))
```

data

결과 :

	X	Y
A	3	4
B	1	5
C	7	9
D	5	4
E	6	8

```
F 4 5  
G 9 8  
H 7 8  
I 6 7  
J 2 1
```

2. 위에서 만든 데이터 셋으로 plot 그래프를 그린다.

```
plot(data)  
  
km <- kmeans( data, 2)  
  
km$cluster  
  
# 설명 : x와 y좌표로 되어있는 data를 2개로 군집화해라!  
# km이라는 모델이 생성되었다.  
  
cbind(data, km$cluster)
```

결과 :

```
X Y  
A 3 4 1  
B 1 5 1  
C 7 9 2  
D 5 4 1  
E 6 8 2  
F 4 5 1  
G 9 8 2  
H 7 8 2  
I 6 7 2  
J 2 1 1
```

```
km$centers
```

결과 :

```
X Y  
1 3 3.8  
2 7 8.0
```

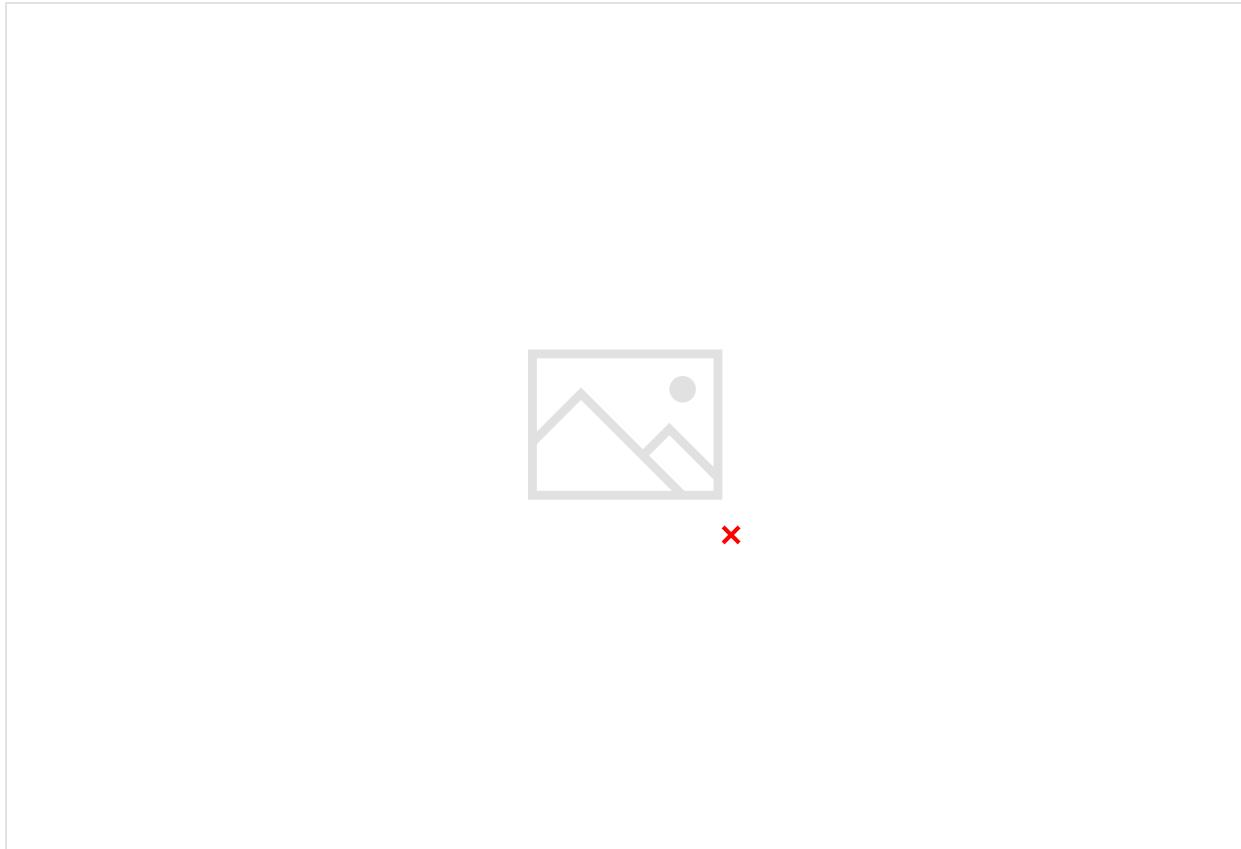
3. km 파라미터값들을 가지고 다시 한번 시각화하시오 !

```
plot( round(km$center), col=km$center, pch=22, bg="dark blue",  
      xlim=range(0:10), ylim=range(0:10) )
```

4. 원래 데이터를 위의 그래프에 합쳐서 출력하시오 !

```
plot( round(km$center), col=km$center, pch=22, bg="dark blue",
```

```
xlim=range(0:10), ylim=range(0:10) )  
par(new=T) #두개의 그래프를 합칠 때 사용  
plot( data, col=c("blue", "red"), xlim=range(0:10), ylim=range(0:10) )
```



■ 군집간의 거리 측정 방법 (빅데이터 기사시험 수제비 p3-58)

1. 최단 연결법 : 두 군집 사이의 거리를 각 군집에서 하나씩 관측값을 뽑았을 때 나타날 수 있는 거리의 최솟값으로 측정
2. 최장 연결법: 두 군집 사이의 거리를 각 군집에서 하나씩 관측값을 뽑았을 때 나타날 수 있는 거리의 최댓값으로 측정
3. 중심 연결법 : 두 군집간의 중심간의 거리를 측정
4. 평균 연결법 : 두 군집 사이의 거리를 각 군집에서 하나씩 관측값을 뽑았을 때 나타날 수 있는 거리의 평균값으로 측정
5. 와드 연결법 : 군집간의 거리에 기반하는 다른 연결법과는 다른 군집간의 오차 제곱합에 기초하여 군집을 수행

■ 군집간의 거리 계산하는 수학식

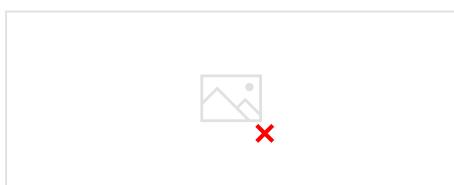
1. 연속형 변수 거리 계산

1.1 수학적 거리

1.1.1 유클리드 거리 계산 (R 머신러닝 p 410)



1.1.2 맨하튼 거리 계산



1.1.3 민코프스키 거리 계산

1.2 통계적 거리

1.2.1 표준화 거리 계산

1.2.2 마할라노비스 거리 계산

2. 명목형 변수 거리 계산

■ k-means 군집의 절차 (빅데이터 기사 수제비 p 3-61)

" 유클리드 거리로 어떻게 군집화를 했는가? "

단계	알고리즘	설명
1	k개 객체 선택	초기 군집중심으로 k개의 객체를 임의로 선택한다.
2	할당	자료를 가장 가까운 군집 중심에 할당한다. (임의로 중심 선택)
3	중심을 갱신	각 군집 내의 자료들의 평균을 계산하여 군집 중심을 갱신한다.
4	반복	군집 중심의 변화가 거의 없을 때까지 단계2와 3을 반복한다.

위의 설명을 그림으로 설명 :

<https://cafe.daum.net/oracleoracle/SeFi/458> 의 두번째 ppt

■ k-means의 단점

1. k값을 지정하기가 어렵다.

적당한 k 값을 계산하는 공식 : $k = \sqrt{n/2}$, n은 전체건수이다.

2. 이상치에 민감하다. (빅데이터기사시험 수제비 p3-61)



k-means

참고할 것

→ 이상치를 미리 제거하고 군집화 한다.

3. k-means로 군집화 한 결과에 대한 타당성 검증을 하기가 어렵다.

지도학습은 예측값과 정답과의 이원교차표를 통해서 좋은 모델인지 확인을 했는데 비지도 학습은 정답이 없어서 타당성 검증이 어렵다.

클러스터링한 결과를 가지고 기술 통계 기법을 써서 잘 군집화 했는지 분석을 한다.

Ex) R 머신러닝 책에서의 기술통계 기법 사용 결과 (p 430)

소설 미디어의 글들을 희소 행렬로 만들고 k-means로 군집화해서 사람들의 특성을 5가지로 분류함.

→ 공주형, 똑똑한 사람, 운동선수, 번죄자, 무력한 사람

※ k-means 시각화 할 때 fviz_cluster 함수가 아주 유용하다.

■ k-means 두번째 실습

1. 사과, 베이컨, 바나나, 당근, 셀러리, 치즈, 토마토 데이터를 준비한다. x축은 단맛, y축은 아삭한 정도

```
c <- c(10,9,1,4,10,1,7,10,3,10,1,1,6,7)
row <- c("APPLE","BACON","BANANA","CARROT","SAL","CHEESE","TOMATO")
col <- c("X","Y")
data <- matrix( c, nrow= 7, ncol=2, byrow=TRUE, dimnames=list(row,col))
data
```

결과:

	X	Y
APPLE	10	9
BACON	1	4
BANANA	10	1
CARROT	7	10
SAL	3	10
CHEESE	1	1
TOMATO	6	7

#설명 : 정답(라벨)은 없는 데이터이다.

```
plot(data)
```

2. 야채, 과일, 단백질 3가지를 k-means 가 잘 분류 했는지 시각화 해서 확인한다.

```
km <- kmeans(data, 3) #야채, 과일, 단백질 3가지여서 k=3으로 줌
```

```
km
```

```
cbind(data, km$cluster)
```

```
plot(round(km$center), col=km$center, pch=22, bg=km$center, xlim=range(0:10), ylim=range(0:10))
```

```
par(new=T) # 그래프 겹치기
```

```
plot( data, col=km$cluster+1, xlim=range(0:10), ylim=range(0:10), pch=22, bg=km$cluster+1 )
```



✗

3. 야채, 과일, 단백질 3가지를 k-means 가 잘 분류 했는지 시각화 해서 확인한다.

```
install.packages("factoextra")
library(factoextra)
km <- kmeans(data,3)
fviz_cluster( km, data = data, stand=F)
```



✗

■ k-means 세번째 실습(국영수 점수 데이터)

국영수 점수 데이터를 가지고 k 값을 4 두고 학생들을 분류하시오!

1. 수학,영어 둘다 잘하는 학생들
2. 수학은 잘하는데 영어를 못하는 학생들
3. 영어는 잘하는데 수학을 못하는 학생들
4. 수학,영어 둘다 못하는 학생들

첨부파일 **academy.csv**

1. 데이터를 로드한다.

```
academy <- read.csv("academy.csv")
```

```
# 수학점수와 영어점수만 선택
```

```
academy <- academy[ , c(3,4) ]
```

2. k 값을 4로 주고 비지도학습 시켜 모델을 생성한다.

```
km <- kmeans( academy, 4)
```

```
km
```

3. 시각화를 한다.

```
library(factoextra)
```

```
fviz_cluster(km , data=academy, stand=F)
```

4. 학생번호, 수학점수, 영어점수, 분류번호가 같이 출력되게하시오 !

```
academy <- read.csv("academy.csv")  
cbind( academy[ ,c(1,3,4)], km$cluster)
```

```
academy_seg[km$cluster==1, 1]
```

```
academy_seg[km$cluster==1, "학생 번호"]
```

설명 : 마케팅을 위해서 k-means를 이용해서 세그멘테이션을 해서 관련 학생들에게 광고를 따로 진행

■ k-means 네번째 실습(SNS의 글로 같은 성향을 가진 사람들을 분류)

예제를 실습 하기 전에 미리 알아야 하는 내용 2가지

1. imputation (결측치를 다른 값으로 치환)

2. ave 함수

▣ 결측치를 다른 값으로 대체하는 방법 3가지?

1. 최빈값으로 치환 (hot deck)
2. 평균값으로 치환 (mean imputation)
3. 회귀의 추정계수로 치환 (regression imputation)

▣ ave 함수

Subsets of x[] are averaged, where each subset consist of those observations with the same factor levels.

예 : 1:10

```
ave(1:10) # 1~10까지의 숫자들의 평균값이 입력된 숫자의 개수만큼 출력되고 있다.
```

■ k-means 5번째 실습 (쇼설 미디어에 같은 성향을 갖는 사람들을 분류)

2006년도에 잘 알려진 sns의 30000명의 미국 고등학생 데이터

첨부파일 snsdata.csv

1. 데이터를 로드한다.

corpus 패키지를 가지고 만들어낸 데이터이고 원래 raw 데이터는 sns의 글들이다.

R책의 부록실습 참고해보기

```
teens <- read.csv("snsdata.csv")
```

설명 : 30000건이나 되는 데이터이다. sns 글에서 특정단어에 대한 언급이 있었으면 1로 표시하고 없었으면 0으로 표시한다.

2. 성별이 남자가 몇명이고 여자가 몇명인지 확인한다.

```
table(teens$gender)
```

결과 :

	F	M
22054	5222	

3. 성별에 NA 가 몇개인지도 출력되게하시오

```
table(teens$gender, useNA="ifany")
```

결과 :

```
F      M <NA>  
22054 5222 2724
```

설명 : useNA="ifany" 옵션을 주어야 결측치(NA)에 대한 건수도 확인할 수 있다.

결측치가 있으면 유clidean 거리계산 공식으로 거리계산을 할 수 없으므로 결측치를 다른값으로 치환해 줘야한다.

4. 고등학생 데이터라는 정확한 데이터 분석을 위해서 나이가 13세 ~ 20세 가 아니면 다 NA 처리해라 !

```
teens$age <- ifelse(teens$age>=13 & teens$age <20, teens$age, NA)
```

설명 : 나이가 13세 이상이고 20세보다 작으면 나이가 나오게 하고 아니면 NA가 나오게 해라

5. 유clidean 거리 계산을 위해 성별에 관련한 더미변수 2개를 생성한다.

설명 : 성별이 여자이고 결측치가 아닌 데이터는 1을 출력하고 아니면 0으로 출력하시오.

파생변수를 female로 생성한다.

```
teens$female <- ifelse(teens$gender=="F" & !is.na(teens$gender), 1, 0)
```

#성별이 없는 사람들은 no_gender라는 파생변수를 통해 처리될 수 있도록

성별이 NA이면 1아니면 0을 출력하는 no_gender 파생변수 생성한다.

```
teens$no_gender <- ifelse(is.na(teens$gender),1,0)
```

#먼저 기존 성별 데이터에 NA 값이 몇개가 있는지 확인한다.

```
table(teens$gender, useNA="ifany")
```

#여성다면 1 아니면 0인 건수가 어떻게 되는지 확인

```
table(teens$female, useNA="ifany" )
```

결과:

```
0      1  
7946 22054
```

#성별이 NA이면 1 아니면 0인 건수가 어떻게 되는지 확인

```
table(teens$no_gender, useNA="ifany")
```

결과 :

```
0      1
```

설명 : 기존 컬럼인 gender 컬럼의 데이터에는 NA가 포함되어 있어서 유클리드 거리계산을 제대로 수행할 수 없었지만 성별과 관련해서 파생변수로 만든 female과 no_gender는 NA가 없으므로 유클리드 거리계산을 가능하게 할 수 있는 데이터이다.

6. 나이가 결측치로 나온 데이터를 졸업년도로 나이를 추정해서 결측치를 채워넣는 작업

나이의 결측치도 5523건이나 되므로 성별처럼 파생변수를 생성하던지 아니면 다른 값으로 치환하면 된다. 보통 숫자는 3가지(평균, 최빈, 회귀로 추정한 값) 중에 하나로 치환한다. 아래의 코드는 졸업년도별 나이의 평균값을 구해서 그 값으로 나이의 결측치를 치환한다.

```
ave_age <- ave(teens$age, teens$gradyear, FUN=function(x) mean(x, na.rm=TRUE) )
teens$age <- ifelse( is.na(teens$age), ave_age, teens$age)
```

7. sns 에 나타났던 관심사 횟수를 표현하는 36개의 수치형 데이터 컬럼을 정규화 시킨다.

```
interests <- teens[5:40]
interests_z <- as.data.frame(lapply(interests, scale))
```

설명 : scale 함수를 이용해서 정규화를 시키는데 scale 함수는 평균이 0이고 표준편차가 1인 데이터로 정규화 시킴

8. kmeans 함수로 5개의 클래스로 분류 한다.

```
set.seed(2345)
teen_clusters <- kmeans(interests_z, 5)
```

```
teen_clusters
```

9. 각 클래스의 갯수가 각각 어떻게 되는지 확인한다.

```
teen_clusters$size
```

10. 클러스터의 중심점의 좌표를 확인한다

```
teen_clusters$centers
```

11. 어떻게 클러스터링 되었는지 확인한다.

```
teen_clusters$cluster
```



설명 : 군집화를 했으면 그 군집화한 결과의 타당성을 조사해야 하는데 기술통계를 이용해서 타당성을 조사한다.

분석결과 설명 : 5개로 분류를 했는데 (**princesses**(공주형), **brains**(두뇌형), **criminals**(범죄형), **athletes**(운동형), **basket cases**(무력함형)) 전체적으로 sns 사용자의 74%가 여성이고 공주형은 거의 90%가 여성인 반면 **brains**와 **basket cases**는 70%가 여성이다.

평균적으로 공주형이 가장 많은 친구를 가지며, 다음으로는 운동형과 두뇌형이다.

■ [빅데이터 기사] SOM(Self Organizing Map) 분석

머신러닝 종류:

1. 지도학습 : 정답이 있는 데이터로 학습

분류 : knn, naivebayes, decision tree, 신경망, 서포트 벡터 머신

예측 : 회귀분석, 신경망

2. 비지도학습 : 정답이 없는 데이터로 학습

연관규칙, k-means, SOM

3. 강화학습 : 특정 환경에 적응하기 위해 상과 벌을 통해 학습

■ SOM (Self - Organization Maps) 수제비 p.3-26

비지도학습 신경망으로 고차원의 데이터를 이해하기 위해 저차원의 뉴런으로 정렬하여 지도(map)의 형태로 형상화 하는 기법

k-means 군집화 : 유클리드 거리공식을 이용해서 가장 가까운 이웃끼리 군집화하는 알고리즘

SOM : 비지도학습 + 신경망

■ 아이리스 데이터를 som 비지도학습 신경망을 이용해서 군집화하는 실습

실습코드:

1. 아이리스 데이터의 컬럼 이름 확인 및 건수 확인

```
colnames(iris)  
nrow(iris) #150건
```

2. 아이리스의 정답인 Species가 팩터이므로 level을 확인

```
levels(iris$Species) # "setosa" "versicolor" "virginica"
```

3. som 패키지 설치

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

4. 훈련데이터와 테스트 데이터를 3대 1로 분류 (훈련데이터 100건, 테스트는 50건)

```
train <- sample(1:150, 100) #무작위로 100개 추출 (학습데이터)
train_Set <- list( x = as.matrix(iris[train,-5]), Species = as.factor(iris[train,5])) #학습데이터 list형
test_Set <- list(x = as.matrix(iris[-train,-5]), Species = as.factor(iris[-train,5])) #테스트 데이터 list형
```

설명 : som 함수가 list 형태의 데이터를 원하므로 list 형태로 변환해준다.

5. somgrid 함수를 이용해서 경쟁층을 구현

```
gr <- somgrid(xdim = 3, ydim = 5, topo = "hexagonal") #grid 갯수 및 모양 설정
```

#토폴로지(영어: topology, 문화어: 망구성방식)는 컴퓨터 네트워크의 요소들(링크, 노드 등)을 물리적으로 연결해 놓은 것

6. 훈련데이터 100개로 som 학습시키기

rlen = 200은 150개의 데이터를 200번 학습 시키겠다라는 뜻

alpha = c(0.05, 0.01) 러닝 레이트를 0.05에서 시작해서 0.01까지 학습시킨다.

```
ss <- supersom(train_Set, gr, rlen = 200, alpha = c(0.05, 0.01)) #som 학습하기
```

ss

#실습시 나오는 som 함수 파라미터 설명

<https://www.rdocumentation.org/packages/kohonen/versions/3.0.10/topics/supersom>

7. 군집화가 잘 되었는지 시각화 하면서 확인하는 방법

학습이 진행될수록 경쟁층의 뉴런과 입력 데이터와의 거리가 짧아지는지 확인

```
plot(ss, type="changes")
```

8. som 모델의 각 뉴런(경쟁층의 뉴런)이 몇 개의 입력 데이터와 맵핑이 되는지 확인

```
plot(ss, type="count", main="Node Counts")
```

9. 경쟁층의 각 뉴런끼리의 거리를 확인하는 방법

나는 친구가 10명이고 옆에 뉴런은 친구가 8명인데 알고보니 옆의 뉴런이 나와 같은 성향의

뉴런이어서 같은 종류였음을 확인하고 싶을 때 유용함

```
plot(ss, type="dist.neighbours", main = "SOM neighbour distances")
```

10. 경쟁층의 뉴런에 속한 입력층의 친구들(아이리스는 3가지 종류중에 무엇인지)이 누구인지 # 확인하는 방법

```
plot(ss, type="codes")
```

11. 테스트 데이터를 잘 맞추는지 확인

test_Set\$Species #실제 정답

```
test.pred <- predict(ss, newdata = test_Set) # 테스트 데이터로 예측
```

```
test.pred$predictions$Species
```

12. 예측값과 실제값이 일치하는지 확인

```
table(test.pred$predictions$Species,test_Set$Species)
```

결과:

	setosa	versicolor	virginica
setosa	16	0	0
versicolor	0	15	0
virginica	0	0	19

■ 10장. 모델 성능 평가

학습목표 : 앞에서는 우리가 여러가지 머신러닝 모델을 생성 해 보았다. 여러가지 머신러닝 모델에서 현재 분석하는 데이터에 맞는 모델은 이 모델이 맞다라고 판단할 수 있게 하는데 도움을 주는 내용이 이번장의 내용이다.

관련된 면접 질문 :

제출한 데이터 분석 포트폴리오에서 이 머신러닝 모델을 선택한 이유가 무엇입니까?

답변 : 모델 성능 평가를 통해서 확인한 결과 이 모델이 가장 우수했습니다.

■ 모델 성능 평가가 중요한 이유는 무엇일까?

머신러닝이 수행한 결과(분류, 예측)에 대한 공정한 평가를 통해 머신러닝이 앞으로도 미래의 데이터에 대해서 잘 분류하고 예측할 수 있도록 해주고 분류결과가 요행수로 맞힌게 아니라는 것을 확신하게 해주며 분류 결과를 좀 더 일반화 할 수 있기 때문이다.

■ 그동안 성능 평가에서 사용한 방법 ?

분류할 때는 정확도로 확인했고 회귀분석일 때는 결정계수로 확인

■ 모델 성능평가가 정확도만으로는 충분하지 않은 이유?

암판정을 하는 분류기가 99%의 정확도를 갖고 있다고 하면 1%의 오류율이 있기 때문에 어떤 데 이터에 대해서는 오류를 범할 수도 있게 된다.

그래서 정확도 만으로는 성능 측정에 충분하지 않다.

정확도와 더불어서 분류기의 유용성에 대한 다른 성능 척도를 정의하는게 중요하다.

" 정확도 + 다른 성능 척도 "

그럼 다른 성능 척도에는 무엇이 있는가?

1. 카파통계량
2. 민감도와 특이도
3. 정밀도와 재현율
4. Roc 곡선
5. F1 score

예 :

의사 曰 -

갑상선 암판정을 받고 의사 선생님이 하는 말이 갑상선이 목에 2개가 있는데 한쪽에 암 종양이 있습니다. 한쪽만 떼어낼까요? 둘다 떼어낼까요? 지금 결정하세요.

기존의 갑상선 질환의 환자들의 경우 한쪽만 떼어냈을 때 다른 한쪽에 암이 걸릴 확률이 23%입니다.

■ 정확도 계산하는 방법 수제비 책 p 4-6

실제 분류 범주를 정확하게 예측한 비율

전체 예측에서 참 긍정(TP)와 참 부정(TN)가 차지하는 비율

$$TP + TN$$

정확도 = -----

$$TP + TN + FP + FN$$

Positive : 관심범주

예 : 암을 잘 분류하는 분류기를 만들고 싶다면?

(관심범주가 암이기 때문에 암이 Positive, 정상이 Negative)

스팸메일을 분류하는 분류기를 만들고 싶다면 ?

관심범주 : 스팸메일 / 스팸메일(positive), 햄메일 (negative)

예측이 정확한 경우 : TP (True Positive) : 실제값이 positive이고 예측값도 positive인 경우

TN (True Negative) : 실제값이 negative이고 예측값도 negative인 경우

예측이 틀린 경우 : FP (False Positive) : 실제값은 negative 이었으나 예측값은 positive인 경우

FN (False Negative) : 실제값은 positive 이었으나 예측값은 negative인 경우

게시글 791 스팸 분류기 이원 교차표

<https://cafe.daum.net/oracleoracle/Sg0x/791>

■ 카파통계량 (R책 p 499, 수제비책 p 4-6)

두 관찰자간의 측정 범주값에 대한 일치도를 측정하는 방법이다.

$$k = \frac{Pr(a) - Pr(e)}{1 - Pr(e)}$$

Pr(a) : 데이터에서 관찰된 2명의 평가자들의 일치확률

Pr(e) : 2명의 평가자들이 데이터로부터 계산된 확률적 일치확률(우연히 일치할 확률)

카파통계량이 0이면 완전 불일치이고 1이면 완전 일치이다.

수제비 책 p 4-6

k	일치정도
0.8 ~ 1.0	매우 좋은 일치
0.6 ~ 0.8	좋은 일치
0.4 ~ 0.6	보통 일치
0.2 ~ 0.4	어느 정도 일치
0.0 ~ 0.2	거의 일치하지 않음

예시 : 시험을 응시한 학생이 100명이라고 할 때, 2명의 평가자가 합격, 불합격을 각각 판정하고 두 평

가자의 일치도를 아래와 같이 보여주고 있다.



Pr(a) : 데이터에서 관찰된 2명의 평가자들의 일치확률

$$Pr(a) = \frac{40 + 30}{100} = 0.7$$

평가자 A : 합격을 60번, 불합격을 40번 주었다.

평가자 A는 합격을 $60/100 = 0.6$ 확률

불합격을 $40/100 = 0.4$ 확률

평가자 B : 합격을 50번, 불합격을 50번을 주었다.

평가자 B는 합격을 $50/100 = 0.5$ 확률

불합격을 $50/100 = 0.5$ 확률

Pr(e) : 2명의 평가자들이 데이터로부터 계산된 확률적 일치확률(우연히 일치할 확률)

평가자 A와 평가자 B 둘 모두 "합격"을 줄 확률은?

$$0.6 \times 0.5 = 0.3$$

평가자 A와 평가자 B 둘 모두 "불합격"을 줄 확률은?

$$0.4 \times 0.5 = 0.2$$

Pr(e)는 데이터로부터 계산된 확률적으로 일치할 확률이므로 이 둘을 더해서
 $0.3 + 0.2 = 0.5$ 이다.

$$k = \frac{Pr(a) - Pr(e)}{1 - Pr(e)} = \frac{0.7 - 0.5}{1 - 0.5} = \frac{0.2}{0.5} = 0.4$$

k	일치정도
0.8 ~ 1.0	매우 좋은 일치
0.6 ~ 0.8	좋은 일치
0.4 ~ 0.6	보통 일치
0.2 ~ 0.4	어느 정도 일치
0.0 ~ 0.2	거의 일치하지 않음

■ 민감도와 특이도 (R책 p 454 수제비 p 4-6)

유용한 분류기를 찾으려면 보통 지나치게 보수적인 예측과 지나치게 공격적인 예측사이에 균형이 필요하다.

보수적인 예측과 공격적인 예측에 대한 것을 정하는 기준이 되는 정보가 민감도와 특이도이다.

* 민감도 : 실제로 '긍정'인 범주 중에서 '긍정'으로 올바르게 예측한 비율

$$\text{sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

예측	
실제	TN FP
	FN TP

예 : 실제 구매한 사람중에서 구매할 것이라고 예측한 사람의 비율

* 특이도 : 실제로 '부정'인 범주 중에서 '부정'으로 올바르게 예측한 (TN)

$$\text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

실제로 구매안한 사람중에서 구매 안할 것으로 예측한 비율

민감도와 특이도가 카파 통계량처럼 0~1까지의 범위에 있으며, 값이 1에 가까울수록 바람직하나 실제로는 한쪽이 높으면 한쪽이 낮아져서 둘다 높게 맞출 수 없다.

만약 유방암을 예측하는 머신러닝 모델을 여러 개 만들어놓고 그 중 하나를 선택해야 한다면

	knn 모델	decision tree	신경망
정확도 :	99%	99%	99%
민감도 :	0.6	0.7	0.7
특이도 :	0.5	0.4	0.5

민감도를 최고로 높게 맞춰놓고 그 중에 특이도가 높은것을 선택한다.

민감도가 높으면 특이도가 다소 낮더라도 가치는 충분하다.

■ 정밀도와 재현율 p 456

정밀도 : 참인것으로 예측한 것 중에 실제 참인것의 비율

$$\text{precision} = \text{TP} / (\text{TP} + \text{FP})$$

재현율 : 실제 참인것 중에서 참이라고 예측한 비율(민감도와 같다)

$$\text{재현율(민감도)} = \text{TP} / (\text{TP} + \text{FN})$$

	예측	
실제	TN	FP
	FN	TP

$$\text{특이도} : \text{TN} / (\text{TN} + \text{FP})$$

1. 소극적 예측 : 암이라고 판단하는 것 자체를 소극적으로 봐서 확실한 경우가 아니면 암으로 판단하지 않는 것이다.

정밀도 ↑ 재현율 ↓

2. 공격적 예측 : 조금만 의심이가도 다 암이라고 판단한다.

정밀도 ↓ 재현율 ↑

이 모델은 암인데 암이라고 판단을 하면 잘한거고 암이 아닌 사람을 암으로 예측한 것인데 실제로 나중에 검사결과가 정상이었다면 오히려 다행인 것이어서 다시 재검사 받으면 된다.

정리 :

(오늘 배운 것) :

카파통계량, 민감도, 특이도, 정밀도, 재현율(민감도와 같다)

■ 지난시간에 배웠던 R수업 복습

머신러닝의 종류 :

1. **지도학습** : 분류 - knn, naivebayes, decision tree, rule, riper, 신경망, 서포트 벡터 머신, 로지스틱 회귀, 랜덤포레스트, xgboost

예측 - 다중회귀분석

2. **비지도학습** : 연관분석, k-means, SOM

3. **강화학습**

데이터를 변경해서 위의 알고리즘을 돌려보면서 예측과 분류를 따로 연습

카파통계량 : 두 평가자간의 평가 결과가 우연에 의한 것인지 아닌지를 나타내는 일치도

		예측			
		NO	YES		
실제	NO	TN	FP	특이도	$\frac{TN}{TN + FP}$
	YES	FN	TP	민감도	$\frac{TP}{FN + TP}$
		정밀도			
		$\frac{TP}{TP + FP}$			

예측 : 평가자 A

실제 : 평가자 B

일치도 - 0 ~ 1

1에 가까울수록 정확도 ↑

Positive : 관심범주 (암, 스팸메일 ...)

TP : True (맞췄다) + Positive (암환자 / 관심범주)

↓
(암환자를) 암환자로 잘 판단했다.

FP : False(잘못 판단했다) + Positive (암환자로 / 관심범주)

↓
(정상환자를) 암환자로 잘못 판단했다.

TN : True (맞췄다) + Negative (정상환자로)

↓
(정상환자를) 정상환자로 잘 판단했다.

FN : False (잘못 판단했다) + Negative (정상환자로)

↓
(암환자를) 정상환자로 잘못 판단했다.

민감도 : 평가자 실제를 기준으로 관심범주(암환자)를 잘 판단한 비율
(재현율)

TP

TP + FN

특이도 : 평가자 실제를 기준으로 관심범주가 아닌것(정상환자)을 잘 판단한 비율
TN

TN + FP

정밀도 : 평가자 예측을 기준으로 관심범주(암환자)를 잘 판단한 비율
TP

TP + FP

■ 카파 통계량, 정확도, 민감도, 특이도, 정밀도, 재현율을 출력하는 실습 :

#1. 데이터를 로드한다.

```
credit <- read.csv("credit.csv", stringsAsFactor=TRUE)
str(credit)
```

#설명 : 독일은행의 대출금 상환여부 데이터를 로드

#2. 데이터에 각 컬럼들을 이해한다.

#라벨 컬럼 : default ---> yes : 대출금 상환 안함 / no : 대출금 상환

```
prop.table( table(credit$default) )  
summary( credit$amount)
```

#3. 데이터가 명목형 데이터인지 확인해본다.

```
str(credit)
```

#4. 데이터를 shuffle 시킨다.

```
set.seed(31)  
credit_shuffle <- credit[ sample( nrow(credit) ), ]
```

#5. 데이터를 9 대 1로 나눈다.

```
train_num <- round( 0.9 * nrow(credit_shuffle), 0)  
  
credit_train <- credit_shuffle[1:train_num , ]  
  
credit_test <- credit_shuffle[(train_num+1) : nrow(credit_shuffle), ]  
  
nrow(credit_train) #900  
  
nrow(credit_test) #100
```

#6. C5.0 패키지와 훈련 데이터를 이용해서 모델을 생성한다.

```
library(C50)  
  
credit_model <- C5.0( credit_train[ ,-17] , credit_train[ , 17] )
```

#7. 위에서 만든 모델을 이용해서 테스트 데이터의 라벨을 예측한다.

```
credit_result <- predict( credit_model, credit_test[ , -17] )
```

#8. 이원 교차표로 결과를 확인한다.

```
library(gmodels)  
  
CrossTable( credit_test[ , 17], credit_result )
```

■ 실제값과 예측값 대입

```
actual_type <- credit_test[ , 17] #실제 테스트 데이터의 라벨( 정답 )
predict_type <- credit_result # 머신러닝 모델이 예측한 정답
positive_value <- 'yes'       # 관심범주 yes : 돈 못 갚은 사람
negative_value <- 'no'        # 관심범주가 아닌것
```

■ 정확도

```
g <- CrossTable( actual_type, predict_type )

x <- sum(g$prop.tbl *diag(2)) # 정확도 확인하는 코드
x
```

#■ 카파통계량

```
#install.packages("vcd")
library(vcd)

table( actual_type, predict_type)

Kappa( table( actual_type, predict_type) )
```

#■ 민감도

```
head(credit_results)

#install.packages("caret")
library(caret)
sensitivity( predict_type, actual_type,
             positive=positive_value)
```

#■ 특이도

```
specificity( predict_type, actual_type,
              negative=negative_value)
```

#■ 정밀도

```
posPredValue( predict_type, actual_type,
               positive=positive_value)
```

#■ 재현율

```
sensitivity( predict_type, actual_type,
               positive=positive_value)
```

■ ROC 곡선

<https://cafe.daum.net/oracleoracle/Sg0x/827> 참조

Receiver Operating Characteristic

이진 분류기의 성능 평가 기법 중 하나이다.

2차원 위에 그려지는 그래프이고 x축이 False Positive Rate이고 y축이 True Positive Rate이다.

0. ROC 커브 설명 그림0



ROC 커브의 x축은 FPR(False Positive Rate)로 정상환자를 암환자로 잘못 판정한 비율이고
y축은 TPR(True Positive Rate)로 암환자를 암환자로 잘 판정한 비율이다.

FPR은 낮추면서 TPR을 높일 수 있는 모형이 가장 좋은 모형이고 최대한 FPR은 낮추면서 TPR을 높일 수 있는 cut off 지점을 찾는게 ROC 그래프를 그리는 이유가 된다.

1. ROC 커브 설명 그림1



다른 성능척도인 민감도와 특이도를 숫자로만 보는게 아니라 시각화해서 여러 모델중에서 가장 좋은 모델을 가장 좋은 모델을 쉽게 찾게해주는 그래프가 바로 ROC 그래프이다.

* 독일은행 데이터로 ROC 그래프 그리기 실습

#1. 데이터를 로드한다.

```
credit <- read.csv("credit.csv", stringsAsFactor=TRUE)
str(credit)
```

#2. 데이터에 각 컬럼들을 이해한다.

```
#라벨 컬럼 : default ---> yes : 대출금 상환 안함
#no : 대출금 상환
```

```
prop.table( table(credit$default) )
summary( credit$amount)
```

#3. 데이터가 명목형 데이터인지 확인해본다.

```
str(credit)
```

#4. 데이터를 shuffle 시킨다.

```
set.seed(31)
credit_shuffle <- credit[ sample( nrow(credit) ), ]
```

#5. 데이터를 9 대 1로 나눈다.

```
train_num <- round( 0.9 * nrow(credit_shuffle), 0)
```

```
credit_train <- credit_shuffle[1:train_num , ]  
credit_test <- credit_shuffle[(train_num+1) : nrow(credit_shuffle), ]
```

#6. C5.0 패키지와 훈련 데이터를 이용해서 모델을 생성한다.

```
library(C50)  
  
credit_model <- C5.0( credit_train[ ,-17] , credit_train[ , 17] )
```

#7. 위에서 만든 모델을 이용해서 테스트 데이터의 라벨을 예측한다.

```
credit_result <- predict( credit_model, credit_test[ , -17] )
```

#8. 이원 교차표로 결과를 확인한다.

```
library(gmodels)  
  
CrossTable( credit_test[ , 17], credit_result )
```

#■ 실제값과 예측값 대입

```
credit_test_prob <- predict(credit_model, credit_test[ , -17], type = "prob")  
  
head(credit_test_prob)
```

결과 :

	no	yes
550	0.8669090	0.1330910
601	0.1747222	0.8252778
696	0.8669090	0.1330910
949	0.8669090	0.1330910
85	0.8090572	0.1909428
200	0.8090572	0.1909428

설명 : type='prob'옵션을 주게 되면 확률이 출력된다.

```
# combine the results into a data frame  
  
credit_results <- data.frame(actual_type =credit_test[ , 17], # 테스트 데이터의 실제정답  
                                predict_type = credit_result,          # 테스트 데이터에 대한 예측값  
                                prob_yes = round(credit_test_prob[ , 2], 5), # 관심범주(yes)일 확률  
                                prob_no = round(credit_test_prob[ , 1], 5)) # 관심범주가 아닐(no) 확률  
  
head( credit_results )
```

결과 :

```
actual_type predict_type prob_yes prob_no
550      no        no  0.13309 0.86691
601      no       yes  0.82528 0.17472
696      no        no  0.13309 0.86691
949     yes        no  0.13309 0.86691
85       no        no  0.19094 0.80906
200     yes        no  0.19094 0.80906
```

#3. 예측 데이터 프레임을 csv로 저장합니다.

```
# uncomment this line to output the sms_results to CSV
write.csv(credit_results, "final_results.csv", row.names = FALSE)
```

#■ 실제값과 예측값 대입

```
actual_type <- credit_test[ , 17] # 테스트 데이터의 라벨
predict_type <- credit_result # 테스트 데이터의 예측값
positive_value <- 'yes' # 관심범주
negative_value <- 'no' # 관심범주가 아닌것
```

#■ 정확도

```
g <- CrossTable( actual_type, predict_type )
x <- sum(g$prop.tbl *diag(2)) # 정확도 확인하는 코드
x
```

#■ 카파통계량

```
#install.packages("vcd")
library(vcd)

table( actual_type, predict_type )
Kappa( table( actual_type, predict_type) )
```

#■ 민감도

```
head(credit_results)

#install.packages("caret")
library(caret)
sensitivity( predict_type, actual_type,
```

```
positive=positive_value)
```

#■ 특이도

```
specificity( predict_type, actual_type,  
negative=negative_value)
```

#■ 정밀도

```
posPredValue( predict_type, actual_type,  
positive=positive_value)
```

#■ 재현율

```
sensitivity( predict_type, actual_type,  
positive=positive_value)
```

#■ ROC 곡선 그리기

```
#install.packages("ROCR")  
library(ROCR)  
head(credit_results) # 3번째 컬럼과 4번째컬럼의 확률을 확인한다.  
pred <- prediction(predictions = credit_results$prob_yes,  
                      labels = credit_results$actual_type)  
pred
```

설명 : prediction(predictions = 관심범주의 확률, labels = 정답)
pred에 ROC 커브를 그리기 위한 100개의 데이터 포인트가 담긴다.

```
> pred  
A prediction instance  
with 100 data points
```

```
# ROC curves
```

```
perf <- performance(pred, measure = "tpr", x.measure = "fpr")  
# 설명 : performance( 100개의 데이터 포인트, y축값, x축값 )  
# x축인 fpr과 y축인 tpr에 해당하는 실제 data point 24개 생성됨
```

```
plot(perf, main = "ROC curve for SMS spam filter", col = "blue", lwd = 2)
```

```
# add a reference line to the graph
```

```
# 대각선 출력
```

```
abline(a = 0, b = 1, lwd = 2, lty = 2)
```

설명 : a는 직선의 절편, 1은 직선의 기울기
lwd는 선의 굵기, lty=2는 점선으로 표현

```

# calculate AUC

# AUC : Area under Curve의 약자로 곡선 아래쪽의 넓이를 말한다.
# 넓이가 넓을수록 좋다. ( 0 ~1 사이로 출력된다 )

perf.auc <- performance(pred, measure = "auc")
str(perf.auc)
unlist(perf.auc@y.values) # 0.6668286

```

■ ROC 그래프에서 cut off 지정하기

<https://cafe.daum.net/oracleoracle/Sg0x/827>

로지스틱 회귀로 데이터를 분류하고 ROC 커브를 그린 다음 cutoff 지점을 알아내는 코드

↓

종속변수가 이진 데이터인 명목형 데이터인 회귀분석

종속변수가 명목형인 경우의 데이터일 때 적합한 분석

```

install.packages('aod')
install.packages('ggplot2')
library(aod)
library(ggplot2)

binary <- read.csv("http://www.karlin.mff.cuni.cz/
~pesta/prednasky/NMFM404/Data/binary.csv")
str(binary)

```

gre 점수와 gpa 점수와 학과 순위가 admit(대학원입학)에 미치는 영향도를 분석하는 로지스틱 회귀분석

admit : 1이 입학

0이 불합격

```

#Logistic Regression Model
install.packages("nnet")
library(nnet)

mymodel <- multinom(admit~.,data = binary)

# 설명 : multinom은 로지스틱 회귀함수이다.

```

```

#mis classification rate
p <- predict(mymodel,binary)
tab <- table(p,binary$admit)

```

```
tab  
1-sum(253,29)/400 #0.295
```

```
# Model Performance Evaluation  
install.packages("ROCR")  
library(ROCR)  
pred <- predict(mymodel,binary,type = "prob") #예측값에 대한 확률을 출력  
head( pred )
```

결과 :

1	2	3	4	5	6
0.18955119	0.31778152	0.71781816	0.14894795	0.09795241	0.37867798

```
hist(pred) # 확률의 분포를 확인
```



```
pred <- prediction(pred,binary$admit)  
pred #roc 커브를 그리기 위한 400개의 데이터 포인트 뽑기  
eval <- performance(pred,"acc") # y축을 정확도로 출력  
eval # 392개의 데이터 포인트 추출  
plot(eval)  
  
abline(h=0.71,v=.45) # 숫자는 코드 작성자가 임의로 지정한 것
```

설명 : h는 수평선, v가 수직선의 지점

```

#Identifying the best cutoff and Accuracy
eval # x축이 cutoff이고 y축이 정확도를 그래프로 시각화하기 위한 392개의 데이터 포인트

slot(eval,"y.values") #392개의 데이터 포인트 출력
max <- which.max(slot(eval,"y.values")[[1]]) # 392개의 데이터 포인트 중 max값에 해당하는
                                              # index 번호 출력

max # 61, 61번째 인덱스 번호에 해당하는 데이터가 가장 큰 값

acc <- slot(eval,"y.values")[[1]][[max]] # y축의 정확도 중에 61번째 해당하는 값 출력
cut <- slot(eval,"x.values")[[1]][[max]] # x축의 cutoff 값을 중에 61번째 해당하는 값 출력
print(c(Accuracy=acc,Cutoff = cut))

```

결과 :

Accuracy	Cutoff
0.7175000	0.4683497

```
#Receiver Operating Characteristic (ROC) curve
```

```

pred <- predict(mymodel,binary,type = "prob")
pred <- prediction(pred,binary$admit)
roc <- performance(pred,"tpr","fpr")
plot(roc,colorize = T,
      main = "ROC Curve",
      ylab = "Sensitivity",
      xlab = "1-Specificity")
abline(a=0,b=1)

```

#AUC

```

auc <- performance(pred,"auc")
auc <- unlist(slot(auc,"y.values"))
round(auc,3)

```

```
legend(0.6,0.2, auc, title = "AUC", cex = .50)
```

■ x축을 fpr로 하고 y축을 tpr 하는 2차원 그래프에 cutoff 지점을 시각화 하는 코드

```

perf <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(perf)
max
tpr <- slot(perf,"y.values")[[1]][[max]]

```

```
fpr <- slot(perf, "x.values")[[1]][[max]]  
print(c(tpr,fpr))  
abline(h=0.37, v=0.028)
```

■ 정확도외에 확인해야 할 다른 성능 척도

정확도
카파통계량
민감도
특이도
정밀도
재현율
AUC
cut off

수제비 책 p4-38

$$F1 \text{ SCORE} : 2 \times \frac{\text{Precision(정밀도)} \times \text{recall (재현율)}}{\text{Precision(정밀도)} + \text{recall (재현율)}}$$

F1 값이 높으면 Precision와 recall 모두 좋은 결과를 보인다.
이 하나의 값으로 2가지 정보를 유추할 수 있다는 장점이 있다.

F1 스코어 값이 작으면 잘못 판단한 False값들에 문제가 있다는 것이다.
FP에 문제가 있는 것인지 FN에 문제가 있는 것인지 확인하는 작업이 필요하다.

■ F1 스코어 출력

1. 직접 계산하는 방법

```
Fmeasure <- 2 * precision * recall / (precision + recall)
```

2. 패키지를 이용하는 방법

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

```
actual_type <- credit_test[, 17]  
predict_type <- credit_result  
positive_value <- 'yes'
```

```
negative_value <- 'no'  
F1_Score(actual_type, predict_type, positive_value)
```

설명 : 0~1사이의 범위를 가짐. 정밀도와 민감도(재현율)을 하나로 합한 성능평가 지표
정밀도와 민감도 양쪽이 모두 클 때 F1값도 큰 값을 가진다.

02/16

2021년 2월 16일 화요일 오전 9:48

■ 복습. 10장에서 배웠던 정확도외에 다른 성능 척도는 무엇인가 ?

답:

1. 카파통계량
2. 민감도
3. 특이도
4. 정밀도
5. 재현율
6. ROC 곡선
7. AUC
8. F1 SCORE

■ 이론설명. 모델 성능개선을 위해 11장에서 소개하고 있는 5가지 내용은 무엇인가 ?

1. k-foldout
2. caret을 이용한 모델 자동튜닝
3. 양상률
4. 배깅
5. 부스팅
6. 랜덤포레스트

■ 이론설명. k-foldout 무엇인가 ?



답 : 데이터 집합을 무작위로 동일한 크기를 갖는 **k개의 부분집합으로 나누고** 그 중 1개 집합을 평가 데이터 (test data)로 하고 **나머지(k-1개)** 집합을 학습 데이터로 선정하여 분석 모형을 평가하는 **기법**이다.

■ k-foldout 실습

설명 :

이 실습의 목적은 k-foldout을 통해서 의사결정트리 C50 패키지의 최적의 하이퍼파라미터를 테스트하기 전에 알아내겠다는 것이다.

이 실습에서는 독일 은행 데이터 1000개를 다 사용해서 k-foldout 테스트를 진행한다.

제대로 실습하려면 1000개에서 테스트 데이터를 따로 분리해놓고 나머지 훈련 데이터로 k-foldout을 진행해야 한다.

예 :

전체 1000개 --> 훈련 900개, 테스트 100개



k-foldout 진행

여기서 k값을 10으로 줬다면 10개의 파티션으로 나누고

10개 중 9개를 훈련데이터로 쓰고 나머지를 테스트 데이터로 사용하면서 k-foldout을 진행한다.

```
setwd("d:WWWdata")
install.packages("caret")
library(caret)
```

```
credit <- read.csv("credit.csv") # 독일 은행의 채무 불이행자를 예측  
nrow(credit)
```

10-fold CV #k값을 10개로 해서 fold를 하겠다.

```
 folds <- createFolds(credit$default, k = 10) # k값을 10으로 지정
```

str(folds) #설명: 전체 10폴드 교차검증을 수행하기 위해 샘플링 된 인덱스가 생성됨

훈련 데이터 내에서 훈련 데이터와 테스트 데이터를 나누는 코드

```
credit01_test <- credit[folds$Fold01, ] #Fold1을 테스트 데이터로 쓰겠다.
```

```
credit01_train <- credit[-folds$Fold01, ] #Fold1을 제외한 나머지 9개를 훈련데이터로 쓰겠다.
```

```
nrow(credit01_test) # 100
```

```
nrow(credit01_train) # 900
```

#전체 10폴드 교차검증을 수행하려면 이 단계는 10회 반복되어야한다.

함수 lapply(데이터, 함수) --> 함수에 데이터를 하나씩 자동으로 입력하는 함수

```
result <- lapply( 1:3, function( x ) x *2 )
```

```
result
```

결과 :

```
[[1]]
```

```
[1] 2
```

```
[[2]]
```

```
[1] 4
```

```
[[3]]
```

```
[1] 6
```

```
## Automating 10-fold CV for a C5.0 Decision Tree using lapply() ---- |
```

```
library(caret)
```

```
library(C50) # 의사결정트리 모델 사용
```

```
library(irr) # 10 foldout을 진행하기 위한 패키지
```

```
credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
```

```
set.seed(123)
```

```
folds <- createFolds(credit$default, k = 10)
```

```
str(folds)
```

```

##### kappa 구하기

cv_results <- lapply(folds, function(x) {
credit_train <- credit[-x, ]
credit_test <- credit[x, ]
credit_model <- C5.0(default ~ ., data = credit_train)
credit_pred <- predict(credit_model, credit_test)
credit_actual <- credit_test$default
kappa <- kappa2(data.frame(credit_actual, credit_pred))$value
return(kappa)

})

str(cv_results)

```

문제1. 위의 결과가 카파지수가 아니라 정확도가 출력되게 하시오.

```

cv_results <- lapply(folds, function(x) {
credit_train <- credit[-x, ]
credit_test <- credit[x, ]
credit_model <- C5.0(default ~ ., data = credit_train)
credit_pred <- predict(credit_model, credit_test)
credit_actual <- credit_test$default

x <- data.frame(credit_actual, credit_pred)
a <- sum(x$credit_actual == x$credit_pred) / length(x$credit_actual)
return(a)

})

str(cv_results)

```

문제2. 위의 정확도의 평균을 구하시오 !

```

> mean(unlist(cv_results))
[1] 0.728

```

문제3. 위의 데이터를 훈련 데이터 75%, 테스트 데이터 25%로 나누고 훈련 데이터 75%에 대해서만 k-foldout을 진행하고 훈련 데이터의 정확도의 평균을 확인하시오 !

힌트코드:

```

library(caret)
in_train <- createDataPartition(credit$default, p = 0.75, list = FALSE)
credit_train <- credit[in_train, ] # 훈련 데이터 구성

```

```
credit_test <- credit[-in_train, ] # 테스트 데이터 구성
```

#####

답 :

```
library(caret)
library(C50)
library(irr)
credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
str(credit)
```

#힌트 코드: 훈련 데이터와 테스트 데이터를 분리하는 코드

여기서 만든 테스트 데이터는 맨 마지막에 한번 테스트할 때 사용할 것이다.

```
library(caret)
in_train <- createDataPartition(credit$default, p = 0.75, list = FALSE)
credit_train <- credit[in_train, ] # 훈련 데이터 구성
credit_test <- credit[-in_train, ] # 테스트 데이터 구성
```

지금부터는 credit_train을 가지고 k-foldout 을 진행할 것이다.

```
set.seed(123)
```

```
folds <- createFolds(credit_train$default, k = 10)

cv_results <- lapply(folds, function(x) {
  credit_train2 <- credit_train[-x, ]
  credit_test2 <- credit_train[x, ]
  credit_model <- C5.0(default ~ ., data = credit_train2)
  credit_pred <- predict(credit_model, credit_test2)
  credit_actual <- credit_test2$default

  x <- data.frame(credit_actual, credit_pred)
  a <- sum(x$credit_actual == x$credit_pred) / length(x$credit_actual)
  return(a)
})
```

```
mean(unlist(cv_results))
```

결과 :

```
> mean(unlist(cv_results))
[1] 0.745404
```

#설명 : trials를 1로 주고 했을 때 훈련 데이터에 대한 10-foldout의 평균 정확도는 0.745404이다.

위와 같이 훈련 데이터를 일부 분리해서 validation 데이터로 사용하는 것은 훈련 데이터를 가지고 최적의 하이퍼 파라미터를 찾아내기 위해서이다.

문제4. 하이퍼 파라미터인 seed 값은 659로 하고 trials는 100으로 해서 다시 훈련 데이터의 정확도 평균을 확인하시오

```
library(caret)
library(C50)
library(irr)

credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
str(credit)

library(caret)

set.seed(659)
in_train <- createDataPartition(credit$default, p = 0.75, list = FALSE)
credit_train <- credit[in_train, ] # 훈련 데이터 구성
credit_test <- credit[-in_train, ] # 테스트 데이터 구성

set.seed(123)
folds <- createFolds(credit_train$default, k = 10)

cv_results <- lapply(folds, function(x) {
  credit_train2 <- credit_train[-x, ]
  credit_test2 <- credit_train[x, ]
  credit_model <- C5.0( default ~., data=credit_train2, trials=100)
  credit_pred <- predict(credit_model, credit_test2)
  credit_actual <- credit_test2$default
  x <- data.frame(credit_actual, credit_pred)
  a <- sum(x$credit_actual==x$credit_pred) / length(x$credit_actual==x$credit_pred)
  return(a)
})

str(cv_results)

mean(unlist(cv_results)) #0.7506999
```

문제5. 위에서 알아낸 seed값과 trials 를 가지고 만든 모델로 테스트 데이터의 정확도를 확인하시오 ~

```
library(caret)
library(C50)
library(irr)
```

```

credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
str(credit)

library(caret)

set.seed(659)
in_train <- createDataPartition(credit$default, p = 0.75, list = FALSE)
credit_train <- credit[in_train, ] # 훈련 데이터 구성
credit_test <- credit[-in_train, ] # 테스트 데이터 구성

set.seed(123)

credit_model <- C5.0( default ~., data=credit_train, trials=100)
credit_pred <- predict(credit_model, credit_test)
credit_actual <- credit_test$default
x <- data.frame(credit_actual, credit_pred)
a <- sum(x$credit_actual==x$credit_pred) / length(x$credit_actual==x$credit_pred)
a #0.74

```

설명 : 훈련 데이터의 정확도가 0.75이고 테스트 데이터의 정확도가 0.74면 아주 작게 오버피팅이 일어난 것이므로 나쁘지 않다.

문제6. [빅데이터 기사 시험] 다음중 k-fold cross validation에 대한 설명으로 가장 부적절한 것은 ? 수제비 p4-39

1. 데이터 집합을 무작위로 k개의 부분집합으로 나누어 검증하는 방법이다.
2. 전체 데이터를 k개의 동일한 크기로 나눈다.
3. 모든 데이터를 학습과 평가에 사용할 수 있다.
4. k값이 증가하면 수행시간과 계산량이 감소한다.

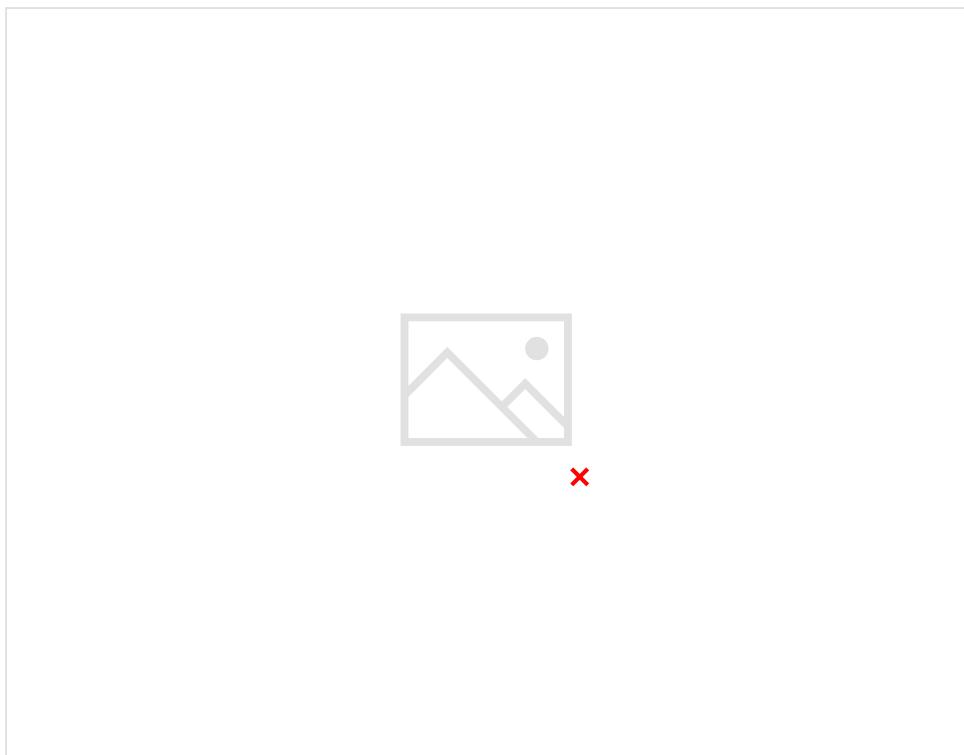
정답 : 4번, 오히려 수행시간과 계산량이 증가한다.

■ 이론설명. caret 패키지의 역할은 무엇입니까 ?

이전에는 머신러닝 모델의 성능을 높이기 위해 우리가 직접 모델의 성능을 높이는 하이퍼파라미터를 직접 알아내야 했다.

그런데 caret을 이용하면 이 패키지가 알아서 최적의 파라미터를 찾아준다.

- 이론설명. 머신러닝 모델별로 최적화해야 할 하이퍼 파라미터는 무엇이 있습니까 ?



의사결정트리의 튜닝은 model, trials, winnow 설정에 대해 3의 3승의 27개의 조합으로 정확도를 각각 계산해서 가장 정확도가 높은 조합을 찾아내야하는데 caret 패키지가 이를 자동으로 찾아준다.

■ 실습1. caret 패키지 사용 실습

의사결정트리 C5.0 패키지의 하이퍼파라미터인 trials, winnow, model의 27개의 조합에 대한 각각의 # 정확도를 구하는 작업

```
library(caret)
library(C50)
library(irr)
```

```
credit <- read.csv("credit.csv", stringsAsFactors=TRUE)
set.seed(300)
```

C5.0 의 하이퍼파라미터인 trials, winnow, model 의 27개의 조합에 대한 # 각각의 정확도를 구하는 작업

```
m <- train( default~. , data=credit, method="C5.0")
m # 튜닝한 결과를 확인할 수 있다.
```

```
p <- predict( m , credit )
table(p, credit$default)
```

점심시간문제(2/16) 아래의 코드는 credit 전체코드를 한번에 사용해서 최적의 하이퍼파라미터를 찾고 credit 전체코드를 한번에 사용해서 평가를 하는 코드입니다.
훈련데이터를 75%로 하고 테스트 데이터를 25%로 나눠서 훈련데이터에 대해서 caret을 이용한 자동튜닝을 진행하시오! 만든 모델의 25%의 테스트 데이터를 평가하세요!

```
library(caret)
library(C50)
library(irr)

credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
str(credit)

library(caret)

set.seed(123)
in_train <- createDataPartition(credit$default, p = 0.75, list = FALSE)
credit_train <- credit[in_train, ] # 훈련 데이터 구성
credit_test <- credit[-in_train, ] # 테스트 데이터 구성
```

```
m <- train( default~ . , data=credit_train, method="C5.0")
m # 튜닝한 결과를 확인할 수 있다.
```

```
p <- predict( m , credit_test )
table(p, credit_test$default)

library(gmodels)
y <- CrossTable(credit_test$default, p)
sum( y$prop.tbl*diag(2)) #0.716
```

■ winnow가 무엇일까?

<https://cafe.daum.net/oracleoracle/Sg0x/888> 참조

<https://www.rdocumentation.org/packages/C50/versions/0.1.3.1/topics/C5.0Control>

<https://analysisbugs.tistory.com/112>

winnow 는 TRUE 또는 FALSE 로 지정할 수 있는데 **Feature Selection**을 할지 말지를 결정하는 파라미터이다.

Feature Selection의 주된 목적은 독립 변수중에서, 중복되거나 종속변수 (Y)와 관련이 없는 변수들을 제거하여, Y를 가장 잘 예측하는 변수들의 조합을 찾아내는 것이기 때문에, 최적화 문제로도 정의할 수 있습니다.

문제_5_1 점심시간 문제 코드를 다시 수행하는데 k-fold의 k값을 디폴트인 25개가 아니라 30개로 변경해서 수행하면 정확도가 더 올라가는지 확인하시오!

```
library(caret)
library(C50)
library(irr)

credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
str(credit)

library(caret)

set.seed(123)
in_train <- createDataPartition(credit$default, p = 0.75, list = FALSE)
credit_train <- credit[in_train, ] # 훈련 데이터 구성
credit_test <- credit[-in_train, ] # 테스트 데이터 구성

ctrl <- trainControl( method="cv", number=30, selectionFunction="oneSE" )
m <- train( default~ . , data=credit_train, method="C5.0", trControl= ctrl)
m # 튜닝한 결과를 확인할 수 있다.

p <- predict( m , credit_test )
table(p, credit_test$default)

library(gmodels)
y <- CrossTable(credit_test$default, p)
sum( y$prop.tbl*diag(2)) #0.732
```

정확도가 73.2로 올라감

문제_5_2 k값을 40개로 변경해서 수행하면 정확도가 더 올라가는지 확인하시오!

```
library(caret)
library(C50)
library(irr)

credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
str(credit)

library(caret)

set.seed(123)
```

```

in_train <- createDataPartition(credit$default, p = 0.75, list = FALSE)
credit_train <- credit[in_train, ] # 훈련 데이터 구성
credit_test <- credit[-in_train, ] # 테스트 데이터 구성

ctrl <- trainControl( method="cv", number=40, selectionFunction="oneSE" )
m <- train( default~ . , data=credit_train, method="C5.0", trControl= ctrl)
m # 튜닝한 결과를 확인할 수 있다.

p <- predict( m , credit_test )
table(p, credit_test$default)

library(gmodels)
y <- CrossTable(credit_test$default, p)
sum( y$prop.tbl*diag(2)) #0.752

```

#정확도가 75.2로 올라갔다.

문제7. 이번에는 데이터를 iris로 바꿔보세요.

```

library(caret)
library(C50)
library(irr)
data(iris)
head(iris)

set.seed(123)
in_train <- createDataPartition(iris$Species, p = 0.75, list = FALSE)
iris_train <- iris[in_train, ] # 훈련 데이터 구성
iris_test <- iris[-in_train, ] # 테스트 데이터 구성

m <- train( Species~ . , data=iris_train, method="C5.0")

m # 튜닝한 결과를 확인할 수 있다.

p <- predict( m , iris_test )
table(p, iris_test$Species)

library(gmodels)
y <- CrossTable(iris_test$Species ,p)
sum(y$prop.tbl * diag(3)) #0.88

```

문제8. 위에서는 의사결정트리의 C5.0 패키지를 이용해서 의사결정트리로 분류하라고 했는데 이번에는 양상을 기법의 랜덤포레스트를 써서 분류하라고 하시오!

랜덤포레스트 + 최적의 파라미터를 자동으로 찾게 하는 방법

```
library(caret)
library(C50)
library(irr)
data(iris)
head(iris)

set.seed(123)
in_train <- createDataPartition(iris$Species, p = 0.75, list = FALSE)
iris_train <- iris[in_train, ] # 훈련 데이터 구성
iris_test <- iris[-in_train, ] # 테스트 데이터 구성

m <- train( Species~ . , data=iris_train, method="rf")
#랜덤포레스트 : 의사결정트리 + 앙상블

m # 튜닝한 결과를 확인할 수 있다.

p <- predict( m , iris_test )
table(p, iris_test$Species)

library(gmodels)
y <- CrossTable(iris_test$Species ,p)
sum(y$prop.tbl * diag(3)) #0.97
```

■ 문제1. 이번에는 폴드수를 10이 아닌 20으로 늘려서 테스트하시오 !

답:

```
ctrl <- trainControl( method="cv", number=20, selectionFunction="oneSE" )

grid <- expand.grid( .model="tree", .trials= c(1, 5, 10, 15, 20, 25, 30, 35), .winnow="FALSE" )

m <- train( default~ . , data=credit,
            method="C5.0",
            metric="Kappa",
            trControl= ctrl,
            tuneGrid= grid )

p <- predict( m, credit ) table( p, credit$default )
```

■ knn의 k 값을 자동으로 알아내는 방법

```
colnames(iris)
levels(iris$Species)

set.seed(1)
train <- sample(1:150, 100) #무작위로 100개 추출 (학습데이터)
train_Set <- iris[train, ] #학습데이터 list형
test_Set <- iris[-train, ] #테스트 데이터 list형
nrow( train_Set ) #100
nrow( test_Set ) #50

## Do 5 repeats of 10-Fold CV for the iris data. We will fit
## a KNN model that evaluates 12 values of k and set the seed ## at each iteration.

set.seed(123) #아래의 코드에서 sample을 사용해서 랜덤으로 생성한 숫자값을 seeds 변수에
               # 담기 때문에 지정한 seed

seeds <- vector(mode = "list", length = 51) #리스트 자료구조를 생성하는데 리스트의 요소를 51개로
                                              # 해서 seeds라는 이름으로 생성함
seeds

for(i in 1:50) seeds[[i]] <- sample.int(1000, 22) # for문을 이용해서 seeds 리스트의 1번부터
                                                 # 50번까지의 요소안에 숫자를 22개씩 랜덤으로 채워넣는다.
seeds

## For the last model:

seeds[[51]] <- sample.int(1000, 1)
seeds

ctrl <- trainControl(method = "repeatedcv", repeats = 5, seeds = seeds)

# 설명 : repeatedcv : 반복된 교차검증을 하겠다.
# repeats = 5 : 반복을 5번하겠다.

set.seed(1)
mod <- train(Species ~ ., data = train_Set, method = "knn", tuneLength = 12, trControl = ctrl)
mod

#결과 :
# Accuracy was used to select the optimal model using the largest value.
# The final value used for the model was k = 13.

test.pred <- predict(mod, newdata = test_Set)
```

```

test.pred
table(test.pred,test_Set$Species)

# 정확도 확인

library(gmodels)
g <- CrossTable( test_Set$Species, test.pred )
x <- sum(g$prop.tbl *diag(3)) # 정확도 확인하는 코드
x #0.96

```

■ 문제2. 이번에는 **method="adaptive_cv"** 로 해서 수행하고 정확도를 확인하시오 !

knn 문제2번 답 :

```

colnames(iris)
levels(iris$Species)

set.seed(1)
train <- sample(1:150, 100) #무작위로 100개 추출 (학습데이터)
train_Set <- iris[train, ] #학습데이터 list형
test_Set <- iris[-train, ] #테스트 데이터 list형

## Do 5 repeats of 10-Fold CV for the iris data. We will fit
## a KNN model that evaluates 12 values of k and set the seed ## at each iteration.

set.seed(123)
seeds <- vector(mode = "list", length = 51)
seeds

for(i in 1:50) seeds[[i]] <- sample.int(1000, 22)
seeds

## For the last model:

seeds[[51]] <- sample.int(1000, 1)
seeds

ctrl <- trainControl( method = "adaptive_cv", repeats=5, seeds= seeds )

set.seed(1)
mod <- train(Species ~ ., data = train_Set, method = "knn", tuneLength = 12, trControl = ctrl)
mod

test.pred <- predict(mod, newdata = test_Set)
test.pred

```

```
table(test.pred,test_Set$Species)
```

정확도 확인

```
library(gmodels)
g <- CrossTable( test_Set$Species, test.pred )
x <- sum(g$prop.tbl *diag(3)) # 정확도 확인하는 코드
x #0.98
```

■ 문제3. 위의 스크립트에서 튜닝되는 과정을 출력하시오 !

```
colnames(iris)
levels(iris$Species)

set.seed(1)
train <- sample(1:150, 100) #무작위로 100개 추출 (학습데이터)
train_Set <- iris[train, ] #학습데이터 list형
test_Set <- iris[-train, ] #테스트 데이터 list형

## Do 5 repeats of 10-Fold CV for the iris data. We will fit
## a KNN model that evaluates 12 values of k and set the seed ## at each iteration.

set.seed(123)
seeds <- vector(mode = "list", length = 51)
seeds

for(i in 1:50) seeds[[i]] <- sample.int(1000, 22)
seeds

## For the last model:

seeds[[51]] <- sample.int(1000, 1)
seeds

ctrl <- trainControl(method = "adaptive_cv", repeats = 5, verboseTer = TRUE , seeds = seeds)

설명 : verbose + !(대문자)ter이다

set.seed(1)
mod <- train(Species ~ ., data = train_Set, method = "knn", tuneLength = 12, trControl = ctrl)
mod

test.pred <- predict(mod, newdata = test_Set)
test.pred

table(test.pred,test_Set$Species)
```

정확도 확인

```
library(gmodels)
g <- CrossTable( test_Set$Species, test.pred )
x <- sum(g$prop.tbl *diag(3)) # 정확도 확인하는 코드
x
```



■ 복습. 10장에서 배웠던 정확도외에 다른 성능 척도는 무엇인가 ?

답:

1. 카파통계량
2. 민감도
3. 특이도
4. 정밀도
5. 재현율
6. ROC 곡선
7. AUC
8. F1 SCORE

■ 이론설명. 모델 성능개선을 위해 11장에서 소개하고 있는 5가지 내용은 무엇인가 ?

답:

■ 이론설명. caret 패키지의 역할은 무엇입니까 ?

■ 이론설명. 머신러닝 모델별로 최적화해야할 하이퍼 파라미터는 무엇이 있습니까 ?

의사결정트리의 튜닝은 model, trials, winnow 설정에 대해 3의 3승의 27개의 조합으로 정확도를 각각 계산해서 가장 정확도가 높은 조합을 찾아내야하는데 caret 패키지가 이를 자동으로 찾아준다.

■ 실습1. caret 패키지 사용 실습

```
library(caret)
```

```
library(C50)
```

```
library(irr)
```

```
credit <- read.csv("credit.csv", stringsAsFactors=TRUE)  
set.seed(300)
```

C5.0 의 하이퍼파라미터인 trials, winnow, model 의 27개의 조합에 대한 # 각각의 정확도를 구하는 작업

```
m <- train( default~. , data=credit, method="C5.0")  
m # 튜닝한 결과를 확인할 수 있다.
```

```
p <- predict( m , credit )  
table(p, credit$default)
```

■ winnow 가 무엇인지 ?

winnow 는 TRUE 또는 FALSE 로 지정할 수 있는데 Feature Selection을 할지 말지를 결정하는 파라미터이다.

Feature Selection의 주된 목적은 독립 변수중에서, 중복되거나 종속변수 (Y)와 관련이 없는 변수들을 제거하여,

Y를 가장 잘 예측하는 변수들의 조합을 찾아내는 것이기 때문에, 최적화 문제로도 정의할 수 있습니다.

■ 이론설명. 튜닝절차 커스터 마이징 하기

튜닝절차를 customizing 해서 더 성능을 높이도록 설정할 수 있다.

예를들면 방금 수행한 튜닝은 앞에서 배웠던 k-폴드교차검증의 k 값을 25개를 사용하는 방법이었습니다.

옵션을 주어서 튜닝할 때 사용할 수 있게 설정을 할 수 있다.

1. 기존 방법:

```
m <- train( default~ . , data=credit, method="C5.0")
```

2. 커스터마이징 방법 :

```
ctrl <- trainControl( method="cv", number=30,  
                      selectionFunction="oneSE" )
```

※ method="cv" --> k 폴드 교차검증 하겠다.

number=30 --> 폴드수

oneSE --> 다양한 후보중에서 최적의 모델을 선택하는 방법중 하나인데 이 방법이 3가지가 있다.

best, oneSE, tolerance 세가지가 있다.

best --> 후보중에 단순히 성능척도값중에 최고값을 갖는 후보를 선택(default)

oneSE --> 최고성능의 1표준오차내의 가장 단순한 후보를 선택한다.

tolerance--> 사용자 지정 비율내에 가장 단순한 후보를 선택한다.

```
m <- train( default~ . , data=credit, method="C5.0",  
            metric="Kappa",  
            trControl= ctrl )
```

문제_5_2. 점심시간 문제의 코드를 다시 수행하는데 k-fold 의 k 값을 디폴트인 25개가 아니라 30개로 변경해서 수행하면 정확도가 더 올라가는지 확인하시오!

```
library(caret)  
library(C50)  
library(irr)  
credit <- read.csv("credit.csv", stringsAsFactors = TRUE)  
str(credit)  
library(caret)  
  
set.seed(123)  
in_train <- createDataPartition(credit$default, p = 0.75, list = FALSE)  
credit_train <- credit[in_train, ] # 훈련 데이터 구성  
credit_test <- credit[-in_train, ] # 테스트 데이터 구성  
  
ctrl <- trainControl( method="cv", number=30, selectionFunction="oneSE" )  
m <- train( default~ . , data=credit_train, method="C5.0", trControl= ctrl )  
m # 튜닝한 결과를 확인할 수 있다.  
  
p <- predict( m , credit_test )
```

```

table(p, credit_test$default)

library(gmodels)
y <- CrossTable(credit_test$default ,p)
sum(y$prop.tbl * diag(2))

문제6. k 값을 40으로 하면 정확도가 더 올라가는지 확인하시오 !

```

```

library(caret)
library(C50)
library(irr)
credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
str(credit)
library(caret)

set.seed(123)
in_train <- createDataPartition(credit$default, p = 0.75, list = FALSE)
credit_train <- credit[in_train, ] # 훈련 데이터 구성
credit_test <- credit[-in_train, ] # 테스트 데이터 구성

ctrl <- trainControl( method="cv", number=40, selectionFunction="oneSE" )

m <- train( default~ . , data=credit_train, method="C5.0", trControl= ctrl)

m # 튜닝한 결과를 확인할 수 있다.

p <- predict( m , credit_test )
table(p, credit_test$default)

library(gmodels)
y <- CrossTable(credit_test$default ,p)
sum(y$prop.tbl * diag(2))

```

※ trials 의 갯수도 아래와 같이 8개로 제한할 수 있다.

```

grid <- expand.grid( .model="tree",
                      .trials= c(1, 5, 10, 15, 20, 25, 30, 35),
                      .winnow="FALSE" )

m <- train( default~ . , data=credit, method="C5.0",
            metric="Kappa",
            trControl= ctrl,
            truneGrid= grid )

문제7. 이번에는 데이터를 iris 데이터로 구현하시오 !

```

```

library(caret)
library(C50)
library(irr)
data(iris)
head(iris)

set.seed(123)
in_train <- createDataPartition(iris$Species, p = 0.75, list = FALSE)
iris_train <- iris[in_train, ] # 훈련 데이터 구성
iris_test <- iris[-in_train, ] # 테스트 데이터 구성

m <- train( Species~ . , data=iris_train, method="C5.0" )

m # 튜닝한 결과를 확인할 수 있다.

p <- predict( m , iris_test )
table(p, iris_test$Species)

library(gmodels)
y <- CrossTable(iris_test$Species ,p)
sum(y$prop.tbl * diag(3))

```

문제8. 위에서는 의사결정트리의 C5.0 패키지를 이용해서 의사결정트리로 분류하라고 했는데

이번에는 양상을 기법의 랜덤포레스트를 써서 분류하라고 하시오 !

랜덤포레스트 + 최적의 파라미터를 자동으로 찾게하는 방법

```

library(caret)
library(C50)
library(irr)
data(iris)
head(iris)

set.seed(123)
in_train <- createDataPartition(iris$Species, p = 0.75, list = FALSE)
iris_train <- iris[in_train, ] # 훈련 데이터 구성
iris_test <- iris[-in_train, ] # 테스트 데이터 구성

m <- train( Species~ . , data=iris_train, method="rf" )

# 랜덤포레스트: 의사결정트리 + 양상을 기법

```

```
m # 튜닝한 결과를 확인할 수 있다.
```

```
p <- predict( m , iris_test )
table(p, iris_test$Species)

library(gmodels)
y <- CrossTable(iris_test$Species ,p)
sum(y$prop.tbl * diag(3))
```

■ knn의 k 값을 자동으로 알아내는 방법

```
colnames(iris)
levels(iris$Species)

set.seed(1)
train <- sample(1:150, 100) #무작위로 100개 추출 (학습데이터)
train_Set <- iris[train, ] #학습데이터 list형
test_Set <- iris[-train, ] #테스트 데이터 list형
nrow(train_Set) # 100
nrow(test_Set) # 50

## Do 5 repeats of 10-Fold CV for the iris data. We will fit
## a KNN model that evaluates 12 values of k and set the seed ## at each iteration.

set.seed(123) # 아래의 코드에서 sample 을 사용해서 랜덤으로 생성한 숫자값을 seeds 변수에
# 담기 때문에 지정한 seed

seeds <- vector(mode = "list", length = 51) # 리스트 자료구조를 생성하는데 리스트의 요소
seeds # 를 51개로 해서 seeds라는 이름으로 생성함

for(i in 1:50) seeds[[i]] <- sample.int(1000, 22) # for 문을 이용해서 seeds 리스트의 1번부터
seeds # 50번까지의 요소안에 숫자를 22개씩 랜덤으로 채워넣는다.

## For the last model:

seeds[[51]] <- sample.int(1000, 1)
seeds

ctrl <- trainControl(method = "repeatedcv", repeats = 5, seeds = seeds)

set.seed(1)
mod <- train(Species ~ ., data = train_Set, method = "knn", tuneLength = 12, trControl = ctrl)
mod

test.pred <- predict(mod, newdata = test_Set)
```

```

test.pred
table(test.pred,test_Set$Species)

# 정확도 확인

library(gmodels)
g <- CrossTable( test_Set$Species, test.pred )
x <- sum(g$prop.tbl *diag(3)) # 정확도 확인하는 코드
x

```

TrinControl 함수 메뉴얼:

<https://www.rdocumentation.org/packages/caret/versions/6.0-86/topics/trainControl>

문제2. 이번에는 method="adaptive_cv" 로 해서 수행하고 정확도를 확인하시오 !

답:

```

colnames(iris)
levels(iris$Species)

set.seed(1)
train <- sample(1:150, 100) #무작위로 100개 추출 (학습데이터)
train_Set <- iris[train, ] #학습데이터 list형
test_Set <- iris[-train, ] #테스트 데이터 list형

## Do 5 repeats of 10-Fold CV for the iris data. We will fit
## a KNN model that evaluates 12 values of k and set the seed ## at each iteration.

set.seed(123)
seeds <- vector(mode = "list", length = 51)
seeds

for(i in 1:50) seeds[[i]] <- sample.int(1000, 22)
seeds

## For the last model:

seeds[[51]] <- sample.int(1000, 1)
seeds

ctrl <- ?

```

```

set.seed(1)
mod <- train(Species ~ ., data = train_Set, method = "knn", tuneLength = 12, trControl = ctrl)
mod

test.pred <- predict(mod, newdata = test_Set)
test.pred

table(test.pred,test_Set$Species)

# 정확도 확인

library(gmodels)
g <- CrossTable( test_Set$Species, test.pred )
x <- sum(g$prop.tbl *diag(3)) # 정확도 확인하는 코드
x

```

문제3. 위의 스크립트에서 튜닝되는 과정을 출력하시오 !

답:

```

colnames(iris)
levels(iris$Species)

set.seed(1)
train <- sample(1:150, 100) #무작위로 100개 추출 (학습데이터)
train_Set <- iris[train, ] #학습데이터 list형
test_Set <- iris[-train, ] #테스트 데이터 list형

## Do 5 repeats of 10-Fold CV for the iris data. We will fit
## a KNN model that evaluates 12 values of k and set the seed ## at each iteration.

set.seed(123)
seeds <- vector(mode = "list", length = 51)
seeds

for(i in 1:50) seeds[[i]] <- sample.int(1000, 22)
seeds

## For the last model:

seeds[[51]] <- sample.int(1000, 1)
seeds

ctrl <- trainControl(method = "adaptive_cv",
                      repeats = 5,
                      ?
                      ,
                      seeds = seeds)

```

```

set.seed(1)
mod <- train(Species ~ ., data = train_Set, method = "knn", tuneLength = 12, trControl = ctrl)
mod

test.pred <- predict(mod, newdata = test_Set)
test.pred

table(test.pred,test_Set$Species)

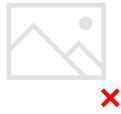
# 정확도 확인

library(gmodels)
g <- CrossTable( test_Set$Species, test.pred )
x <- sum(g$prop.tbl *diag(3)) # 정확도 확인하는 코드
x

문제4. 위의 knn 모델에서 k=11 과 함께 최적의 seed 값이 무엇인가 ?

```

■ 이론설명2. 양상블 기법은 무엇입니까 ?



답 : 다양한 전문가 팀을 만드는 것과 유사한 원리를 활용하는 메타 학습방법을 양상블이라고 한다. 모든 양상블 방법은 약한 학습자 여럿개를 결합하면 강한 학습자가 만들어 진다는 아이디어를 기반으로 한다. 양상블 모형은 여러개의 분류 모형을 같이 사용하여 한꺼번에 평가하는 모형을 말한다.

■ 실습1. 정확도가 60% 밖에 되지 않는 분류기 모형들이 즐비한데 이 모형들을 최소한 몇개를 써야 정확도를 90% 를 능가하게 만들수 있을까 ?

```
ret_err <- function(n,err) {  
  sum <- 0  
  
  for(i in floor(n/2):n) {  
    sum <- sum + choose(n,i) * err^i * (1-err)^(n-i)  
  }  
  sum  
}  
for(j in 1:60) {  
  err <- ret_err(j , 0.4)  
  cat(j,'--->',1-err,'₩n')  
  if(1-err >= 0.9) break  
}
```

■ 이론설명3. 양상블 기법의 원리가 어떻게 되는가 ?



답 : 여러 모델을 이용하여 데이터를 학습하고 모든 모델의 예측 결과를 평균하여 예측한다.

■ 이론설명4. 양상블을 이용했을 때의 장점은 무엇인가 ?



답 :

1. 다양한 모델의 결과를 종합하여 전반적으로 오류를 줄여주어 정확도를 높여준다.
2. 모델별로 다양한 bias(실제값과 예측값과의 차이)를 종합하여 결과를 생성하게 되어 오버피팅을 줄여준다.

오버피팅 : 훈련data는 잘 맞추는데 test data를 잘 못 맞춘다.

■ 이론설명5. 양상불을 이용해서 나온 좋은 모형의 모습은 무엇입니까?



답 :

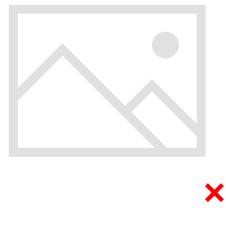
1. 학습 모형의 예측 오류는 크게 두가지, bias와 variance 이다.
2. bias는 예측값과 정답과의 거리가 얼마나 떨어져 있는지 이다.
3. variance는 학습한 모델별로 애측한 값들의 차이이다.
4. 양상불을 이용한 좋은 모형이란 bias를 낮추고 variance를 최소화 하는 모형이다.

■ 이론설명6. 그럼 양상불을 이용해서 오류를 줄이고 좋은 모형이 나오도록 한 양상불의 종류에는 무엇이 있습니까?

답 :

1. **Bagging**: generate weak models (decision trees) from random samples and aggregate them simply
2. **Boosting**: sequentially generate weak models from previous residuals & combine them with different weights
3. **Random Forest**: inject more randomness compared with Bagging

■ 이론설명7. 배깅(bagging)이 무엇입니까 ?



답 :

동일한 모델을 사용하고 데이터만 분할하여 여러개의 모델을 학습하는 양상을 기법을 말한다.
위의 그림에서 데이터를 샘플링할 때는 복원추출한다.

■ 이론설명8. 배깅(bagging) 은 어떻게 예측의 정확도를 높입니까 ?



위의 그림에서는 model1 하나로만 보면 하나의 bag으로만 학습된 모델이다.

각 모델별로 보면 학습 데이터에 오버피팅되어 테스트 데이터로 검증하면 예측성능이 낮다. 배깅은 이렇게 여러 weak model을 여러 개 결합하여 전체적으로 높은 variance를 낮은 variance로 만들면서 예측 성능을 향상 시킨다.

위의 그림을 보면 보델끼리 서로 보안하고 양보하면서 예측한다.

■ 실습1. 독일 은행 데이터로 채무 불이행자를 예측하는 배깅 실습

#1. 데이터를 로드한다.

```
credit <- read.csv("credit.csv", stringsAsFactor=TRUE)
str(credit)
```

#2. 데이터에 각 컬럼들을 이해한다.

#라벨 컬럼 : default ---> yes : 대출금 상환 안함
#no : 대출금 상환

```
prop.table( table(credit$default) )
summary( credit$amount)
```

#3. 데이터가 명목형 데이터인지 확인해본다.

```
str(credit)
```

#4. 데이터를 shuffle 시킨다.

```
set.seed(31)
credit_shuffle <- credit[ sample( nrow(credit) ), ]
```

#5. 데이터를 9 대 1로 나눈다.

```
train_num <- round( 0.9 * nrow(credit_shuffle), 0)

credit_train <- credit_shuffle[1:train_num , ]

credit_test <- credit_shuffle[(train_num+1) : nrow(credit_shuffle), ]

nrow(credit_test) #100
nrow(credit_train) #900
```

배깅으로 성능 높이기

```
#install.packages("ipred")
library(ipred)
set.seed(300)

mybag <- bagging( default ~ . , data=credit_train, nbagg=25)
```

#※ 설명: nbagg=25 은 양상블에 사용되는 bag의 갯수를 25개

mybag

```
credit_pred <- predict( mybag, credit_test[ , -17] )
credit_pred

table( credit_pred, credit_test$default )
prop.table( table( credit_pred==credit_test$default) )
```

#■ 정확도

```
g <- CrossTable( credit_test$default, credit_pred )

x <- sum(g$prop.tbl *diag(2)) # 정확도 확인하는 코드
x # 0.69
```

문제1. (오늘의 마지막문제 2/16) bag 의 갯수를 50개로 늘리고 확인해 봅니다.

#1. 데이터를 로드한다.

```
credit <- read.csv("credit.csv", stringsAsFactor=TRUE)
str(credit)
```

#2. 데이터에 각 컬럼들을 이해한다.

#라벨 컬럼 : default ---> yes : 대출금 상환 안함
#no : 대출금 상환

```
prop.table( table(credit$default) )
summary( credit$amount)
```

#3. 데이터가 명목형 데이터인지 확인해본다.

```
str(credit)
```

#4. 데이터를 shuffle 시킨다.

```
set.seed(31)
credit_shuffle <- credit[ sample( nrow(credit) ), ]
```

#5. 데이터를 9 대 1로 나눈다.

```
train_num <- round( 0.9 * nrow(credit_shuffle), 0)

credit_train <- credit_shuffle[1:train_num , ]

credit_test <- credit_shuffle[(train_num+1) : nrow(credit_shuffle), ]
nrow(credit_test)
nrow(credit_train)

#install.packages("ipred")
library(ipred)
set.seed(300)

mybag <- bagging( default ~ . , data=credit_train, nbagg=50)

mybag

credit_pred <- predict( mybag, credit_test[ , -17] )
credit_pred

table( credit_pred, credit_test$default )
prop.table( table( credit_pred==credit_test$default) )
```

#■ 정확도

```
g <- CrossTable( credit_test$default, credit_pred )

x <- sum(g$prop.tbl *diag(2)) # 정확도 확인하는 코드
x #0.71
```

R을 활용한 머신러닝

쉬움주의

부스팅

■ 복습. 모델 성능개선을 위해 11장에서 소개하고 있는 5가지 내용은 ?

1. k-fold 교차검증방법 : 문제집, 시험문제(수능)
모의고사를 보면서 실력을 가늠하는 것
2. caret 패키지를 이용한 자동 튜닝 : 최적의 하이퍼파라미터를 자동으로 머신러닝이 알아냄
(의사결정트리, knn)
3. 양상블 - 배깅 : 정확도 ↑, 오버피팅 ↓ / 약한 학습자들을 모아서 강한 학습자를 만듦
4. 양상블 - 부스팅
5. 양상블 - 랜덤포레스트

배깅은 각각에 똑은 가중치를 준다. 그러나 부스팅은 틀린것에 대해서 가중치를 더 준다.

■ 이론설명2. 부스팅이 무엇입니까 ?

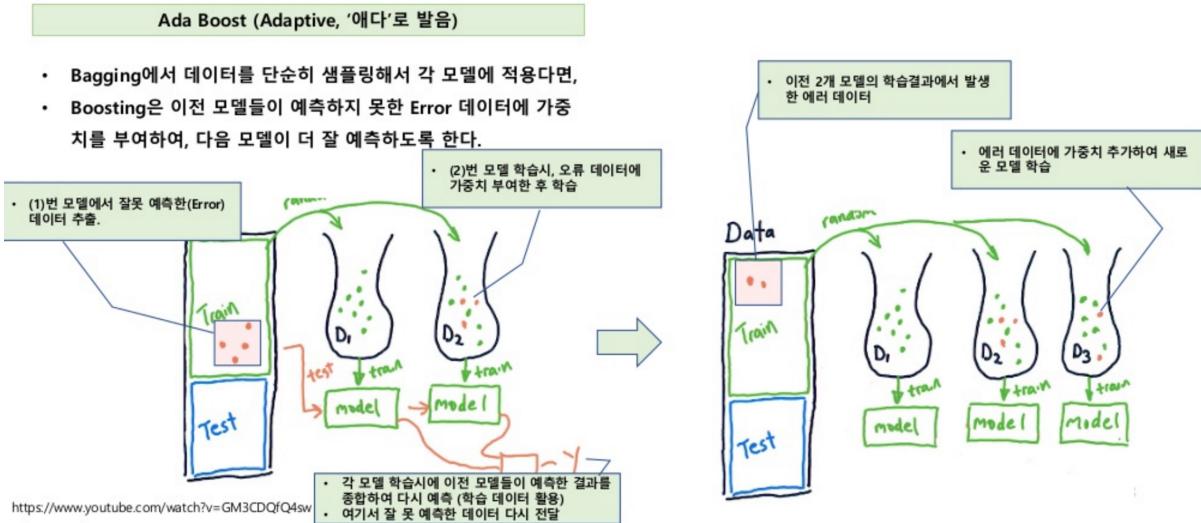
3. Boosting

Clip slide

3. Boosting

Clip slide

Bagging의 변형으로, 모델이 잘 예측하지 못하는 부분을 개선하기 위한 모델



답 :

boost 의 뜻? 격려하다. 향상시키다.

부스팅은 배깅을 약간 개선시키는 알고리즘이데 샘플링하는 과정에서 복원추출할 때 동일한 확률로 하는게 아니라 추출할 때마다 확률을 서로 다르게 개선시키는 방법을 쓴다. 처음에는 모두 동일한 확률로 복원추출하지만 다음번 과정에서는 오분류된 데이터를 추출확률이 더 높도록 조정하고 올바르게 분류된 데이터는 추출확률을 낮게 조정하여 복원추출한다.

이런 작업을 정해진 단계의 수 만큼 반복적으로 사용한다.

■ 이론설명3. 배깅과 부스팅의 차이점이 무엇입니까 ?



✗

답 :

배깅과 부스팅은 모두 의사결정나무의 안정성을 높인다는 공통점이 있다. 둘 다 표본 추출에 있어서 데이터셋에서 복원 랜덤추출하지만 부스팅은 가중치를 사용한다는 차이가 있다. 또한 배깅은 병렬적으로 모델을 만들지만 부스팅은 하나의 모델을 만들어 그 결과로 다른 모델을 만들어 나가는 순차적 모델을 완성시켜 나간다.

가중치에 대해서도 배깅은 $1/n$ 으로 가중치를 주지만 부스팅은 오차가 큰 객체에 대해 더 높은 가중치를 부여한다.

■ 실습1. 독일 데이터로 부스팅 실습

#1. 데이터를 로드한다.

```
credit <- read.csv("credit.csv", stringsAsFactor=TRUE)
str(credit)
```

#2. 데이터에 각 컬럼들을 이해한다.

```
#라벨 컬럼 : default ---> yes : 대출금 상환 안함
```

```
#no : 대출금 상환
```

```
prop.table( table(credit$default) )  
summary( credit$amount)
```

#3. 데이터가 명목형 데이터인지 확인해본다.

```
str(credit)
```

#4. 데이터를 shuffle 시킨다.

```
set.seed(31)  
credit_shuffle <- credit[ sample( nrow(credit) ), ]
```

#5. 데이터를 9 대 1로 나눈다.

```
train_num <- round( 0.9 * nrow(credit_shuffle), 0)  
credit_train <- credit_shuffle[1:train_num , ]  
credit_test <- credit_shuffle[(train_num+1) : nrow(credit_shuffle), ]  
nrow(credit_test) #900  
nrow(credit_train) #100
```

#6. 부스팅으로 성능 높이기

```
install.packages("adabag")  
library(adabag)  
set.seed(300) #boosting이 데이터를 복원추출하기 때문에 필요  
m_adaboost <- boosting( default ~ . , data=credit_train )  
p_adaboost <- predict( m_adaboost, credit_test )  
head(p_adaboost$class)  
p_adaboost$confusion  
table( p_adaboost$class, credit_test$default)
```

#7. 정확도 확인

```
library(gmodels)  
g <- CrossTable( credit_test$default, p_adaboost$class )  
  
x <- sum(g$prop.tbl *diag(2)) # 정확도 확인하는 코드  
x #0.76
```

문제1. iris 데이터셋을 boosting을 이용해서 분류하시오!

#1. 데이터를 로드한다.

```
data(iris)  
str(iris)
```

#2. 데이터에 각 컬럼들을 이해한다.

```
prop.table( table(iris$Species) )  
summary( iris$Species)
```

#3. 데이터가 명목형 데이터인지 확인해본다.

```
str(iris)
```

#4. 데이터를 shuffle 시킨다.

```
set.seed(31)  
iris_shuffle <- iris[ sample( nrow(iris) ), ]
```

#5. 데이터를 8 대 2로 나눈다.

```
train_num <- round( 0.8 * nrow(iris_shuffle), 0)  
iris_train <- iris_shuffle[1:train_num , ]  
iris_test <- iris_shuffle[(train_num+1) : nrow(iris_shuffle), ]  
nrow(iris_test) #15  
nrow(iris_train) #135
```

#6. 부스팅으로 성능 높이기

```
install.packages("adabag")  
library(adabag)  
set.seed(300) #boosting이 데이터를 복원추출하기 때문에 필요  
m_adaboost <- boosting( Species ~ . , data=iris_train )  
p_adaboost <- predict( m_adaboost, iris_test )  
head(p_adaboost$class)  
p_adaboost$confusion  
table( p_adaboost$class, iris_test$Species)
```

#7. 정확도 확인

```
library(gmodels)  
g <- CrossTable( iris_test$Species, p_adaboost$class )  
  
x <- sum(g$prop.tbl *diag(3)) # 정확도 확인하는 코드  
x #1
```

문제2. 위의 boosting 모델의 성능을 높이시오!

```
m_adaboost <- boosting( Species ~ . , data=iris_train, boos=TRUE, mfinal=3 )
```

문제3. 위의 옵션인 **boos=TRUE**와 **mfinal=3**은 무엇인가?

?boosting

boos ? 학습이 반복될 때마다 가중치를 부여할지 말지를 결정하는 옵션

if TRUE (by default), a bootstrap sample of the training set is drawn using the weights for each observation on that iteration. If FALSE, every observation is used with its weights.

mfinal? 학습 반복횟수

an integer, the number of iterations for which boosting is run or the number of trees to use. Defaults to mfinal=100 iterations.



■ 이론설명1. random forest 가 무엇인가요 ?

의사결정 트리에 양상불을 입힘

단점 : 오버피팅 ↑



답 :

의사결정트리는 오버피팅 될 가능성이 높다는 약점을 가지고 있다. 가지치기를 통해 최대 높이를 조정해서 오버피팅될 가능성을 낮출수는 있지만 이것만으로는 오버피팅 문제를 충분히 해결할 수 없다. 그래서 의사결정트리에 양상을 기법을 추가해서 훈련 데이터를 여러 의사결정트리 모델들이 샘플링하여 학습하고 예측결과를 투표하여 가장 많이 득표한 결과를 최종 분류의 결과로 선택한다.

■ 이론설명2. random forest 는 어떻게 수행되어지나요 ?



답 :

랜덤포레스트는 제일 먼저 bagging이라는 과정을 거친다. bagging은 트리를 만들어 training set의 부분집합을 활용하여 형성하는 것을 말한다. 모든 트리는 각기 다른 데이터를 바탕으로 형성이 되며 예측 결과는 최종 투표 결과로 산출된다.

■ 이론설명3. random forest 의 하이퍼 파라미터는 무엇입니까 ?





답 :

1. ntree → 몇개의 나무를 생성할지 설정하는 인자
2. mtry → 각 노드에서 랜덤하게 고려될 변수의 개수를 지정
3. node size → 나무의 깊이를 설정하는 인자로 최소한의 노드의 개수를 뜻한다.
 이 값이 크면 깊이가 얕은 나무가 생성되고 이 값이 작으면 얕은 나무가 생성된다.

■ 11장. 양상을 random forest 실습1

krphosis 데이터 설명 :

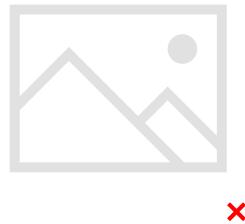
<https://medium.com/data-py-blog/kyphosis-disease-classification-fe275f05dbb5>

랜덤포레스트 패키지 설명 :

<https://www.rdocumentation.org/packages/randomForest/versions/4.6-14/topics/randomForest>

#rattle 패키지로 시작화 한 것

#start : 척추수



0. 필요한 패키지를 다운로드 한다.

```
install.packages("rpart")
install.packages("rattle")
install.packages("randomForest")
```

1. 데이터를 로드한다.

```
kyphosis <- read.csv("kyphosis.csv", stringsAsFactors = TRUE)
head(kyphosis)
```

2. rpart 를 이용해서 의사 결정트리 모델을 생성한다.

```
library(rpart)
fit <- rpart(kyphosis ~ age + number + start, method="class", data=kyphosis)
```

3. 모델을 시각화 한다.

```
library(rattle) #시각화하는 패키지
library(rpart.plot)

fancyRpartPlot(fit)
```

4. 정확도를 확인한다.

```
result <- predict(fit , newdata = kyphosis)
sum(kyphosis$kyphosis == ifelse(result[,1]>0.5 , "absent" , "present"))/NROW(kyphosis)
```

정확도 : [1] 0.8395062

그럼 83% 로 나온다.

이 수치를 높이기 위해서 random forest 를 사용한다.

1. 랜덤 포레스트 모델을 만든다.

```
library(randomForest)
```

```
fit <- randomForest(kyphosis ~ age + number + start,  data=kyphosis)
```

```
res2 <- predict(fit , newdata = kyphosis)
```

```
res2
```

```
sum(res2 == kyphosis$kyphosis)/NROW(kyphosis)
```

결과 : [1] 0.9876543 <--- 98% 로 정확도가 올라간다.

위에서 500개의 트리가 나오는데 이 트리들은 다 똑같은 트리는 아니고 우리는 모르게 미세하게 조금씩 파라미터가 다르다.

그 값들도 자동으로 함수에서 알아서 조정해준다.

```
summary(fit)
```

문제1. (점심시간 문제) 척추 데이터의 랜덤 포레스트 모델을 다시 생성하는데

mtry의 개수를 늘려서 테스트 하시오!

랜덤 포레스트 모델을 만든다.

```
library(randomForest)
```

```
fit <- randomForest(kyphosis ~ age + number + start, data=kyphosis, mtry=3)
```

```
res2 <- predict(fit , newdata = kyphosis)
```

```
res2
```

```
sum(res2 == kyphosis$kyphosis)/NROW(kyphosis)
```

결과 : [1] 1 <--- 100% 로 정확도가 올라간다.

위에서 500개의 트리가 나오는데 이 트리들은 다 똑같은 트리는 아니고 우리는 모르게 미세하게 조금씩 파라미터가 다르다.

그 값들도 자동으로 함수에서 알아서 조정해준다.

```
str(fit) #하면 mtry값이 변경된 것을 확인할 수 있다.
```

■ 11장. 랜덤포레스트 실습2 (iris데이터)

#필요한 패키지 다운로드

```
#install.packages('randomForest')
library(randomForest)
```

#0. shuffle을 먼저 한다.

```
set.seed(123)
iris_shuffle <- sample(1:150, 150)
iris_shuffle
iris2 <- iris[iris_shuffle, ]
iris2
```

1. 훈련 데이터와 테스트 데이터 분리

```
set.seed(123)
in_train <- createDataPartition(iris2$Species, p = 0.75, list = FALSE)
iris_train <- iris2[in_train, ] # 훈련 데이터 구성
iris_test <- iris2[-in_train, ] # 테스트 데이터 구성
nrow(iris_train) #114
nrow(iris_test) #36

prop.table(table(iris_train$Species))
prop.table(table(iris_test$Species))
```

#2. 모델 훈련

```
forest_m <- randomForest(Species ~ ., data=iris_train)
forest_m
```

```
forest_m$predicted # 학습된 모델을 통한 train data 의 예측값 확인
length(forest_m$predicted) # 114
```

```
forest_m$importance # 각 feature importance(각 불순도 기반 설명변수 중요도)
forest_m$mtry      # 모델의 mtry 값 확인
forest_m$ntree     # 모델의 ntree 값 확인
```

3. 모델을 통한 예측

```
new_data <- iris_train[10,-5] + 0.2
new_data
```

```
predict(forest_m, newdata = new_data, type = 'class') # 500개 트리의 다중투표 결과  
iris_train[10,'Species']
```

4. 모델 평가

4-1) test data에 대한 score 확인

```
prd_v <- predict(forest_m, newdata = iris_test, type = 'class')  
sum(prd_v == iris_test$Species) / nrow(iris_test) * 100
```

4-2) train data에 대한 score 확인

```
prd_v2 <- predict(forest_m, newdata = iris_train, type = 'class')  
sum(prd_v2 == iris_train$Species) / nrow(iris_train) * 100
```

5. 모델 시각화

```
layout(matrix(c(1,2),nrow=1),width=c(4,1))  
par(mar=c(5,4,4,0)) # 오른쪽 마진 제거  
plot(forest_m)  
par(mar=c(5,0,4,2)) # 왼쪽 마진 제거  
plot(c(0,1),type="n", axes=F, xlab="", ylab="")  
legend("top", colnames(forest_m$err.rate),col=1:4,cex=0.8,fill=1:4)
```

문제. 아래의 사이트를 참고해서 랜덤포레스트의 최적의 파라미터를 자동으로 찾는 loop문을 구현 하시오!

(사이트 맨 아래에 코드가 나옴) <https://hongsamm.tistory.com/20>

```
ntree <- c(600, 700, 800)  
mtry <- c(2:4) #독립변수의 개수보다 작거나 같게 줘야 함  
  
param <- data.frame(n=ntree, m=mtry)  
param  
  
for (i in param$n) {  
  #cat('ntree=', i, '\n')  
  for(j in param$m) {  
    cat('\n')  
    cat('ntree=', i, ' ', 'mtry=', j, '\n')  
    model_iris <- randomForest(Species ~ ., data=iris_train, ntree=i, mtry=j,  
                                na.action=na.omit)  
    print(model_iris)  
  }  
}
```

■ 11장. 양상블과 자동 튜닝을 결합하여 최적의 모델찾기 실험

```
library(caret)
library(C50)
library(irr)
data(iris)
head(iris)
```

#0.shuffle 을 먼저 합니다.

```
set.seed(123)
iris_shuffle <- sample(1:150, 150)
iris_shuffle
iris2 <- iris[iris_shuffle,]
iris2
```

1. 훈련데이터 75%, 테스트 데이터 25%로 분리한다.

```
set.seed(123)
in_train <- createDataPartition(iris2$Species, p = 0.75, list = FALSE)
iris_train <- iris2[in_train, ] # 훈련 데이터 구성
iris_test <- iris2[-in_train, ] # 테스트 데이터 구성
```

2. 랜덤 포레스트를 이용해서 훈련을 시키는데 자동 파라미터 튜닝도 같이 진행함

```
m <- train( Species~ . , data=iris_train, method="rf")
```

```
str(m) #ntree 500개, mtry 2개
```

랜덤포레스트: 의사결정트리 + 앙상블 기법

```
m # 튜닝한 결과를 확인할 수 있다.
```

```
p <- predict( m , iris_test )
table(p, iris_test$Species)
```

```
library(gmodels)
y <- CrossTable(iris_test$Species ,p)
sum(y$prop.tbl * diag(3)) #0.94
```



■ 이론설명1. 서포트 벡터 머신이란 무엇인가요 ?

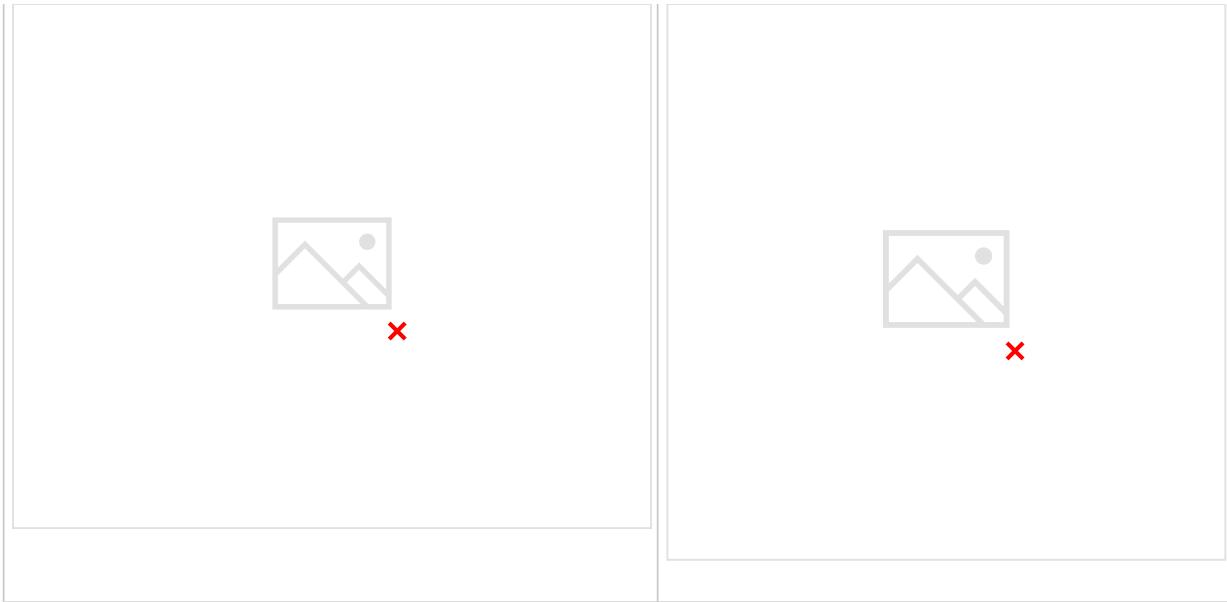
답:

서포트 벡터 머신(support vector machine)은 기계학습의 분야 중 하나로 패턴인식, 자료분석을 위한 지도학습 모델이며 주로 분류와 회귀를 위해 사용한다. 두 클래스가 주어졌을 때 SVM 알고리즘은 주어진 데이터 집합을 바탕으로 하여 새로운 데이터가 어느 클래스에 속할지 판단하는 비확률적 이진 선형 분류 모델을 생성한다.

선형분류와 더불어 비선형 분류에서도 사용된다.

■ 이론설명2. 서포트 벡터 머신에서 결정경계가 무엇인가요?

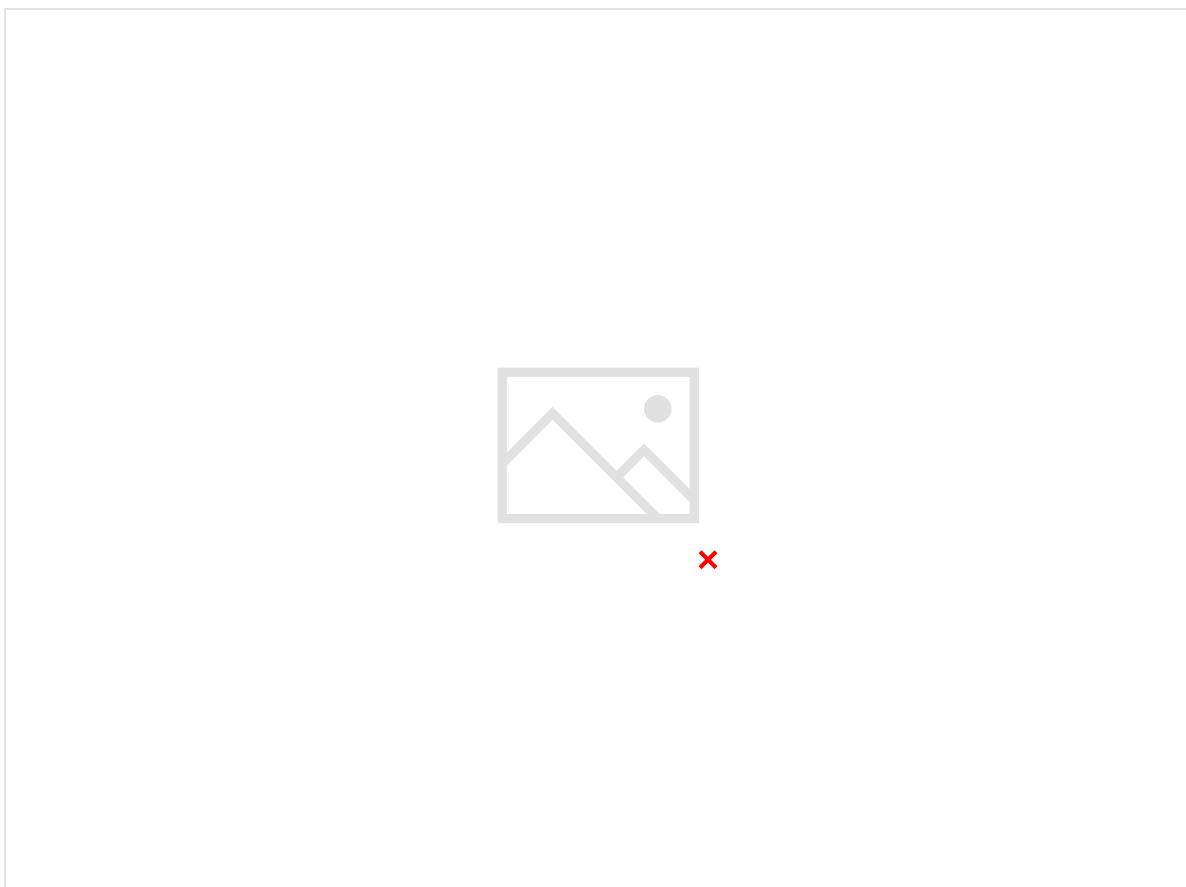
2차원일때	3차원일때
-------	-------



답 :

결정경계란 분류를 위한 기준선이다. 2차원일때는 선이고 3차원일때는 평면이 된다. 우리가 생각할 수 있는 부분은 3차원까지고 차원이 더 많아지게 되면 평면이 아니라 초평면이 된다.

■ 이론설명3. 아래에서 데이터를 가장 잘 분류한 결정경계는 ?



답 :

A 또는 F인데 두 클래스(분류) 사이의 거리가 가장 먼 선이 좋은 결정경계이다.

■ 이론설명4. 서포트 벡터머신에서 서포트 벡터는 무엇입니까?



답 :

결정경계와 가까이있는 데이터 포인트들을 의미한다. 이 데이터들이 경계를 정의하는 결정적인 역할을 한다.

■ 이론설명5. 서포트 벡터머신에서 마진(Margin)이 무엇입니까?



✗

답 :

점선으로부터 결정경계까지의 거리가 마진이다. 최적의 결정경계는 마진을 최대화 한다.

■ 이론설명6. 소프트 마진은 무엇이고 하드 마진은 무엇입니까 ?



하드마진 : 오버피팅 문제 발생할 수 있다. 훈련 정확도↑, 테스트 정확도↓

소프트마진 : 언더피팅 문제가 발생할 수 있다. 훈련정확도↓, 테스트 정확도↓

답 :

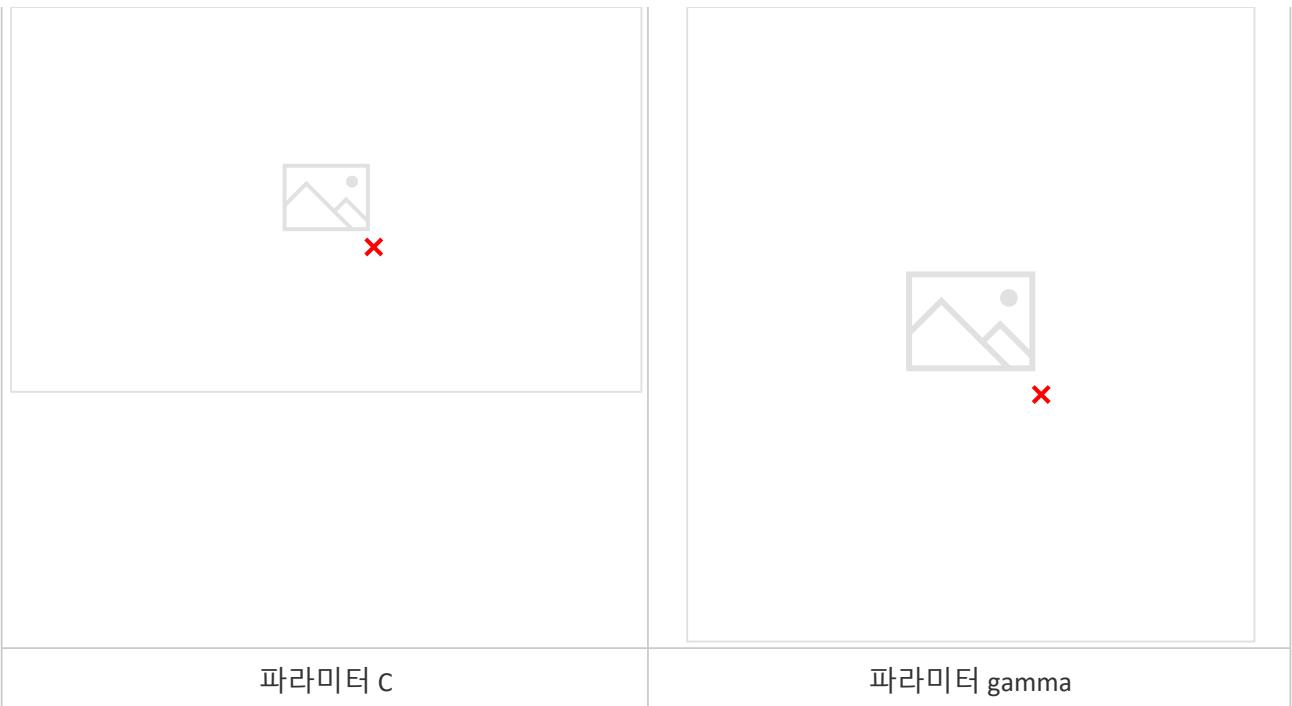
위의 그림 중 위쪽 그림은 아웃 라이어를 허용하지 않고 기준을 까다롭게 세운 모델이다.

이걸 **하드마진**이라고 한다.

그리고 서포트 벡터와 결정경계사이의 거리가 매우 좁다. 즉 마진이 작아진다.

아래쪽 그림은 아웃라이어들이 마진 안에 어느정도 포함되도록 너그럽게 기준을 잡았다. 이걸 **소프트마진**이라고 한다. 이렇게 너그럽게 잡아놓으면 서포트 벡터와 결정경계 사이의 거리가 멀어진다. 대신 너무 대충대충 학습하는 꼴이라 언더피팅 문제가 발생할 수 있다.

■ 이론설명7. 그럼 이런 오류를 어느정도 허용하는지 결정하는 하이퍼 파라미터는 무엇입니까 ?



C가 너무 높으면 훈련데이터의 정확도 ↑, 그러나 오버피팅 발생

C가 낮으면 정확도가 떨어지고 언더피팅 발생

따라서 적절한 C를 알아내기!!

감마 : 비선형 분류시 경계선 결정하는 것

답 :

C는 C가 클수록 하드마진(오류를 허용 안함)이 일어나고 C가 작을수록 소프트마진(오류를 허용함)이 일어난다.

gamma는 결정경계를 얼마나 유연하게 그을 것인지 정해주는 것이다. gamma를 높이면 학습 데이터에 많이 의존해서 결정경계를 구불구불하게 긋게 된다. 그래서 오버피팅을 초래할 수 있다.

반대로 gamma를 낮추면 학습 데이터에 별로 의존하지 않고 결정경계를 긋게 되므로 언더피팅이 발생 할 수 있다.

■ 원리설명1. 서포트 벡터 머신에서 벡터에서 벡터가 무엇인가요 ?

답 :



■ 원리설명2. 벡터가 실생활에서 어떻게 활용되고 있나요?



■ 원리설명3. normal vector 가 무엇인가요 ?



답 :

법선벡터란 한 평면이나 직선에 대해 수직인 벡터를 말한다.

■ 원리설명4. 벡터의 내적 공식이 어떻게 되나요 ?

답 :



■ 원리설명5. 벡터를 이용해서 어떻게 데이터를 분류하는가?



■ 7장. 서포트 벡터 머신 실습

#1. Data

```
data(iris)
str(iris)
library(ggplot2)
qplot( Petal.Length, Petal.Width, data=iris, color=Species)
```

#2. support Vector Machine

```
library(e1071)
mymodel <- svm(Species~. , data=iris)
summary(mymodel)
plot(mymodel, data=iris, Petal.Width~Petal.Length,
     slice = list(Sepal.Width=3, Sepal.Length=4))
```

#3. Confusion Matrix and Misclassification Error

```
pred <- predict(mymodel, iris)

tab <- table(Predicted = pred, Actual = iris$Species)
tab
1- sum(diag(tab))/sum(tab)
```

#4. 커널변경1 (kernel="linear")

```
library(e1071)
mymodel <- svm(Species~. , data=iris , kernel="linear" )
summary(mymodel)
plot(mymodel, data=iris, Petal.Width~Petal.Length,
     slice = list(Sepal.Width=3, Sepal.Length=4))

pred <- predict(mymodel, iris)

tab <- table(Predicted = pred, Actual = iris$Species)
tab
1- sum(diag(tab))/sum(tab)
```

#5. 커널변경2 (kernel="polynomial")

관련 영상: <https://www.youtube.com/watch?v=3liCbRZPrZA>



```
library(e1071)
mymodel <- svm(Species~. , data=iris , kernel="polynomial" )
summary(mymodel)
plot(mymodel, data=iris, Petal.Width~Petal.Length,
     slice = list(Sepal.Width=3, Sepal.Length=4))

pred <- predict(mymodel, iris)

tab <- table(Predicted = pred, Actual = iris$Species)
tab
1- sum(diag(tab))/sum(tab)
```

#6. 커널변경3 (kernel="sigmoid")

```
library(e1071)
mymodel <- svm(Species~. , data=iris , kernel="sigmoid" )
summary(mymodel)
```

```
plot(mymodel, data=iris, Petal.Width~Petal.Length,  
     slice = list(Sepal.Width=3, Sepal.Length=4))  
  
pred <- predict(mymodel, iris)  
  
tab <- table(Predicted = pred, Actual = iris$Species)  
tab  
1- sum(diag(tab))/sum(tab)
```

#7. 성능개선

```
set.seed(123)  
tmodel <- tune( svm, Species~. , data=iris, range=list(epsilon=seq(0.1,0.1), cost=2^(2:9) ))  
plot(tmodel)  
summary(tmodel)  
  
#best model  
mymodel <- tmodel$best.model  
summary(mymodel)  
plot( mymodel, data=iris, Petal.Width~Petal.Length, slice=list(Sepal.Width=3, Sepal.Length=4))  
  
pred <- predict(mymodel, iris)  
tab <- table(Predicted = pred, Actual = iris$Species)  
tab  
1- sum(diag(tab))/sum(tab)
```

(오늘의 마지막 문제 2/17) 미국 대학원 입학데이터를 서포트벡터 머신으로 분류하고 정확도를 확인하시오!



×

#1. Data

```
binary <- read.csv("binary.csv")
library(ggplot2)
str(binary) #변수 형태 확인
qplot( gre, gpa, data=binary, color=admit)
binary$admit <- as.factor(binary$admit) # 종속변수 팩터로 변경
```

#2. support Vector Machine

```
library(e1071)
mymodel <- svm(admit~. , data=binary)
summary(mymodel)
plot(mymodel, data=binary, gre~gpa)
```

#3. Confusion Matrix and Misclassification Error

```
pred <- predict(mymodel, binary)

tab <- table(Predicted = pred, Actual = binary$admit)
tab
1- sum(diag(tab))/sum(tab) #0.2775
```

#4. 성능개선

```
set.seed(123)
tmodel <- tune( svm, admit~. , data=binary, range=list(epsilon=seq(0,1,0.1), cost=2^(2:9) ))
plot(tmodel)
summary(tmodel)
```

#best model

```
mymodel <- tmodel$best.model
summary(mymodel)
plot(mymodel, data=binary, gpa ~ gre)

pred <- predict(mymodel2, binary)
tab <- table(Predicted = pred, Actual = binary$admit)
tab
1- sum(diag(tab))/sum(tab) #0.265
```


02/18

2021년 2월 18일 목요일 오전 10:01

■ 원리설명7. normal vector 를 찾기 위해 어떻게 학습이 되는가?

답 :

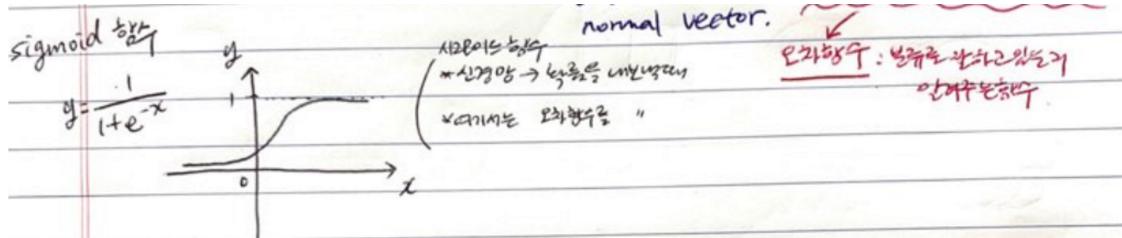
- 로지스틱 회귀와 서포트 백터 머신 식 비교

Logistic regression cost function

$$J(\theta) = -\frac{1}{m} \left[\sum_{i=1}^m y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)})) \right]$$

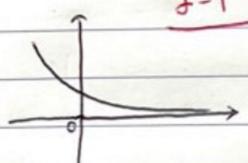
Cost function of SVM

$$\min_{\theta} C \sum_{i=1}^m \left[y^{(i)} \text{cost}_1(\theta^T x^{(i)}) + (1 - y^{(i)}) \text{cost}_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^n \theta_j^2$$



x가 양수쪽으로 올라갈 때 y가 1에 가까워지고 (나아짐)
 "증수쪽으로" 때 "0이" "0이" (보정안짐)
 ∴ 보정의 반대이다.

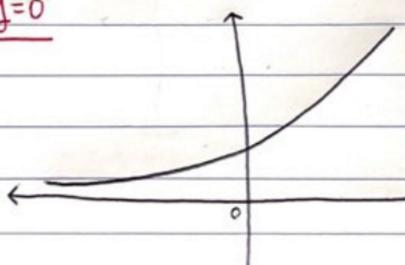
* $-\log(g(\vec{w} \cdot \vec{x}))$: 시그모이드함수가 1을 쓰려고 -를 넣겠다.



양수쪽으로 증수록 y가 0에 가까워진다
 ⇒ x가 증수의 증수록, 양수쪽
 x가 증수쪽으로 증수록 y가 많이 가까워지다
 ∴ 22번 y=1 일때, 맞는데 y=0일때 맞지 X.

$\frac{1}{n} \sum (-\log(g(\vec{w} \cdot \vec{x})) \cdot y - \log(g(\vec{w} \cdot \vec{x})) \times (1-y))$ "로지스틱 회귀식"

$y=0$



$=j(\theta)$

누락치를 더해나가며 차례로
 비율 차이를 더해나가며 차례로
 차이

■ iris 데이터를 로지스틱 회귀로 분류했을 때와 서포트벡터 머신으로 분류했을 때의 차이를 테스트

####

Logistic Regression

Read data file

```
mydata <- read.csv("d://data//binary.csv", header = T)
str(mydata)
mydata$admit <- as.factor(mydata$admit)
mydata$rank <- as.factor(mydata$rank)
```

Two-way table of factor variables

```
xtabs(~admit + rank, data = mydata)
```

Partition data - train (80%) & test (20%)

```
set.seed(1234)
ind <- sample(2, nrow(mydata), replace = T, prob = c(0.8, 0.2))
train <- mydata[ind==1,]
test <- mydata[ind==2,]
```

```

# Logistic regression model
mymodel <- glm(admit ~ gpa + rank, data = train, family = 'binomial')
summary(mymodel)

# Prediction
p1 <- predict(mymodel, train, type = 'response')
head(p1)
head(train)

# Misclassification error - train data
pred1 <- ifelse(p1>0.5, 1, 0)
tab1 <- table(Predicted = pred1, Actual = train$admit)
tab1
1 - sum(diag(tab1))/sum(tab1)

# Misclassification error - test data
p2 <- predict(mymodel, test, type = 'response')
pred2 <- ifelse(p2>0.5, 1, 0)
tab2 <- table(Predicted = pred2, Actual = test$admit)
tab2
1 - sum(diag(tab2))/sum(tab2)

# Goodness-of-fit test
with(mymodel, pchisq(null.deviance - deviance, df.null-df.residual, lower.tail = F))

```

####2####

1. 로지스틱 회귀로 했을때

```

# Logistic Regression

# Read data file
mydata <- read.csv("d:\data\binary.csv", header = T)
str(mydata)
mydata$admit <- as.factor(mydata$admit)
mydata$rank <- as.factor(mydata$rank)

# Two-way table of factor variables
xtabs(~admit + rank, data = mydata)

# Partition data - train (80%) & test (20%)
set.seed(1234)
ind <- sample(2, nrow(mydata), replace = T, prob = c(0.8, 0.2))
train <- mydata[ind==1,]
test <- mydata[ind==2,]

# Logistic regression model
mymodel <- glm(admit ~ gpa + rank, data = train, family = 'binomial')

```

```

summary(mymodel)

# Prediction
p1 <- predict(mymodel, train, type = 'response')
head(p1)
head(train)

# Misclassification error - train data
pred1 <- ifelse(p1>0.5, 1, 0)
tab1 <- table(Predicted = pred1, Actual = train$admit)
tab1
1 - sum(diag(tab1))/sum(tab1)

# Misclassification error - test data
p2 <- predict(mymodel, test, type = 'response')
pred2 <- ifelse(p2>0.5, 1, 0)
tab2 <- table(Predicted = pred2, Actual = test$admit)
tab2
1 - sum(diag(tab2))/sum(tab2)

# Goodness-of-fit test
with(mymodel, pchisq(null.deviance - deviance, df.null-df.residual, lower.tail = F))

```

2. 서포트 벡터 머신일때 (비선형으로) 실험

```

####Caret Packages -- train()
##0.1 Load data
binary <- read.csv("binary.csv")
binary$admit <- factor( binary$admit, labels=c("Y", "N"))
str(binary)
##0.2 Devide Db

set.seed(5311)
library(caret)
k <- createDataPartition(binary$admit, p=0.80, list=F)
trainset <- binary[k, ]
testset <- binary[-k, ]
nrow(trainset) #321
nrow(testset) #79

##3. svmPoly -----
#install.packages("kernlab")
library(kernlab)

ctrl <- trainControl(method="cv", number=10, # k-hold 교차검정하겠다.
                      selectionFunction="oneSE")

```

```

grid <- expand.grid(C=c(1,2,3,4,5)) # 하이퍼파라미터 C 를 지정

m <- train(admit~, data=trainset, method="svmPoly",
            metric="Accuracy",
            trControl=ctrl,
            tunegrid=grid)

m #check model details

#predict
pred <- predict(m, testset)

#Definition of label column vector & Labldata
actual_type <- testset$admit
predict_type <- pred
positive_val <- "Y"
negative_val <- "N"

#1) Accuracy
tab <- table(Predicted=pred, Actual=testset$admit)
tab
sum(diag(tab)) / sum(tab) #testset_Accuracy

#2) Kappa statistics
#install.packages("vcd")
library(vcd)
table( actual_type, predict_type)
Kappa( table( actual_type, predict_type) )

```

(점심시간 문제 2/18) 방금 실습했던 서포트 벡터 머신의 커널을 다른 커널로 변경해서 실험 하시오!

정확도	svm	caret
1. svmPoly 0.721519	svmPoly	polynomial
3. sigmoid 0.721519	sigmoid	

■ 서포트 벡터 머신으로 필기체 데이터를 분류하기

필기체 데이터는 딥러닝 때 배운 mnist 데이터인데 모두 숫자로 되어있는 데이터이다.

mnist 데이터는 필기체 숫자 0~9번까지 중 총 10개의 클래스로 되어있고 한 클래스당 6000개의 필기체 데이터를 포함하고 있다.

하나의 필기체는 784개의 픽셀로 되어있다. 총 6만장의 데이터로 구성되어 있다.
전부 숫자로만 되어있는 데이터여서 신경망이나 서포트 벡터 머신을 구현할 때 유리한 데이터

이다.

```
install.packages("caret")
install.packages("doParallel") # 병렬로 작업하기 위한 패키지
install.packages("kernlab")
install.packages("ggplot2")
install.packages("lattice")

library(ggplot2)
library(lattice)
library(kernlab)
library(caret)
library(doParallel)

# Enable parallel processing.

cl <- makeCluster(detectCores())
cl

# 결과 : socket cluster with 8 nodes on host 'localhost' 병렬처리로 8개의 프로세서가 동시에
# 수행하게 함

registerDoParallel(cl)

# Load the MNIST digit recognition dataset into R
# http://yann.lecun.com/exdb/mnist/
# assume you have all 4 files and gunzip'd them
# creates train$n, train$x, train$y and test$n, test$x, test$y
# e.g. train$x is a 60000 x 784 matrix, each row is one digit (28x28)
# call: show_digit(train$x[5,]) to see a digit.
# brendan o'connor - gist.github.com/39760 - anyall.org

load_mnist <- function() {
  load_image_file <- function(filename) {
    ret = list()
    f = file(filename,'rb')
    readBin(f,'integer',n=1,size=4,endian='big')
    ret$n = readBin(f,'integer',n=1,size=4,endian='big')
    nrow = readBin(f,'integer',n=1,size=4,endian='big')
    ncol = readBin(f,'integer',n=1,size=4,endian='big')
    x = readBin(f,'integer',n=ret$n*nrow*ncol,size=1,signed=F)
    ret$x = matrix(x, ncol=nrow*ncol, byrow=T)
    close(f)
    ret
  }
  load_label_file <- function(filename) {
    f = file(filename,'rb')
    readBin(f,'integer',n=1,size=4,endian='big')
    n = readBin(f,'integer',n=1,size=4,endian='big')
    y = readBin(f,'integer',n=n,size=1,signed=F)
    close(f)
```

```

y
}

train <- load_image_file('train-images.idx3-ubyte')
test <- load_image_file('t10k-images.idx3-ubyte')

train$y <- load_label_file('train-labels.idx1-ubyte')
test$y <- load_label_file('t10k-labels.idx1-ubyte')
}

show_digit <- function(arr784, col=gray(12:1/12), ...) {
  image(matrix(arr784, nrow=28)[,28:1], col=col, ...)
}

# load_mnist 함수는 mnist 데이터를 R studio에 로드하기 위한 함수
# show_digit 함수는 필기체 데이터를 하나 시각화 하려고 만든 함수

# 훈련데이터와 테스트 데이터를 구성하기 위한 데이터 프레임 생성

train <- data.frame()
test <- data.frame()

```

```

# Load data.

load_mnist()

length(train$y) #60000
length(train$x) #47040000

# Normalize: X = (X - min) / (max - min) => X = (X - 0) / (255 - 0) => X = X / 255.

train$x <- train$x / 255

# Setup training data with digit and pixel values with 60/40 split for train/cv.

inTrain = data.frame(y=train$y, train$x)
head(inTrain$y, 100) #head로 숫자 미리보기

inTrain$y <- as.factor(inTrain$y)
str(inTrain) #y만 팩터로 보여짐

# 훈련 데이터 60000개를 6대 4로 나눠서 6은 훈련시킬 때 쓰고 4는 테스트할 때 사용
trainIndex = createDataPartition(inTrain$y, p = 0.60, list=FALSE)
training = inTrain[trainIndex,]
cv = inTrain[-trainIndex,]

nrow(training) #36004
nrow(cv) # 23996

```

```
# SVM. 95/94.
```

```
fit <- train(y ~ ., data = head(training, 1000), method = 'svmRadial', tuneGrid =  
data.frame(sigma=0.0107249, C=1))  
fit
```

```
#1000개의 필기체 데이터로 훈련한 모델에 테스트 데이터 1000개를 예측한다.
```

```
results <- predict(fit, newdata = head(cv, 1000))  
results
```

```
# 잘 예측했는지 확인한다.
```

```
confusionMatrix(results, head(cv$y, 1000))
```

```
#훈련데이터 중 5번째 숫자가 무엇인지 시각화해서 확인
```

```
show_digit(as.matrix(training[5,2:785]))
```

```
# Predict the digit.
```

```
predict(fit, newdata = training[5,])
```

```
# Check the actual answer for the digit.
```

```
training[5,1]
```

■ 문제. 훈련 데이터의 102번째 행이 무엇인지 확인하고 만든 svm 모델에 102 번째 훈련 데이터를 넣고 무엇으로 예측하는지 확인하시오!

```
#훈련데이터 중 102번째 숫자가 무엇인지 시각화해서 확인
```

```
show_digit(as.matrix(training[102,2:785]))
```

```
# Predict the digit.
```

```
predict(fit, newdata = training[102, ])
```

```
# Check the actual answer for the digit.
```

```
training[102,1]
```

[캐글 도전] 타이타닉 데이터 준비 작업

1. 파이썬에서 아래와 같이 seaborn 에 내장되어 있는 타이타닉 데이터를 내려받습니다.

```
import seaborn as sns  
import pandas as pd
```

```

tat = sns.load_dataset('titanic')

print(tat)

tat2 = pd.DataFrame(tat)

# 판다스의 데이터 프레임으로 바꿔줌
tat2.to_csv('d:\data\titanic.csv', sep=',', na_rep='NaN') # missing data representation
(결측값 표기)

```

[캐글도전] 랜덤 포레스트로 타이타닉 생존자 분류 실험

타이타닉 생존자 분류

1. 데이터 로드

```

tat <- read.csv("titanic.csv", stringsAsFactors = TRUE)
View(tat)
head(tat)
nrow(tat) #891

#survived : 생존=1, 죽음=0 --> 종속변수
#pclass : 승객 등급/ 1등급=1, 2등급=2, 3등급=3
#sibsp : 함께 탑승한 형제 또는 배우자 수
#parch : 함께 탑승한 부모 또는 자녀 수
#ticket : 티켓 번호
#cabin : 선실 번호
#embarked : 탑승장소 S=Southhampton, C=Cherbourg, Q=Queenstown

```

#2. 결측치 확인

```

colSums(is.na(tat))
?mean

#나이의 결측치를 나이의 평균값으로 치환
tat$age[is.na(tat$age)] <- mean(tat$age,na.rm=TRUE)
colSums(is.na(tat)) #이제 결측치 없다

```

#3. 이상치 확인

```

library(outliers)

grubbs.flag <- function(x) {
  outliers <- NULL

```

```

test <- x
grubbs.result <- grubbs.test(test)
pv <- grubbs.result$p.value
while(pv < 0.05) {
  outliers <- c(outliers,as.numeric(strsplit(grubbs.result$alternative," ")[[1]][3]))
  test <- x[!x %in% outliers]
  grubbs.result <- grubbs.test(test)
  pv <- grubbs.result$p.value
}
return(data.frame(X=x,Outlier=(x %in% outliers)))
}

wisc <- read.csv("tatanic.csv")

for (i in c(2,3,5,6,8)){

  a = grubbs.flag(wisc[,colnames(wisc)[i]])
  b = a$a$Outlier==TRUE,"Outlier"]
  print ( paste( colnames(wisc)[i] , '--> ', length(b) ) )

}

```

#4. 랜덤포레스트로 분류합니다.

```

library(caret)
library(C50)
library(irr)

nrow(tat) #891

set.seed(123)
tat_shuffle <- sample(1:891, 891)
tat_shuffle
tat2 <- tat[tat_shuffle,]
tat2

set.seed(123)
in_train <- createDataPartition(tat2$survived, p = 0.75, list = FALSE)
tat_train <- tat2[in_train, ] # 훈련 데이터 구성
tat_test <- tat2[-in_train, ] # 테스트 데이터 구성
nrow(tat_train) #669
nrow(tat_test) #222

m <- train( survived~ . , data=tat_train, method="rf" )

# 랜덤포레스트: 의사결정트리 + 앙상블 기법
m # 튜닝한 결과를 확인할 수 있다.

p <- predict( m, tat_test )
# 0~1사이의 확률로 나오기 때문에 0.5이상인것은 1이 되게 함

```

```

round(p)

table(round(p), tat_test$survived)

library(gmodels)
y <- CrossTable(tat_test$survived ,round(p) )
sum(y$prop.tbl * diag(2)) #100% 나온다.

```

문제1. 이번에는 나이의 결측치를 나이의 평균값으로 하지말고 승객 나이중에 최대값으로 결측치를 채우고 다시 테스트 데이터의 정확도를 확인하시오!

타이타닉 생존자 분류

1. 데이터 로드

```

tat <- read.csv("titanic.csv", stringsAsFactors = TRUE)
View(tat)
head(tat)
nrow(tat) #891

#survived : 생존=1, 죽음=0 --> 종속변수
#pclass : 승객 등급/ 1등급=1, 2등급=2, 3등급=3
#sibsp : 함께 탑승한 형제 또는 배우자 수
#parch : 함께 탑승한 부모 또는 자녀 수
#ticket : 티켓 번호
#cabin : 선실 번호
#embarked : 탑승장소 S=Southhampton, C=Cherbourg, Q=Queenstown

```

#2. 결측치 확인

```

colSums(is.na(tat) )
?mean

#나이의 결측치를 나이의 평균값으로 치환
tat$age[is.na(tat$age)] <- max(tat$age,na.rm=TRUE)
colSums(is.na(tat)) #이제 결측치 없다

```

#3. 이상치 확인

```

library(outliers)

grubbs.flag <- function(x) {
  outliers <- NULL
  test <- x
  grubbs.result <- grubbs.test(test)
  pv <- grubbs.result$p.value
  while(pv < 0.05) {

```

```

outliers <- c(outliers,as.numeric(strsplit(grubbs.result$alternative," ")[[1]][3]))
test <- x[!x %in% outliers]
grubbs.result <- grubbs.test(test)
pv <- grubbs.result$p.value
}
return(data.frame(X=x,Outlier=(x %in% outliers)))
}

wisc <- read.csv("tatanic.csv")

for (i in c(2,3,5,6,8)){

  a = grubbs.flag(wisc[,colnames(wisc)[i]])
  b = a[a$Outlier==TRUE,"Outlier"]
  print ( paste( colnames(wisc)[i] , '--> ', length(b) ) )

}

```

#4. 랜덤포레스트로 분류합니다.

```

library(caret)
library(C50)
library(irr)

nrow(tat) #891

set.seed(123)
tat_shuffle <- sample(1:891, 891)
tat_shuffle
tat2 <- tat[tat_shuffle,]
tat2

set.seed(123)
in_train <- createDataPartition(tat2$survived, p = 0.75, list = FALSE)
tat_train <- tat2[in_train, ] # 훈련 데이터 구성
tat_test <- tat2[-in_train, ] # 테스트 데이터 구성
nrow(tat_train) #669
nrow(tat_test) #222

m <- train( survived~ . , data=tat_train, method="rf" )

# 랜덤포레스트: 의사결정트리 + 앙상블 기법
m # 튜닝한 결과를 확인할 수 있다.

p <- predict( m, tat_test )
# 0~1사이의 확률로 나오기 때문에 0.5이상인것은 1이 되게 함
round(p)

table(round(p), tat_test$survived)

```

```
library(gmodels)
y <- CrossTable(tat_test$survived ,round(p) )
sum(y$prop.tbl * diag(2))
```

#100% 나온다.

[캐글도전] 파이썬의 sklearn 의 보스톤 하우징 데이터 가져오기

```
from sklearn.datasets import load_boston #scikit-learn의 datasets에서 sample data import
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

boston = load_boston() # boston dataset load
print(boston.keys()) # 각 key 확인
print(boston.DESCR) # boston datasets description

# 데이터 프레임으로 변환
df = pd.DataFrame(data=boston.data, columns=boston.feature_names)
```

[캐글도전] 보스톤 하우징 데이터 회귀분석 R 전체코드(승순이 코드)

1. 데이터 불러오기

```
getwd()
library(data.table)

boston <- read.csv("d://data//boston.csv" )

#0.shuffle 을 먼저 합니다.
set.seed(123)
boston <- boston[1:891, 891]
boston_shuffle
boston2 <- boston[boston_shuffle,]
```

```
boston2
```

```
set.seed(123)
set.seed(1234)
ind <- sample(2, nrow(boston2), replace = T, prob = c(0.8, 0.2))
boston_train <- boston2[ind==1,]
boston_test <- boston2[ind==2,]
```

```
# 상관관계 분석, vif 확인등의 절차를 거친 후 추가되는 데이터
```

```
# 1)
```

```
boston_train$lstat_rm <- ifelse( ( boston_train$lstat >5 & boston_train$rm <= 5) , 1, 0 )
boston_test$lstat_rm <- ifelse( ( boston_test$lstat >5 & boston_test$rm <= 5) , 1, 0 )
boston_train$age_indus <- ifelse( ( boston_train$age < 30 & boston_train$indus <= 10) , 1,
0 )

boston_test$age_ind <- ifelse( ( boston_test$age < 30 & boston_test$indus <= 10) , 1, 0 )
```

```
##### 2. 데이터 전처리 #####
```

```
# 1) 결측치
```

```
# 1-1. 결측치 확인
```

```
sum(is.na(boston_train)) #0개
sum(is.na(boston_test)) #0개
```

```
# 1-2. 결측치 위치확인
```

```
#{1} 데이터셋명[complete.cases(데이터셋명),] # 결측치가 없으면 True
#{2} 데이터셋명[!complete.cases(데이터셋명),] # 결측치가 있으면 True
```

```
# 1-3. 결측치 삭제
```

```
#boston_train<-na.omit(boston_train)
#boston_test<-na.omit(boston_test)
```

```
# 1-4. 결측치 대체
```

```
# (1) 결측할 값 직접지정
```

```
#boston_train$대체할변수있는컬럼[is.na(데이터셋$컬럼명)]<-일괄대체할값
# 지정 컬럼에서 na가 있으면 지정하는 값으로 대체
```

```
# (2) 통계값으로 대체
```

```
#install.packages("DMwR")
```

```
#library(DMwR)

#centralImputation(데이터셋명) # NA중央값 대체 : 숫자의 경우 중앙값, 팩터의 경우 최빈값
#knnImputation(데이터셋명) # means를 활용한 결측치 대체
```

2) 정규화

```
#데이터 정규화를 위한 함수생성
```

```
normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x)) )
}
```

```
boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )
```

```
#정규화 결과 확인
```

```
summary(boston_train_norm)
summary(boston_test_norm)
```

```
#####
##### 3. 모델링, 데이터 학습, 성능검증 , 평가
#####
```

1) 모델링 (변수선택)

```
# 1-1. 다중공선성 확인
```

```
#install.packages("car")
library(car)
```

```
lml <- lm(medv~., data=boston_train_norm)
```

```
vif(lml) # 팽창지수(=다중공선성=vif)
```

```
vif(lml) > 5 #vif 5이상이면 TRUE / 5이하면 FALSE
```

```
# strict하게 본다면(vif > 5) indus, rad, tax, age_30, indus_10 의 다중공선성 높음
```

```
# 일반적으로 본다면(vif > 10) = 제거해야할 변수 없음
```

```
# 이번 실습은 vif > 10을 기준으로 진행
```

```
# 1-2. 분산이 0에 가까운 변수제거
```

```
# 데이터의 분산이 0에 가깝다 = 서로 다른 관찰을 구분하는데 소용이 없다. = 제거한다.
```

```
# 결과치에서 nzv 컬럼에 TRUE로 뜨는 값 제거
```

```
#install.packages("caret")
```

```
library(caret)
```

```
nearZeroVar(boston_test_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
```

```
nearZeroVar(boston_train_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
```

```
# nzv = TRUE인 변수들 제거  
#boston_train <- boston_train[,-nearZeroVar(boston_train)]  
#boston_test <- boston_test[,-nearZeroVar(boston_test)]
```

1-3. 상관관계

```
#install.packages("corrplot")  
library(corrplot)  
#기본값은 원형, shade=네모칸, ellipse = 타원(양의상관 오른쪽, 음의상관 왼쪽)  
# circle = 원형, number = 수치로 표현  
corrplot(cor(boston_train_norm), method = "number")
```

1-4. 변수중요도 확인

```
# 변수중요도 : "해당 변수가 상대적으로 얼마만큼 종속변수에 영향을 주는가?"
```

```
# 랜덤 포레스트 방식을 활용
```

```
#install.packages("randomForest")  
library(randomForest)  
#rf <- randomForest(medv~., data=boston_train_norm)  
rf <- randomForest(medv~., data=boston_train_norm, importance=TRUE)
```

```
#중요도 확인방법 2가지
```

1) 중요도 수치로 확인

```
varImp(rf)  
importance(rf) #IncMSE = 정확도, IncNodePurity = 중요도 = importance
```

2) plot형태로 시각화해서 확인

```
varImpPlot(rf, main="Varplot of boston_train")
```

1-5. 단계적 회귀분석으로 유효변수골라내기

```
#1번째시도 (기본적으로 주어진 모든 독립변수 활용하여 모델링)
```

```
# vif 확인시 만든 lm과 동일
```

```
model1 <- lm(formula=medv ~ crim + chas + nox + rm + dis + rad + ptratio + lstat, data = boston_train_norm)
```

```
#stepwise(단계적 회귀분석 방법)로 가장 유효한 변수 확인하기
```

```
lm2 <- step(model1, direction="both")  
# medv ~ crim + chas + nox + rm + dis + rad + ptratio + lstat
```

```
# 결과적 추시가 좋지 않아 단계적 회귀분석을 단독 적용하는 것은 무리라 판단. 상관관계를  
감안하여 적용하기로함
```

#2번째 시도 (상관관계 분석을 통해 알아낸 사실 포함하여 적용)

```
lml2<-step(lml, direction="both")  
#medv ~ crim + zn + indus + chas + nox + rm + dis + rad + tax + ptratio + black +  
lstat + lstat_rm
```

여기까지 확인한 후에 다시 1번으로 돌아가 컬럼 추가 및 전처리를 수행함

2-1. 모델링

```
#stepwise + 상관계수 반영한 결과 고려하여 모델링  
boston_reg_model <- lm(medv ~ crim + zn + indus + chas + nox + rm + dis + rad + tax +  
ptratio + black + lstat + lstat_rm, data=boston_train_norm )  
  
model_results <- predict(boston_reg_model, boston_test_norm)  
model_results
```

2-2. 결과값 역정규화 (kaggle과 데이터 형식 맞추기)

```
denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data  
$medv)  
pred_medv_un <- denormalize(model_results)  
  
sample <- cbind(boston_test$ID, pred_medv_un)  
  
head(sample)  
  
colnames(sample) <- c("id", "medv")  
  
write.csv(sample, "Submission_sample16.csv", row.names=FALSE)
```

(오늘의 마지막 문제 2/18) 상관관계 높다고 오차가 낮은 아니므로 오차가 나오게 하시오.

#정확도 구하는 식

```
tab1 <- table(Predicted = pred_medv_un, Actual = boston_test$x)  
tab1  
1 - sum(diag(tab1))/sum(tab1)
```

위의 코드에서 해당부분만 추가

02/19

2021년 2월 19일 금요일 오전 9:44

■ 어제 배웠던 내용 복습

1. 서포트 벡터머신의 오차함수가 어떻게 만들어졌는지 원리를 설명
2. 로지스틱 회귀와 서포트 벡터 머신의 차이
3. 서포트 벡터머신의 성능을 개선하기 위해 11장의 caret패키지를 이용하는 방법
4. 서포트 벡터머신을 이용해서 필기체 데이터를 분류
5. 캐글에 분류와 회귀 competition에 도전하기 위해 타이타닉과 보스톤 하우징 데이터를 파이썬에 내장되어있는 데이터를 R로 가져옴
6. 타이타닉 데이터를 랜덤 포레스트로 분류해서 테스트 데이터에 대해서 100%의 정확도를 보았다.
7. 보스톤 하우징 데이터는 승순씨의 코드를 가지고 분석해서 오차 5.39의 회귀모델을 생성했다.

" 캐글 도전 " --> 분류
--> 수치예측

오늘 배울 내용 : 1. 11장에서 배웠던 caret함수를 이용해서 앞에 배웠던 알고리즘을 caret패키지의 train 함수에 적용

<https://cafe.daum.net/oracleoracle/Sg0x/963>

양상블 : 1. 배깅
2. 부스팅
3. 랜덤 포레스트

■ [캐글 도전] 보스톤하우징 데이터 회귀분석과 랜덤포레스트와의 예측결과 비교

1. 데이터 불러오기

```
getwd()  
library(data.table)  
  
boston <- read.csv("d:\data\boston.csv" )  
nrow(boston) #506
```

```
#0.shuffle 을 먼저 합니다.  
set.seed(123)  
  
boston_shuffle <- sample(1:506, 506)  
boston2 <- boston[boston_shuffle,]  
boston2  
  
# 훈련 데이터 80%, 테스트 데이터 20%  
set.seed(1234)  
ind <- sample(2, nrow(boston2), replace = T, prob = c(0.8, 0.2))  
boston_train <- boston2[ind==1,]  
boston_test <- boston2[ind==2,]  
nrow(boston_train) # 411  
nrow(boston_test) #95
```

#상관관계 분석, vif 확인등의 절차를 거친 후 추가되는 데이터

1)

```
tolower('AAA')  
colnames(boston_train) <- tolower(colnames(boston_train))  
colnames(boston_test) <- tolower(colnames(boston_test))  
  
str(boston_train)  
str(boston_test)
```

2. 데이터 전처리

1) 결측치

```
# 1-1. 결측치 확인  
sum(is.na(boston_train)) #0개  
sum(is.na(boston_test)) #0개
```

1-2. 결측치 위치확인

```
#(1) 데이터셋명[complete.cases(데이터셋명),] # 결측치가 없으면 True  
 #(2) 데이터셋명[!complete.cases(데이터셋명),] # 결측치가 있으면 True
```

1-3. 결측치 삭제

```
#boston_train<-na.omit(boston_train)  
#boston_test<-na.omit(boston_test)
```

1-4. 결측치 대체

```
# (1) 결측할 값 직접지정  
#boston_train$대체할변수있는컬럼[is.na(데이터셋$컬럼명)]<-일괄대체할값  
# 지정 컬럼에서 na가 있으면 지정하는 값으로 대체
```

```

# (2) 통계값으로 대체
#install.packages("DMwR")
#library(DMwR)

#centralImputation(데이터셋명) # NA중央값 대체 : 숫자의 경우 중앙값, 팩터의 경우 최빈
값
#knnImputation(데이터셋명) # means를 활용한 결측치 대체

# 2) 정규화
#데이터 정규화를 위한 함수생성
normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x) ) )
}

boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

#정규화 결과 확인
summary(boston_train_norm)
summary(boston_test_norm)

#####
##### 3. 모델링, 데이터 학습, 성능검증 , 평가 #####
#####

# 1) 모델링 (변수선택)

# 1-1. 다중공선성 확인
#install.packages("car")
library(car)

lml <- lm(price~., data=boston_train_norm)
vif(lml) # 팽창지수(=다중공선성=vif)
vif(lml) > 10 #vif 5이상이면 TRUE / 5이하면 FALSE

# strict하게 본다면(vif > 5) indus, rad, tax, age_30, indus_10 의 다중공선성 높음
# 일반적으로 본다면(vif > 10) = 제거해야할 변수 없음
# 이번 실습은 vif > 10을 기준으로 진행

# 1-2. 분산이 0에 가까운 변수제거
# 데이터의 분산이 0에 가깝다 = 서로 다른 관찰을 구분하는데 소용이 없다. = 제거한다.
# 결과치에서 nzv 컬럼에 TRUE로 뜨는 값 제거

#install.packages("caret")

```

```
library(caret)
nearZeroVar(boston_test_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
nearZeroVar(boston_train_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
```

2-1. 모델링

```
#stepwise + 상관계수 반영한 결과 고려하여 모델링
#boston_reg_model <- lm(price ~ ., data=boston_train_norm )
#boston_reg_model
#summary(boston_reg_model)

#####
# 성능 개선을 위해 추가된 부분

library(caret)
boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "lm")
#boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "rf")
boston_reg_model

#summary(boston_reg_model)

#####
model_results <- predict(boston_reg_model, boston_test_norm)
model_results
```

2-2. 결과값 역정규화 (kaggle과 데이터 형식 맞추기)

```
denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data
$medv)
pred_medv_un <- denormalize(model_results)

sample <- cbind(boston_test$x, pred_medv_un)

head(sample)
colnames(sample) <- c("id", "medv")
sample <- as.data.frame(sample)

boston_test$price
head(sample)

str(boston_test$price)
str(sample)

cor(boston_test$price, sample$medv) # 승순이가 만든 파생변수 추가전
#method = "lm" :0.8089203
#method = "rf" :0.9014441

rmse <- function(yi, yhat_i){
```

```

    sqrt(mean((yi - yhat_i)^2))
}

rmse(boston_test$price, sample$medv) # 승순이가 만든 파생변수 추가전
# method = "lm" : 5.537822
# method = "rf" : 3.184222

#write.csv(sample, "Submission_sample16.csv", row.names=FALSE)

```

회귀분석 결과 : 상관관계 : 0.8089203, 오차 : 5.537822

■ 양상을 랜덤포레스트를 적용해서 학습시키고 결과를 본다.

```
##### 1. 데이터 불러오기 #####

```

```

getwd()
library(data.table)

boston <- read.csv("d://data//boston.csv" )
nrow(boston) #506

#0.shuffle 을 먼저 합니다.
set.seed(123)

boston_shuffle <- sample(1:506, 506)
boston2 <- boston[boston_shuffle,]
boston2

# 훈련 데이터 80%, 테스트 데이터 20%
set.seed(1234)
ind <- sample(2, nrow(boston2), replace = T, prob = c(0.8, 0.2))
boston_train <- boston2[ind==1,]
boston_test <- boston2[ind==2,]
nrow(boston_train) # 411
nrow(boston_test) #95

```

#상관관계 분석, vif 확인등의 절차를 거친 후 추가되는 데이터

```

# 1)

tolower('AAA')
colnames(boston_train) <- tolower(colnames(boston_train))
colnames(boston_test) <- tolower(colnames(boston_test))

str(boston_train)
str(boston_test)

```

```
##### 2. 데이터 전처리 #####
```

1) 결측치

1-1. 결측치 확인

```
sum(is.na(boston_train)) #0개
```

```
sum(is.na(boston_test)) #0개
```

1-2. 결측치 위치확인

```
#(1) 데이터셋명[complete.cases(데이터셋명),] # 결측치가 없으면 True
```

```
#(2) 데이터셋명[!complete.cases(데이터셋명),] # 결측치가 있으면 True
```

1-3. 결측치 삭제

```
#boston_train<-na.omit(boston_train)
```

```
#boston_test<-na.omit(boston_test)
```

1-4. 결측치 대체

(1) 결측할 값 직접지정

```
#boston_train$대체할변수있는컬럼[is.na(데이터셋$컬럼명)]<-일괄대체할값
```

```
# 지정 컬럼에서 na가 있으면 지정하는 값으로 대체
```

(2) 통계값으로 대체

```
#install.packages("DMwR")
```

```
#library(DMwR)
```

```
#centralImputation(데이터셋명) # NA중央값 대체 : 숫자의 경우 중앙값, 팩터의 경우 최빈값
```

```
#knnImputation(데이터셋명) # means를 활용한 결측치 대체
```

2) 정규화

```
#데이터 정규화를 위한 함수생성
```

```
normalize <- function(x) {  
  return ( (x-min(x)) / (max(x) - min(x)) )  
}
```

```
boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))  
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize))
```

#정규화 결과 확인

```
summary(boston_train_norm)
```

```
summary(boston_test_norm)
```

```
##### 3. 모델링, 데이터 학습, 성능검증 , 평가
```

```
#####
```

```
# 1) 모델링 (변수선택)
```

```
# 1-1. 다중공선성 확인
```

```
#install.packages("car")
```

```
library(car)
```

```
lml <- lm(price~, data=boston_train_norm)
```

```
vif(lml) # 팽창지수(=다중공선성=vif)
```

```
vif(lml) > 10 #vif 5이상이면 TRUE / 5이하면 FALSE
```

```
# strict하게 본다면(vif > 5) indus, rad, tax, age_30, indus_10의 다중공선성 높음
```

```
# 일반적으로 본다면(vif > 10) = 제거해야할 변수 없음
```

```
# 이번 실습은 vif > 10을 기준으로 진행
```

```
# 1-2. 분산이 0에 가까운 변수제거
```

```
# 데이터의 분산이 0에 가깝다 = 서로 다른 관찰을 구분하는데 소용이 없다. = 제거한다.
```

```
# 결과치에서 nzv 컬럼에 TRUE로 뜨는 값 제거
```

```
#install.packages("caret")
```

```
library(caret)
```

```
nearZeroVar(boston_test_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
```

```
nearZeroVar(boston_train_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
```

```
# 2-1. 모델링
```

```
#stepwise + 상관계수 반영한 결과 고려하여 모델링
```

```
#boston_reg_model <- lm(price ~ ., data=boston_train_norm )
```

```
#boston_reg_model
```

```
#summary(boston_reg_model)
```

```
#####
```

```
# 성능 개선을 위해 추가된 부분
```

```
library(caret)
```

```
#boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "lm")
```

```
boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "rf")
```

```
boston_reg_model
```

```
#summary(boston_reg_model)
```

```
#####
```

```

model_results <- predict(boston_reg_model, boston_test_norm)
model_results

# 2-2. 결과값 역정규화 (kaggle과 데이터 형식 맞추기)
denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data
$medv)
pred_medv_un <- denormalize(model_results)

sample <- cbind(boston_test$x, pred_medv_un)

head(sample)
colnames(sample) <- c("id", "medv")
sample <- as.data.frame(sample)

boston_test$price
head(sample)

str(boston_test$price)
str(sample)

cor(boston_test$price, sample$medv) # 승순이가 만든 파생변수 추가전
#method = "lm" :0.8089203
#method = "rf" :0.9014441

rmse <- function(yi, yhat_i){
  sqrt(mean((yi - yhat_i)^2))
}

rmse(boston_test$price, sample$medv) # 승순이가 만든 파생변수 추가전
# method = "lm" : 5.537822
# method = "rf" : 3.184222

#write.csv(sample, "Submission_sample16.csv", row.names=FALSE)

```

랜덤포레스트:

상관관계 : 0.89

오차 : 3.24

■ 보스턴 하우징 데이터에 파생변수를 추가하고 앙상블 학습을 시 키는 테스트

```
##### 1. 데이터 불러오기 #####
```

```

getwd()
library(data.table)

boston <- read.csv("d:\data\boston.csv" )
nrow(boston) #506

# 0.shuffle 을 먼저 합니다.
set.seed(123)

boston_shuffle <- sample(1:506, 506)
boston2 <- boston[boston_shuffle,]
boston2

# 훈련 데이터 80%, 테스트 데이터 20%
set.seed(1234)
ind <- sample(2, nrow(boston2), replace = T, prob = c(0.8, 0.2))
boston_train <- boston2[ind==1,]
boston_test <- boston2[ind==2,]
nrow(boston_train) # 411
nrow(boston_test) # 95

# 상관관계 분석, vif 확인등의 절차를 거친 후 추가되는 데이터

# 1)

tolower('AAA')
colnames(boston_train) <- tolower(colnames(boston_train))
colnames(boston_test) <- tolower(colnames(boston_test))

str(boston_train)
str(boston_test)

# 하위계층의 비율이 5%보다 크고 주택 1가구당 평균방의 개수가 5개 이하인 것
boston_train$lstat_rm <- ifelse( ( boston_train$lstat >5 & boston_train$rm <= 5) , 1, 0 )
boston_test$lstat_rm <- ifelse( ( boston_test$lstat >5 & boston_test$rm <= 5) , 1, 0 )

# 1940년 이전에 건축된 주택의 소유비율이 30%보다 작고 비소매 상업지구가 점유하고 있는
# 소유비율이 10% 이하인 것은 1, 아니면 0
boston_train$age_indus <- ifelse( ( boston_train$age < 30 & boston_train$indus <= 10) ,
1, 0 )
boston_test$age_indus <- ifelse( ( boston_test$age < 30 & boston_test$indus <= 10) , 1,
0 )

##### 2. 데이터 전처리 #####
##### 2.1 결측치 확인 #####
# 1) 결측치

# 1-1. 결측치 확인
sum(is.na(boston_train)) # 0개

```

```

sum(is.na(boston_test)) #0개

# 1-2. 결측치 위치확인
#(1) 데이터셋명[complete.cases(데이터셋명),] # 결측치가 없으면 True
#(2) 데이터셋명[!complete.cases(데이터셋명),] # 결측치가 있으면 True

# 1-3. 결측치 삭제
#boston_train<-na.omit(boston_train)
#boston_test<-na.omit(boston_test)

# 1-4. 결측치 대체

# (1) 결측할 값 직접지정
#boston_train$대체할변수있는컬럼[is.na(데이터셋$컬럼명)]<-일괄대체할값
# 지정 컬럼에서 na가 있으면 지정하는 값으로 대체

# (2) 통계값으로 대체
#install.packages("DMwR")
#library(DMwR)

#centralImputation(데이터셋명) # NA중앙값 대체 : 숫자의 경우 중앙값, 팩터의 경우 최빈
값
#knnImputation(데이터셋명) # means를 활용한 결측치 대체

# 2) 정규화
#데이터 정규화를 위한 함수생성
normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x) ) )
}

boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

#정규화 결과 확인
summary(boston_train_norm)
summary(boston_test_norm)

#####
##### 3. 모델링, 데이터 학습, 성능검증 , 평가
#####

# 1) 모델링 (변수선택)

# 1-1. 다중공선성 확인
#install.packages("car")
library(car)

```

```
lml <- lm(price~, data=boston_train_norm)
vif(lml) # 팽창지수(=다중공선성=vif)
vif(lml) > 10 #vif 5이상이면 TRUE / 5이하면 FALSE

# strict하게 본다면(vif > 5) indus, rad, tax, age_30, indus_10 의 다중공선성 높음
# 일반적으로 본다면(vif > 10) = 제거해야할 변수 없음
# 이번 실습은 vif > 10을 기준으로 진행
```

```
# 1-2. 분산이 0에 가까운 변수제거
# 데이터의 분산이 0에 가깝다 = 서로 다른 관찰을 구분하는데 소용이 없다. = 제거한다.
# 결과치에서 nzv 컬럼에 TRUE로 뜨는 값 제거
```

```
#install.packages("caret")
library(caret)
nearZeroVar(boston_test_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
nearZeroVar(boston_train_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
```

```
# 2-1. 모델링
```

```
#stepwise + 상관계수 반영한 결과 고려하여 모델링
#boston_reg_model <- lm(price ~ ., data=boston_train_norm )
#boston_reg_model
#summary(boston_reg_model)

#####
# 성능 개선을 위해 추가된 부분

library(caret)
boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "lm")
#boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "rf")
boston_reg_model

#summary(boston_reg_model)

#####
model_results <- predict(boston_reg_model, boston_test_norm)
model_results
```

```
# 2-2. 결과값 역정규화 (kaggle과 데이터 형식 맞추기)
denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data
$medv)
```

```

pred_medv_un <- denormalize(model_results)

sample <- cbind(boston_test$x, pred_medv_un)

head(sample)
colnames(sample) <- c("id", "medv")
sample <- as.data.frame(sample)

boston_test$price
head(sample)

str(boston_test$price)
str(sample)

cor(boston_test$price, sample$medv) # 승순이가 만든 파생변수 추가전
#method = "lm" :0.8089203
#method = "rf" :0.9014441

rmse <- function(yi, yhat_i){
  sqrt(mean((yi - yhat_i)^2))
}

rmse(boston_test$price, sample$medv) # 승순이가 만든 파생변수 추가전
# method = "lm" : 5.537822
# method = "rf" : 3.184222

#write.csv(sample, "Submission_sample16.csv", row.names=FALSE)

```

회귀:

파생변수 추가전 : 상관관계 : 0.8089203, 오차 : 5.537822

파생변수 추가후 : 상관관계 : 0.8272225, 오차 : 5.441164

Istat_rm0이 약간 좋은 파생변수 였다는게 증명이 되었다.

```
##### 1. 데이터 불러오기 #####

```

```

getwd()
library(data.table)

boston <- read.csv("d:\data\boston.csv" )
nrow(boston) #506

#0.shuffle 을 먼저 합니다.
set.seed(123)

boston_shuffle <- sample(1:506, 506)
boston2 <- boston[boston_shuffle,]

```

```
boston2
```

```
# 훈련 데이터 80%, 테스트 데이터 20%
set.seed(1234)
ind <- sample(2, nrow(boston2), replace = T, prob = c(0.8, 0.2))
boston_train <- boston2[ind==1,]
boston_test <- boston2[ind==2,]
nrow(boston_train) # 411
nrow(boston_test) # 95
```

```
#상관관계 분석, vif 확인등의 절차를 거친 후 추가되는 데이터
```

```
# 1)
```

```
tolower('AAA')
colnames(boston_train) <- tolower(colnames(boston_train))
colnames(boston_test) <- tolower(colnames(boston_test))

str(boston_train)
str(boston_test)
```

```
#하위계층의 비율이 5%보다 크고 주택 1가구당 평균방의 개수가 5개 이하인 것
```

```
boston_train$lstat_rm <- ifelse( ( boston_train$lstat >5 & boston_train$rm <= 5) , 1, 0 )
boston_test$lstat_rm <- ifelse( ( boston_test$lstat >5 & boston_test$rm <= 5) , 1, 0 )
```

```
# 1940년 이전에 건축된 주택의 소유비율이 30%보다 작고 비소매 상업지구가 점유하고 있는
```

```
# 소유비율이 10% 이하인 것은 1, 아니면 0
```

```
boston_train$age_indus <- ifelse( ( boston_train$age < 30 & boston_train$indus <= 10) ,
1, 0 )
boston_test$age_indus <- ifelse( ( boston_test$age < 30 & boston_test$indus <= 10) , 1,
0 )
```

```
##### 2. 데이터 전처리 #####
```

```
# 1) 결측치
```

```
# 1-1. 결측치 확인
```

```
sum(is.na(boston_train)) #0개
sum(is.na(boston_test)) #0개
```

```
# 1-2. 결측치 위치확인
```

```
#(1) 데이터셋명[complete.cases(데이터셋명)] # 결측치가 없으면 True
#(2) 데이터셋명[!complete.cases(데이터셋명),] # 결측치가 있으면 True
```

```
# 1-3. 결측치 삭제
```

```
#boston_train<-na.omit(boston_train)
```

```

#boston_test<-na.omit(boston_test)

# 1-4. 결측치 대체

# (1) 결측할 값 직접지정
#boston_train$대체할변수있는컬럼[is.na(데이터셋$컬럼명)]<-일괄대체할값
# 지정 컬럼에서 na가 있으면 지정하는 값으로 대체

# (2) 통계값으로 대체
#install.packages("DMwR")
#library(DMwR)

#centralImputation(데이터셋명) # NA중앙값 대체 : 숫자의 경우 중앙값, 팩터의 경우 최빈
값
#knnImputation(데이터셋명) # means를 활용한 결측치 대체

# 2) 정규화
#데이터 정규화를 위한 함수생성
normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x) ) )
}

boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

#정규화 결과 확인
summary(boston_train_norm)
summary(boston_test_norm)

#####
##### 3. 모델링, 데이터 학습, 성능검증 , 평가 #####
#####

# 1) 모델링 (변수선택)

# 1-1. 다중공선성 확인
#install.packages("car")
library(car)

lml <- lm(price~., data=boston_train_norm)
vif(lml) # 팽창지수(=다중공선성=vif)
vif(lml) > 10 #vif 5이상이면 TRUE / 5이하면 FALSE

# strict하게 본다면(vif > 5) indus, rad, tax, age_30, indus_10 의 다중공선성 높음
# 일반적으로 본다면(vif > 10) = 제거해야할 변수 없음
# 이번 실습은 vif > 10을 기준으로 진행

```

```
# 1-2. 분산이 0에 가까운 변수제거  
# 데이터의 분산이 0에 가깝다 = 서로 다른 관찰을 구분하는데 소용이 없다. = 제거한다.  
# 결과치에서 nzv 컬럼에 TRUE로 뜨는 값 제거
```

```
#install.packages("caret")  
library(caret)  
nearZeroVar(boston_test_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)  
nearZeroVar(boston_train_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
```

2-1. 모델링

```
#stepwise + 상관계수 반영한 결과 고려하여 모델링  
#boston_reg_model <- lm(price ~ ., data=boston_train_norm )  
#boston_reg_model  
#summary(boston_reg_model)  
  
#####  
# 성능 개선을 위해 추가된 부분  
  
library(caret)  
boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "lm")  
#boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "rf")  
boston_reg_model  
  
#summary(boston_reg_model)  
  
#####  
  
model_results <- predict(boston_reg_model, boston_test_norm)  
model_results  
  
  
# 2-2. 결과값 역정규화 (kaggle과 데이터 형식 맞추기)  
denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data  
$medv)  
pred_medv_un <- denormalize(model_results)  
  
sample <- cbind(boston_test$x, pred_medv_un)  
  
head(sample)  
colnames(sample) <- c("id", "medv")  
sample <- as.data.frame(sample)  
  
boston_test$price  
head(sample)  
  
str(boston_test$price)
```

```

str(sample)

cor(boston_test$price, sample$medv) # 승순이가 만든 파생변수 추가전
#method = "lm" :0.8089203
#method = "rf" :0.9014441

rmse <- function(yi, yhat_i){
  sqrt(mean((yi - yhat_i)^2))
}

rmse(boston_test$price, sample$medv) # 승순이가 만든 파생변수 추가전
# method = "lm" : 5.537822
# method = "rf" : 3.184222

#write.csv(sample, "Submission_sample16.csv", row.names=FALSE)

```

랜덤포레스트 :

파생변수 추가전 - 상관관계 : 0.89 오차 : 3.24
 파생변수 추가후 - 상관관계 : 0.89 오차 : 3.19

■ 내가 만든 파생변수가 유의미한 파생변수인지 또는 기존에 있는 독립변수들은 다 중요한 독립변수들인지 혹시 예상값과 편차를 크게 만드는 컬럼이 있는건 아닌지 편하게 확인하는 방법은?

종속변수를 설명할 독립변수를 선택하는 방법에 있어서 가장 많이 쓰는 방법이 단계적 변수 선택 방법이다.

이것을 지원하는 R함수가 step함수이다.

```

##### 1. 데이터 불러오기 #####
getwd()
library(data.table)

boston <- read.csv("d://data//boston.csv" )
nrow(boston)

#0.shuffle 을 먼저 합니다.
set.seed(123)
boston_shuffle <- sample(1:506, 506)
boston2 <- boston[boston_shuffle,]
boston2

# 훈련 데이터 80%, 테스트 데이터 20%

```

```

set.seed(1234)
ind <- sample(2, nrow(boston2), replace = T, prob = c(0.8, 0.2))
boston_train <- boston2[ind==1,]
boston_test <- boston2[ind==2,]
nrow(boston_train) # 411
nrow(boston_test) #95

#상관관계 분석, vif 확인등의 절차를 거친 후 추가되는 데이터

# 1)

tolower('AAA')
colnames(boston_train) <- tolower(colnames(boston_train))
colnames(boston_test) <- tolower(colnames(boston_test))

str(boston_train)
str(boston_test)
boston_train$lstat

boston_train$lstat_rm <- ifelse( ( boston_train$lstat >5 & boston_train$rm <= 5) , 1, 0 )
boston_test$lstat_rm <- ifelse( ( boston_test$lstat >5 & boston_test$rm <= 5) , 1, 0 )

boston_train$age_indus <- ifelse( ( boston_train$age < 30 & boston_train$indus <= 10) , 1, 0 )
boston_test$age_indus <- ifelse( ( boston_test$age < 30 & boston_test$indus <= 10) , 1, 0 )

##### 2. 데이터 전처리 #####
# 1) 결측치

# 1-1. 결측치 확인
sum(is.na(boston_train)) #0개
sum(is.na(boston_test)) #0개

# 1-2. 결측치 위치확인
#(1) 데이터셋명[complete.cases(데이터셋명),] # 결측치가 없으면 True
#(2) 데이터셋명[!complete.cases(데이터셋명),] # 결측치가 있으면 True

# 1-3. 결측치 삭제
#boston_train<-na.omit(boston_train)
#boston_test<-na.omit(boston_test)

# 1-4. 결측치 대체

# (1) 결측할 값 직접지정
#boston_train$대체할변수있는컬럼[is.na(데이터셋$컬럼명)]<-일괄대체할값

```

```

# 지정 컬럼에서 na가 있으면 지정하는 값으로 대체

# (2) 통계값으로 대체
#install.packages("DMwR")
#library(DMwR)

#centralImputation(데이터셋명) # NA중앙값 대체 : 숫자의 경우 중앙값, 팩터의 경우 최빈
값
#knnImputation(데이터셋명) # means를 활용한 결측치 대체

# 2) 정규화
#데이터 정규화를 위한 함수생성
normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x) ) )
}

boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

#정규화 결과 확인
summary(boston_train_norm)
summary(boston_test_norm)

#####
##### 3. 모델링, 데이터 학습, 성능검증 , 평가
#####

# 1) 모델링 (변수선택)

# 1-1. 다중공선성 확인
#install.packages("car")
library(car)

lml <- lm(price~, data=boston_train_norm)
vif(lml) # 팽창지수(=다중공선성=vif)
vif(lml) > 10 #vif 5이상이면 TRUE / 5이하면 FALSE

# strict하게 본다면(vif > 5) indus, rad, tax, age_30, indus_10 의 다중공선성 높음
# 일반적으로 본다면(vif > 10) = 제거해야할 변수 없음
# 이번 실습은 vif > 10을 기준으로 진행

# 1-2. 분산이 0에 가까운 변수제거
# 데이터의 분산이 0에 가깝다 = 서로 다른 관찰을 구분하는데 소용이 없다. = 제거한다.
# 결과치에서 nzv 컬럼에 TRUE로 뜨는값 제거

```

```

#install.packages("caret")
library(caret)
nearZeroVar(boston_test_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
nearZeroVar(boston_train_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)

# nzv = TRUE인 변수들 제거
#boston_train <- boston_train[,-nearZeroVar(boston_train)]
#boston_test <- boston_test[,-nearZeroVar(boston_test)]


# 1-3. 상관관계
#install.packages("corrplot")
library(corrplot)
#기본값은 원형, shade=네모칸, ellipse = 타원(양의상관 오른쪽, 음의상관 왼쪽)
# circle = 원형, number = 수치로 표현
corrplot(cor(boston_train_norm), method = "number")


# 1-4. 변수중요도 확인
# 변수중요도 : "해당 변수가 상대적으로 얼마만큼 종속변수에 영향을 주는가?"

# 랜덤 포레스트 방식을 활용

#install.packages("randomForest")
library(randomForest)
#rf <- randomForest(medv~., data=boston_train_norm)
rf <- randomForest(price~., data=boston_train_norm, importance=TRUE)

#중요도 확인방법 2가지

# 1) 중요도 수치로 확인

varImp(rf)
importance(rf) #IncMSE = 정확도, IncNodePurity = 중요도 = importance

# 2) plot형태로 시각화해서 확인
varImpPlot(rf, main="Varplot of boston_train")


# 1-5. 단계적 회귀분석으로 유효변수골라내기
#1번째시도 (기본적으로 주어진 모든 독립변수 활용하여 모델링)

# vif 확인시 만든 lm과 동일
model1 <- lm(formula=price ~ crim + chas + nox + rm + dis + rad + ptratio + lstat,
data = boston_train_norm)
model1
summary(model1)

#stepwise(단계적 회귀분석 방법)로 가장 유효한 변수 확인하기

```

```
lm2 <- step(model1, direction="both")
lm2
# medv ~ crim + chas + nox + rm + dis + rad + ptratio + lstat

# 결과적 추시가 좋지 않아 단계적 회귀분석을 단독 적용하는 것은 무리라 판단. 상관관계
# 를 감안하여 적용하기로 함
```

```
#2번째 시도 (상관관계 분석을 통해 알아낸 사실 포함하여 적용)
lml2<-step(lml, direction="both")
#medv ~ crim + zn + indus + chas + nox + rm + dis + rad + tax + ptratio + black +
lstat + lstat_rm
```

```
#여기까지 확인한 후에 다시 1번으로 돌아가 컬럼 추가 및 전처리를 수행함
```

2-1. 모델링

```
#stepwise + 상관계수 반영한 결과 고려하여 모델링
boston_reg_model <- lm(price ~ crim + zn + indus + chas + nox + rm + dis + rad + tax
+ ptratio + b + lstat + lstat_rm, data=boston_train_norm )
```

```
model_results <- predict(boston_reg_model, boston_test_norm)
model_results
```

```
# 2-2. 결과값 역정규화 (kaggle과 데이터 형식 맞추기)
denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data
$medv)
pred_medv_un <- denormalize(model_results)

sample <- cbind(boston_test$x, pred_medv_un)

head(sample)
colnames(sample) <- c("id", "medv")
sample <- as.data.frame(sample)

boston_test$price
head(sample)

str(boston_test$price)
str(sample)

cor(boston_test$price, sample$medv) # 0.8290219

rmse <- function(yi, yhat_i){
  sqrt(mean((yi - yhat_i)^2))
}

rmse(boston_test$price, sample$medv) # 5.391207
```

```
write.csv(sample, "Submission_sample16.csv", row.names=FALSE)
```

■ kaggle last competition(보스톤) 랜덤포레스트로 도전해보기

- 회귀분석 : 4.25
- 랜덤포레스트 + 파생변수 : ?

1. 오라클 준비
2. kaggle last competition (보스톤)
3. kaggle ongoing competition (타이타닉) - SQL
 - R
4. kaggle ongoing competition (보스톤) - R

```
##### 1. 데이터 불러오기 #####
```

```
library(data.table)

boston <- read.csv("e:\\data\\boston.csv" )
nrow(boston) #506

#0.shuffle
set.seed(123)
boston_shuffle<- sample(1:506, 506)
boston2 <- boston[boston_shuffle,] #랜덤으로 만든 인덱스 번호를 보스턴 데이터에 입히기
boston2

#summary(boston2$AGE)
hist(boston2$AGE)

#summary(boston2$LSTAT)
#hist(boston2$LSTAT)

#hist(boston2$RM)

# 훈련용 & 테스트용 데이터 나누기 (8:2)
set.seed(1234)
ind <- sample(2, nrow(boston2), replace = T, prob = c(0.8, 0.2))
boston_train <- boston2[ind==1,]
boston_test <- boston2[ind==2,]
nrow(boston_train) #411
nrow(boston_test) #95

#(파이썬에서 가져온 데이터)컬럼이 대문자여서 소문자로 바꿔주는 작업
colnames(boston_train) <- tolower(colnames(boston_train))
colnames(boston_test) <- tolower(colnames(boston_test))
```

```

str(boston_train)
str(boston_test)

#####
##### #상관관계 분석, vif 확인등의 절차를 거친 후 추가되는 파생변수 데이터
# 1) 하위계층 비율이 5%보다 작고 주택 1가구당 평균방의 개수가 5개 이상인 것 =1, 아니면 0
boston_train$lstat_rm <- ifelse( ( boston_train$lstat <5 & boston_train$rm > 5) , 1, 0 )
boston_test$lstat_rm <- ifelse( ( boston_test$lstat <5 & boston_test$rm > 5) , 1, 0 )

# 2)
#boston_train$age_ind <- ifelse( ( boston_train$age < 30 & boston_train$indus <= 10) , 1, 0 )
#boston_test$age_ind <- ifelse( ( boston_test$age < 30 & boston_test$indus <= 10) , 1, 0 )

# 3) 하위계층비율이 5%보다 낮고(상위계층), 1940년 이후에 건축된 건물중 최신 일 수록 비
# 쌀것이다.

# 맞으면 1, 아니면 0
boston_train$lstat_age <- ifelse( ( boston_train$lstat <5 & boston_train$age > 60) , 1, 0 )
boston_test$lstat_age <- ifelse( ( boston_test$lstat <5 & boston_test$age > 60) , 1, 0 )
#####

#####

##### 2. 데이터 전처리 #####
# 1) 결측치

# 1-1. 결측치 확인
sum(is.na(boston_train)) #0개
sum(is.na(boston_test)) #0개

# 1-2. 결측치 위치확인
#(1) 데이터셋명[complete.cases(데이터셋명),] # 결측치가 없으면 True
#(2) 데이터셋명[!complete.cases(데이터셋명),] # 결측치가 있으면 True

# 1-3. 결측치 삭제
#boston_train<-na.omit(boston_train)
#boston_test<-na.omit(boston_test)

# 1-4. 결측치 대체

# (1) 결측할 값 직접지정
#boston_train$대체할변수있는컬럼[is.na(데이터셋$컬럼명)]<-일괄대체할값
# 지정 컬럼에서 na가 있으면 지정하는 값으로 대체

# (2) 통계값으로 대체
#install.packages("DMwR")

```

```

#library(DMwR)

#centralImputation(데이터셋명) # NA중央값 대체 : 숫자의 경우 중앙값, 팩터의 경우 최빈값
#knnImputation(데이터셋명) # means를 활용한 결측치 대체

# 2) 정규화

#데이터 정규화를 위한 함수생성(min/max정규화 함수)
normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x)) )
}

#각각의 데이터에서 첫번째 컬럼인 x만 빼고 정규화
boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

#정규화 결과 확인
summary(boston_train_norm)
summary(boston_test_norm)

#####
##### 3. 모델링, 데이터 학습, 성능검증 , 평가 #####
#####

# 1) 모델링 (변수선택)

# 1-1. 다중공선성 확인
#install.packages("car")
library(car)

lml <- lm(price~., data=boston_train_norm)
vif(lml) # 팽창지수(=다중공선성=vif)
vif(lml) > 10 # vif 10이상이면 TRUE / 10이하면 FALSE

# strict하게 본다면(vif > 5)
# 일반적으로 본다면(vif > 10) <- "현업기준"
# 만약 기준을 넘는 변수가 있다면 '상관관계 확인'을 해봐야 함 (1-3번 참조)

# 1-2. 분산이 0에 가까운 변수제거
# 데이터의 분산이 0에 가깝다 = 서로 다른 관찰을 구분하는데 소용이 없다. = 제거한다.
# 결과치에서 nzv 컬럼에 TRUE로 뜨는 값 제거

#install.packages("caret")
library(caret)

nearZeroVar(boston_test_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)
nearZeroVar(boston_train_norm, saveMetrics = TRUE) # all_FALSE (제거대상 없음)

# nzv = TRUE인 변수들 제거

```

```

#boston_train <- boston_train[,-nearZeroVar(boston_train)]
#boston_test <- boston_test[,-nearZeroVar(boston_test)]

# 1-3. 상관관계
#install.packages("corrplot")
library(corrplot)
#기본값은 원형, shade=네모칸, ellipse = 타원(양의상관 오른쪽, 음의상관 왼쪽)
# circle = 원형, number = 수치로 표현
x11()
corrplot(cor(boston_train_norm), method = "number")

```

```

# 1-4. 변수중요도 확인
# 변수중요도 : "해당 변수가 상대적으로 얼마만큼 종속변수에 영향을 주는가?"
# 랜덤 포레스트 방식을 활용

#install.packages("randomForest")
library(randomForest)
rf <- randomForest(price~, data=boston_train_norm)
rf <- randomForest(price~, data=boston_train_norm, importance=TRUE)
#importance 가 강한것 위주로 추려내기

```

```

#중요도 확인방법 2가지

# 1) 중요도 수치로 확인
varImp(rf)
importance(rf) #IncMSE = 정확도, IncNodePurity = 중요도 = importance

# 2) plot형태로 시각화해서 확인
varImpPlot(rf, main="Varplot of boston_train")

#####
#####

# 1-5. 단계적 회귀분석으로 유효변수골라내기
#1번째시도 (기본적으로 주어진 모든 독립변수 활용하여 모델링)
# vif 확인시 만든 lm과 동일
#model1 <- lm(formula=price ~ crim + chas + nox + rm + dis + rad + ptratio + lstat, data =
#boston_train_norm)
#summary(model1)
#R-squared: 0.7341

#stepwise(단계적 회귀분석 방법)로 가장 유효한 변수 확인하기
#lm2 <- step(model1, direction="both")
# price ~ crim + chas + nox + rm + dis + rad + ptratio + lstat
#lm2<-step(lm1, direction = "both")

# 결과적 수치가 좋지 않아 단계적 회귀분석을 단독 적용하는 것은 무리라 판단. 상관관계를
#감안하여 적용하기로함

```

```

#2번째 시도 (상관관계 분석을 통해 알아낸 사실 포함하여 적용)
#lml2<-step(lml, direction="both")
#price ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio + b + lstat + lstat_rm

#3번째 시도 (상관관계 기반으로 파생변수 생성 1&3)
lml2<-step(lml, direction="both")
#price ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio + b + lstat + lstat_rm + lstat_age

#여기까지 확인한 후에 다시 1번으로 돌아가 컬럼 추가 및 전처리를 수행함
#####
#####

```

2-1. 모델링

```

#####
#####
#stepwise(단계적회귀) + 상관계수 반영한 결과 고려하여 모델링
#boston_reg_model <- lm(price ~ crim + zn + indus + chas + nox + rm + dis + rad + tax + ptratio + b +
lstat + lstat_rm, data=boston_train_norm )
#model_results <- predict(boston_reg_model, boston_test_norm)
#model_results

#이 방법을 사용하지 않고 caret의 train()을 활용하면 더 쉽게 구할 수 있다.
#####
#####

# caret의 train() 함수에서 어떤 모델을 쓸지를 아래의 사이트에서 확인
# https://topepo.github.io/caret/available-models.html
# 세세한 특정 코드를 다 적지 않아도 train함수와 method 지정으로 간단하게 모델링하고 학습시킬 수 있다.
```

```

library(caret)
#boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "lm") #회귀분석 방법을 사용하겠다.
#boston_reg_model <- train(price ~ ., data = boston_train_norm, method = "rf") #랜덤포레스트 방법을 사용하겠다.

#stepwise ver
boston_reg_model <- train(price ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio + b + lstat +
lstat_rm + lstat_age, data = boston_train_norm, method = "lm") #회귀분석 방법을 사용하겠다.
#boston_reg_model <- train(price ~ crim + zn + chas + nox + rm + dis + rad + tax + ptratio + b + lstat +
lstat_rm + lstat_age, data = boston_train_norm, method = "rf") #랜덤포레스트 방법을 사용하겠다.
```

```

boston_reg_model
summary(boston_reg_model)
#파생변수 추가 후 - 회귀 결과
# indus, age, age_ind 는 영향력 없음
# 추가한 lstat_rm은 영향력 있음

#lm (회귀) - 변수 추가전
# RMSE Rsquared MAE
# 0.1035051 0.7133275 0.07364974

#lm (회귀) - 변수 추가 후
# RMSE Rsquared MAE
# 0.1094844 0.7341889 0.07623111

#rf(랜덤포레스트) - 변수 추가전
# The final value used for the model was mtry = 7.
#mtry RMSE Rsquared MAE
#7 0.07874802 0.8640941 0.05302565

#rf(랜덤포레스트) - 변수 추가후
# The final value used for the model was mtry = 8
#mtry RMSE Rsquared MAE
#8 0.07870859 0.8679657 0.05333596

#summary(boston_reg_model)
# 회귀 분석 결과표에서 *가 붙지 않은 독립변수들은 종속변수에 유의미한 영향을 주지 않는
# 변수들(삭제)
# rf 모델에서는 summary를 굳이 볼 필요 없음

# 예측결과 확인
model_results <- predict(boston_reg_model, boston_test_norm)
model_results

# 2-2. 결과값 역정규화 (kaggle과 데이터 형식 맞추기)
denormalize <- function(x) { return (x*45+5) } #max(train_data$medv) #min(train_data$medv)
pred_medv_un <- denormalize(model_results)

sample <- cbind(boston_test$x, pred_medv_un)

head(sample)
colnames(sample) <- c("id", "medv")
sample <- as.data.frame(sample)

boston_test$price
head(sample)

str(boston_test$price)
str(sample)

```

```

# 결과치 확인하기

cor(boston_test$price, sample$medv)
## 상관관계 ##

# method = "lm" :0.8089203 - 파생변수 추가전
# method = "rf" :0.8986854 - 파생변수 추가전

# method = "lm" :0.8272225 - 파생변수 추가후
# method = "rf" :0.8997801 - 파생변수 추가후


rmse <- function(yi, yhat_i){
  sqrt(mean((yi - yhat_i)^2))
}

rmse(boston_test$price, sample$medv)
# method = "lm" : 5.537822 - 파생변수 추가전
# method = "rf" : 3.245476 - 파생변수 추가전

# method = "lm" : 5.441164 - 파생변수 추가후
# method = "rf" : 3.199271 - 파생변수 추가후

#####
#####

#제출용 파일 내보내기
#write.csv(sample, "Submission_sample16.csv", row.names=FALSE)

#실제 정답과 예측 결과 상관관계 확인하기
cor(boston_test$price, sample$medv)

#실제 정답과 예측 결과 오차율 확인하기
# 상관관계가 높다고 오차율이 적다는 뜻은 아니기 때문에 같이 확인해줘야 함

#RMSE (Root Mean Square Error Loss)
# 1) postResample함수사용 - caret 패지키내의 함수 (RMSE, Rsquared, MAE 동시에 구할 수 있음)
# 문법 : postResample(pre=, obs=)
postResample(pre=boston_test$price,obs=sample$medv)

#> postResample(pre=boston_test$price,obs=sample$medv)
# RMSE      Rsquared      MAE
# 5.3912074 0.6872772 3.9753040

# 2)RMSE 함수사용
# RMSE(예측값,실제값)

```

```

RMSE(boston_test$price, sample$medv)
#> RMSE(boston_test$price, sample$medv)
#[1] 5.391207

# 3) RMSE 직접 계산하기
sqrt(mean((boston_test$price-sample$medv)^2))
#> sqrt(mean((boston_test$price-sample$medv)^2))
#[1] 5.391207

# R-squared 구하기
# 1) postResample() 함수 사용
postResample(pre=boston_test$price, obs=sample$medv)

# 2) 직접 계산하기
y_bar=mean((sample$medv))
1-(sum((boston_test$price-sample$medv)^2) / sum((sample$medv-y_bar)^2))
#[1] 0.6546078

#MAE(Mean Absolute Error)
# 1) postResample() 함수 사용
postResample(pre=boston_test$price, obs=sample$medv)

# 2) MAE 함수 사용
MAE(boston_test$price, sample$medv)
#> MAE(boston_test$price, sample$medv)
#[1] 3.975304

##### forecast 함수로 결과치 시각화하기 #####
#install.packages("forecast")
library("forecast")

result_s <- data.frame(id=boston_test$x, actual=boston_test$price, pred=pred_medv_un)
accuracy(result_s$actual, result_s$pred)

#      ME    RMSE    MAE    MPE    MAPE
# Test set -0.9564291 5.391207 3.975304 14.82457 80.92733

```

위의 kaggle의 데이터와 지난번 마지막 문제로 작성했던 회귀 코드를 준비하기!!

■ Kaggle ongoing competition (타이타닉) 랜덤포레스트를 이용한 SQL 도전해보기

랜덤 포레스트

1. 수치예측

2. 분류

<https://www.kaggle.com/c/titanic/data>

1. 데이터를 내려 받는다.
2. 오라클에 접속을 한다. (sql developer 이용)

-- ■ 부록 예제1. Kaggle 상위권에 도전하기1

-- 1. 머신러닝 모델이 학습할 테이블을 생성합니다.

```
DROP TABLE TITANIC;
```

```
CREATE TABLE TITANIC
( PASSENGERID      NUMBER(5),
SURVIVED       NUMBER(5),
PCLASS         NUMBER(5),
NAME           VARCHAR2(100),
SEX             VARCHAR2(20),
AGE             NUMBER(5,2),
SIBSP           NUMBER(5),
PARCH           NUMBER(5),
TICKET          VARCHAR2(20),
FARE             NUMBER(18,5),
CABIN           VARCHAR2(50),
EMBARKED        VARCHAR2(5) );
```

-- 데이터 입력: SQL Developer를 이용해서 train.csv 를 titanic 테이블에 입력합니다.

```
SELECT COUNT(*) FROM TITANIC;
--891
```

-- 2. 머신러닝 모델을 구성하기 위한 환경 설정 테이블을 생성합니다.

```
DROP TABLE SETTINGS_GLM;
```

```
CREATE TABLE SETTINGS_GLM
AS
SELECT *
  FROM TABLE (DBMS_DATA_MINING.GET_DEFAULT_SETTINGS)
 WHERE SETTING_NAME LIKE '%GLM%';
```

```
BEGIN
```

```
  INSERT INTO SETTINGS_GLM
    VALUES (DBMS_DATA_MINING.ALGO_NAME, 'ALGO_RANDOM_FOREST');
```

```
  INSERT INTO SETTINGS_GLM
```

```
    VALUES (DBMS_DATA_MINING.PREP_AUTO, 'ON');
```

-- R의 caret 패키지처럼 자동 튜닝 하겠다

```
INSERT INTO SETTINGS_GLM
```

```
VALUES (
```

```
    DBMS_DATA_MINING.GLMS_REFERENCE_CLASS_NAME,
```

```
    'GLMS RIDGE_REG_DISABLE'); -- 회귀가 아니라 분류이다.
```

```
INSERT INTO SETTINGS_GLM
```

```
VALUES (
```

```
    DBMS_DATA_MINING.ODMS_MISSING_VALUE_TREATMENT,
```

```
    'ODMS MISSING_VALUE_MEAN_MODE'); -- 결측치는 평균값으로 치환하겠다
```

```
COMMIT;
```

```
END;
```

```
/
```

-- Begin ~ End : insert 3문장을 한번에 수행하면서 commit

-- 3. 머신러닝 모델을 삭제합니다.

```
BEGIN
```

```
    DBMS_DATA_MINING.DROP_MODEL( 'MD_CLASSIFICATION_MODEL');
```

```
END;
```

```
/
```

-- 4. 머신러닝 모델을 생성합니다.

```
BEGIN
```

```
    DBMS_DATA_MINING.CREATE_MODEL(
```

```
        model_name      => 'MD_CLASSIFICATION_MODEL', -- 모델이름
```

```
        mining_function => DBMS_DATA_MINING.CLASSIFICATION, -- 분류하겠다
```

```
        data_table_name  => 'TITANIC', --훈련 데이터 테이블명
```

```
        case_id_column_name => 'PASSENGERID', -- 식별자 컬럼
```

```
        target_column_name  => 'SURVIVED', -- 종속변수
```

```
        settings_table_name => 'SETTINGS_GLM'); -- 환경설정 테이블명
```

```
END;
```

```
/
```

-- DBMS_DATA_MINING.REGRESSION : 회귀

-- 5. 머신러닝 모델을 확인합니다.

```
SELECT MODEL_NAME,
```

```
    ALGORITHM,
```

```
    MINING_FUNCTION
```

```
FROM ALL_MINING_MODELS
```

```
WHERE MODEL_NAME = 'MD_CLASSIFICATION_MODEL';
```

```
-- where절에 모델명을 써서 잘 만들어졌는지 확인
```

-- 6. 머신러닝 모델 설정 정보를 확인합니다.

```
SELECT SETTING_NAME, SETTING_VALUE  
FROM ALL_MINING_MODEL_SETTINGS  
WHERE MODEL_NAME = 'MD_CLASSIFICATION_MODEL';
```

-- 7. 테스트 데이터를 저장할 테이블을 생성합니다.

```
DROP TABLE TITANIC_TEST;
```

```
CREATE TABLE TITANIC_TEST  
( PASSENGERID      NUMBER(5),  
  PCLASS           NUMBER(5),  
  NAME            VARCHAR2(100),  
  SEX             VARCHAR2(20),  
  AGE             NUMBER(5,2),  
  SIBSP           NUMBER(5),  
  PARCH           NUMBER(5),  
  TICKET          VARCHAR2(20),  
  FARE            NUMBER(18,5),  
  CABIN           VARCHAR2(50),  
  EMBARKED        VARCHAR2(5) );
```

```
-- 데이터 입력: SQL Developer를 이용해서 test.csv 를 titanic_test 테이블에 입력합니다.
```

```
SELECT COUNT(*) FROM TITANIC_TEST;  
--418
```

-- 8. 모델이 예측한 결과를 확인합니다.

```
SELECT passengerid || ',' || -- 이렇게 바꾼 후 csv파일로 저장  
       PREDICTION (MD_CLASSIFICATION_MODEL USING *) MODEL_PREDICT_RESPONSE  
  FROM titanic_test  
 ORDER BY passengerid;
```

```
-- 예측 결과 확인 후에 나온 데이터를 ctrl + a로 전체 선택해서 메모장에 붙여넣고
```

```
-- gender_submission 에 있는 컬럼명을 위에 붙여넣고 csv파일로 저장
```

--■ 부록_예제_2_상위4%_진입 예제

```
-- 정확도를 올리기 위한 3가지 tip
```

- 1. 결측치를 다른 값으로 잘 치환해준다.
 - 호칭으로 나이를 예상해서 나이의 결측치를 채워 넣음
 - 나이에 대한 회귀 예측값으로 결측치를 채워넣음
- 2. 이상치 제거 또는 치환
- 3. 파생변수 생성(feature engineering)
 - a. 중요 파생변수 : "여자와 아이"
 - 성별이 여자이면서 나이가 어리면 1 아니면 0
 - b. 호칭으로 나이를 예상해서 호칭의 평균나이로 나이 데이터를 생성

--1. 여자와 아이를 구분하기 위한 파생 변수를 생성할 컬럼을 추가합니다.

```
ALTER TABLE TITANIC
ADD WOMEN_CHILD NUMBER(5);
```

--2. 추가한 파생변수에 여자와 10세미만의 아이들은 값을 1로 갱신하고 아니면 0으로 값을 갱신합니다.

```
UPDATE TITANIC T1
SET WOMEN_CHILD = ( SELECT CASE WHEN AGE < 10 OR SEX='FEMALE'
                           THEN 1 ELSE 0 END
                      FROM TITANIC T2
                     WHERE T2.PASSENGERID = T1.PASSENGERID );
COMMIT;
```

--3. 테스트 테이블에도 똑같은 컬럼을 추가합니다.

```
ALTER TABLE TITANIC_TEST
ADD WOMEN_CHILD NUMBER(5);
```

--4. 추가한 파생변수에 여자와 10세미만의 아이들은 값을 1로 아니면 0으로 값을 갱신합니다.

```
UPDATE TITANIC_TEST T1
SET WOMEN_CHILD = ( SELECT CASE WHEN AGE < 10 OR SEX='FEMALE' THEN 1
                           ELSE 0 END
                      FROM TITANIC_TEST T2
                     WHERE T2.PASSENGERID = T1.PASSENGERID );
COMMIT;
```

--5. 1과 0의 빈도수를 확인합니다.

```
SELECT WOMEN_CHILD, COUNT(*)
FROM TITANIC
GROUP BY WOMEN_CHILD;
```

--6. 나이(AGE)의 결측치를 확인합니다.

```
SELECT COUNT(*) FROM TITANIC WHERE AGE IS NULL;
```

--7. 나이의 결측치를 이름의 호칭의 평균값으로 채우기 위해 이름에서 호칭만 출력합니다.

```
SELECT name, SUBSTR(name, (instr(name, ',')+1), instr(name, '.')-instr(name, ',')-1) as 호칭  
FROM titanic;
```

--8. titanic 테이블에 title (호칭) 컬럼을 추가합니다.

```
ALTER TABLE TITANIC  
ADD TITLE VARCHAR2(20);
```

--9. titanic 테이블에 추가한 title(호칭) 컬럼을 호칭 값으로 갱신합니다.

```
MERGE INTO TITANIC T  
USING ( SELECT PASSENGERID, NAME,  
        SUBSTR( NAME, INSTR( NAME, ',')+2,  
               INSTR( NAME, '.')-INSTR( NAME, ',')-2 ) AS 호칭  
      FROM TITANIC ) A  
ON ( T.PASSENGERID = A.PASSENGERID )  
WHEN MATCHED THEN  
UPDATE SET T.TITLE = A.호칭 ;
```

```
COMMIT;
```

--10. title(호칭) 컬럼의 값이 잘 변경되었는지 확인합니다.

```
SELECT name, title FROM titanic;
```

--11. titanic 테이블에 title2 (호칭2) 컬럼을 추가합니다.

```
ALTER TABLE TITANIC  
ADD TITLE2 VARCHAR2(20);
```

--12. 변경전 호칭을 변경후 호칭으로 대체한 쿼리문을 실행합니다.

```
SELECT title,  
      CASE WHEN title in ('Mlle','Mme','Ms') then 'Miss'  
            WHEN title in ('Dr','Major','Rev','Sir','Don','Master','Capt') then 'Mr'  
            WHEN title in('Lady','the Countess') then 'Mrs'  
            WHEN title in ('Jonkheer','Col') then 'Other'  
            ELSE title END 호칭2  
FROM titanic;
```

--13. title2(호칭2) 컬럼을 변경 후 호칭으로 변경합니다.

```
MERGE INTO titanic t  
USING titanic i
```

```

On (t.passengerid = i.passengerid)
WHEN MATCHED Then
UPDATE SET title2 = CASE WHEN title in ('Mlle','Mme','Ms') then 'Miss'
                        WHEN title in ('Dr','Major','Capt', 'Sir' , 'Don' , 'Master')
                        THEN 'Mr'
                        WHEN title in ('the Countess', 'Lady') then 'Mrs'
                        WHEN title in ('Jonkheer', 'Col' , 'Rev') then 'Other'
                        ELSE title END;

COMMIT;

```

--14. title2 를 출력하고 title2 별 평균나이를 출력합니다.

```

SELECT title2, round(avg(age))
  FROM titanic
 GROUP BY title2;

```

--15. title2 별로 age의 null 값이 몇 개가 있는지 카운트하여 확인합니다

```

SELECT title2 ,count(*)
  FROM titanic
 WHERE age is null
 GROUP BY title2;

```

--16. 나이의 결측치를 해당 title2 의 평균 나이로 값을 갱신해서 채워 넣습니다.

```

MERGE INTO titanic t
USING ( SELECT title2, round(avg(age)) 평균나이
        FROM titanic
        GROUP BY title2 ) tt
ON ( t.title2 = tt.title2 )
WHEN MATCHED THEN
UPDATE SET t.age = tt.평균나이
WHERE t.age is null ;

COMMIT;

```

--17. titanic_test 테이블에 title (호칭) 컬럼을 추가합니다.

```

ALTER TABLE TITANIC_TEST
ADD TITLE VARCHAR2(20);

```

--18. 추가한 title 컬럼에 호칭을 갱신합니다.

```

MERGE INTO titanic_test t
USING ( SELECT passengerid, name,
            SUBSTR( name, instr( name, ',')+2,
                    instr( name, '.')-instr( name, ',')-2 ) as 호칭
        FROM titanic_test ) a
ON ( t.passengerid = a.passengerid )
WHEN MATCHED THEN

```

```
UPDATE SET t.title = a.호칭 ;
```

```
COMMIT;
```

--19. titanic_test 테이블에 title2 컬럼을 추가합니다.

```
ALTER TABLE TITANIC_TEST  
ADD TITLE2 VARCHAR2(20);
```

--20. title2(호칭2) 컬럼을 변경 후 호칭으로 변경합니다.

```
MERGE INTO titanic_test t  
USING titanic_test i  
ON (t.passengerid = i.passengerid)  
WHEN MATCHED THEN  
UPDATE SET title2 = CASE WHEN title in ('Mlle','Mme','Ms') then 'Miss'  
WHEN title in ('Dr','Major','Capt', 'Sir' , 'Don' , 'Master')  
THEN 'Mr'  
WHEN title in ('the Countess', 'Lady') then 'Mrs'  
WHEN title in ('Jonkheer', 'Col' , 'Rev') then 'Other'  
ELSE title END;
```

```
COMMIT;
```

--21. 나이의 결측치를 해당 title2 의 평균 나이로 값을 갱신하여 결측치를 채웁니다.

```
MERGE INTO titanic_test t  
USING ( SELECT title2, round(avg(age)) 평균나이  
FROM titanic_test  
GROUP BY title2 ) tt  
ON ( t.title2 = tt.title2 )  
WHEN MATCHED THEN  
UPDATE SET t.age = tt.평균나이  
WHERE t.age is null ;
```

```
COMMIT;
```

--22. 운임의 이상치(Outlier)를 확인합니다.

```
SELECT PASSENGERID, FARE, 이상치기준  
FROM( SELECT T.*,  
ROUND( AVG(FARE) OVER () + 5 * STDDEV(FARE) OVER () ) "이상치기준"  
FROM TITANIC T  
)  
WHERE FARE > 이상치기준;
```

--23. 운임의 이상치를 제거하고 훈련 데이터를 구성하기 위해 VIEW를 생성합니다.

```
CREATE OR REPLACE VIEW TT_VIEW
```

```

AS
SELECT *
  FROM( SELECT T.*,
              ROUND( AVG(FARE) OVER () + 5 * STDDEV(FARE) OVER ()) "이상치기준"
        FROM TITANIC T
      )
 WHERE FARE < 이상치기준;

```

--24 머신러닝 모델을 구성하기 위한 환경 설정 테이블을 생성합니다.

```
DROP TABLE SETTINGS_GLM3;
```

```

CREATE TABLE SETTINGS_GLM3
AS
SELECT *
  FROM TABLE (DBMS_DATA_MINING.GET_DEFAULT_SETTINGS)
 WHERE SETTING_NAME LIKE '%GLM%';

```

```
BEGIN
```

```

  INSERT INTO SETTINGS_GLM3
    VALUES (DBMS_DATA_MINING.ALGO_NAME, 'ALGO_RANDOM_FOREST');

```

```

  INSERT INTO SETTINGS_GLM3
    VALUES (DBMS_DATA_MINING.PREP_AUTO, 'ON');

```

```

  INSERT INTO SETTINGS_GLM3
    VALUES (DBMS_DATA_MINING.CLAS_MAX_SUP_BINS, 254);

```

```
  COMMIT;
```

```
END;
/
```

--25. 기존의 머신러닝 모델을 삭제합니다.

```

BEGIN
  DBMS_DATA_MINING.DROP_MODEL( 'MD_CLASSIFICATION_MODEL3');
END;
/

```

--26. 머신러닝 모델을 생성합니다.

```

BEGIN
  DBMS_DATA_MINING.CREATE_MODEL(
    MODEL_NAME      => 'MD_CLASSIFICATION_MODEL3',
    MINING_FUNCTION => DBMS_DATA_MINING.CLASSIFICATION,
    DATA_TABLE_NAME  => 'TT_VIEW',
    CASE_ID_COLUMN_NAME => 'PASSENGERID',
    TARGET_COLUMN_NAME  => 'SURVIVED',
    SETTINGS_TABLE_NAME => 'SETTINGS_GLM3');
END;

```

/

--27. 머신러닝 모델을 확인합니다.

```
SELECT MODEL_NAME,  
       ALGORITHM,  
       MINING_FUNCTION  
  FROM ALL_MINING_MODELS  
 WHERE MODEL_NAME = 'MD_CLASSIFICATION_MODEL3';
```

--28. 테스트 데이터로 예측 값을 확인합니다.

```
SELECT PASSENGERID || ',' ||  
       PREDICTION (MD_CLASSIFICATION_MODEL3 USING *)  
  MODEL_PREDICT_RESPONSE  
  FROM TITANIC_TEST  
 ORDER BY PASSENGERID;
```

--29. 케글 Submit Predictions 템을 클릭합니다.

--30. Step1에 gender_submission.csv 파일을 올립니다.

오늘의 문제(1-

2021년 1월 14일 목요일 오후 4:47

문제1. (오늘의 마지막 문제 1/14) centos7에 R과 Rstudio를 설치하고 putty에서 R studio 실행할 때만 아래와 같이 하면 됩니다.

<https://flipdata.tistory.com/35> 참고, root에서 수행하기

```
[root@centos ~]# export DISPLAY=192.168.19.3:0.0
```

```
[root@centos ~]# firefox http://localhost:8787
```

```
Running without a11y support!
```

```
setwd("/home/scott")
emp <- read.csv("emp2.csv")
emp
```

RStudio Server - Mozilla Firefox@centos

RStudio Server X +

localhost:8787

File Edit Code View Plots Session Build Debug Proj

R Go to file/function Addins

Console Terminal x Jobs x

~/

```
R version 3.6.0 (2019-04-26) -- "Planting of a Tree"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-redhat-linux-gnu (64-bit)

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

Natural language support but running in an English locale

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

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

> setwd("/home/scott")
> emp <- read.csv("emp2.csv")
> emp
   X7839    KING PRESIDENT   X0 X1981.11.17 X5000 X0.1 X10
1  7698  BLAKE    MANAGER 7839  1981-05-01  2850     0   30
2  7782  CLARK    MANAGER 7839  1981-05-09  2450     0   10
3  7566  JONES    MANAGER 7839  1981-04-01  2975     0   20
4  7654 MARTIN  SALESMAN 7698  1981-09-10  1250  1400   30
5  7499  ALLEN  SALESMAN 7698  1981-02-11  1600    300   30
6  7844 TURNER  SALESMAN 7698  1981-08-21  1500     0   30
7  7900  JAMES    CLERK 7698  1981-12-11   950     0   30
8  7521  WARD    SALESMAN 7698  1981-02-23  1250    500   30
9  7902  FORD    ANALYST 7566  1981-12-11  3000     0   20
10 7369  SMITH    CLERK 7902  1980-12-09   800     0   20
11 7788 SCOTT    ANALYST 7566  1982-12-22  3000     0   20
12 7876 ADAMS    CLERK 7788  1983-01-15  1100     0   20
13 7934 MILLER   CLERK 7782  1982-01-11  1300     0   10
>
```

문제2. emp.csv를 리눅스 /home/scott에 올리고 emp.csv를 로드해서 emp 데이터 프레임을 만드시오.

(윈도우에서 할 때)

```
setwd("d:\##data")
emp <- read.csv("emp3.csv")
```

(리눅스에서 할 때)

```
setwd("/home/scott")
```

```
emp <- read.csv("emp3.csv")
emp
```

문제3. emp 데이터 프레임에서 이름과 월급을 출력하시오!

```
emp[행, 열]
emp[, c("ename", "sal")]
```

설명 : c : combine의 약자 / 행이 다 나와야 해서 빈칸임

문제4. 월급이 3000 사원들의 이름과 월급을 출력하시오!

```
emp[emp$sal==3000, c("ename", "sal")]
```

문제5. 월급이 2000이상인 사원들의 이름과 월급을 출력하시오!

```
emp[emp$sal>=2000, c("ename", "sal")]
```

문제6. 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하시오!

```
emp[emp$job=='SALESMAN', c("ename", "sal", "job")]
```

문제7. 1981년 12월 11일에 입사한 사원들의 이름과 입사일을 출력하시오!

```
emp[emp$hiredate=='1981-12-11 0:00', c("ename", "hiredate")]
```

문제8. emp 데이터 프레임의 구조를 확인하시오!

```
str(emp)
```

결과:

```
'data.frame': 14 obs. of 9 variables:
 $ index  : int 1 2 3 4 5 6 7 8 9 10 ...
 $ empno   : int 7839 7698 7782 7566 7654 7499 7844 7900 7521 7902 ...
 $ ename    : chr "KING" "BLAKE" "CLARK" "JONES" ...
 $ job      : chr "PRESIDENT" "MANAGER" "MANAGER" "MANAGER" ...
 $ mgr      : int NA 7839 7839 7839 7698 7698 7698 7698 7566 ...
 $ hiredate: chr "1981-11-17 0:00" "1981-05-01 0:00" "1981-05-09 0:00" "1981-04-01 0:00" ...
 $ sal      : int 5000 2850 2450 2975 1250 1600 1500 950 1250 3000 ...
 $ comm     : int NA NA NA NA 1400 300 0 NA 500 NA ...
 $ deptno   : int 10 30 10 20 30 30 30 30 30 20 ...
```

설명 : emp 데이터 프레임의 hiredate가 chr(문자형)이므로 검색할 때 문자 그대로 검색하면 됩니다.

문제9. 아래의 SQL을 R로 구현하시오!

```
SQL> select ename || '의 직업은' || job
      from emp;
```

답 : paste(emp\$ename, '의 직업은', emp\$job)

문제10. 직업이 SALESMAN, ANALYST인 사원들의 이름과 월급과 직업을 출력하시오

문법: emp[행, 열]

답: emp[emp\$job %in% c("SALESMAN", "ANALYST"), c("ename", "sal", "job")]

문제11. 직업이 SALESMAN, ANALYST가 아닌 사원들의 이름과 월급과 직업을 출력하시오!

답: emp[!emp\$job %in% c("SALESMAN", "ANALYST"), c("ename", "sal", "job")]

문제12. 부서번호가 10번, 20번인 사원들의 이름과 월급과 부서번호를 출력하시오!

답 : emp[emp\$deptno %in% c(10, 20), c("ename", "sal", "deptno")]

문제13. 커미션이 null인 사원들의 이름과 월급과 커미션을 출력하시오!

답: emp[is.na(emp\$comm), c("ename", "sal", "comm")]

설명 :

1. NA (결손값) -----> is.na()
2. NaN (비수치) -----> is.nan()
(Not a Number)
3. NULL (아무것도 없다) -----> is.null

머신러닝 학습시에는 결손값을 다른 값으로 대체하는것이 아주 중요합니다.

문제14. 커미션이 NA가 아닌 사원들의 이름과 월급과 커미션을 출력하시오!

답 : emp[! is.na(emp\$comm), c("ename", "sal", "comm")]

문제15. 월급이 1000에서 3000 사이인 사원들의 이름과 월급을 출력하시오!

SQL > SELECT ename, sal
from emp
where sal between 1000 and 3000;

답: emp[emp\$sal >=1000 & emp\$sal<= 3000, c("ename", "sal")]

➤ str(emp\$sal)

해보면

int [1:14] 5000 2850 2450 2975 1250 1600 1500 950 1250 3000 ...

딱 한개만 뜯다.

문제16. 월급이 1000에서 3000 사이가 아닌 사원들의 이름과 월급을 출력하시오!

답 : `emp[!(emp$sal >= 1000 & emp$sal <= 3000), c("ename", "sal")]`

! 다음에 나오는 행에 괄호 안써주면 결과가 제대로 나오지 않는다!! 유의할 것

문제17. 이름의 첫번째 철자가 A로 시작하는 사원들의 이름과 월급을 출력하시오!

답 : `emp[grep("^[A]", emp$ename), c("ename", "sal")]`

설명 : ^는 시작을 나타냅니다.

문제18. 이름의 끝글자가 T로 끝나는 사원들의 이름과 월급을 출력하시오!

`emp[grep("T$", emp$ename), c("ename", "sal")]`

설명 : \$는 끝을 나타냅니다.

문제19. 이름의 두번째 철자가 M인 사원들의 이름과 월급을 출력하시오!

답 : `emp[grep("^.M", emp$ename), c("ename", "sal")]`

설명 : 점(.)은 자릿수 하나입니다.

문제20. (점심시간 문제) 이름의 세번째 철자가 L인 사원들의 이름과 월급을 출력하시오!

답 : `emp[grep("^.L", emp$ename), c("ename", "sal")]`

문제21. 부서번호를 출력하는데 중복을 제거해서 출력하시오.

```
library(data.table)
> data.table(unique(emp$deptno))
```

문제22. 직업을 출력하는데 중복을 제거해서 출력하시오!

```
library(data.table)
data.table("직업"=unique(emp$job) )
```

문제23. 이름과 월급을 출력하는데 월급이 높은 사원부터 출력하시오!

답 : `emp[order(emp$sal, decreasing=T), c("ename", "sal")]`

설명: `dataframe`에 내장되어 있는 `order` 옵션을 사용하면 됩니다.

문제24. 이름과 입사일을 출력하는데 먼저 입사한 사원부터 출력하시오!

답 : `emp[order(emp$hiredate, decreasing=F), c("ename", "hiredate")]`

문제25. 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하는데 월급이 높은 사원부터 출력하시오!

답 :

1. 직업이 SALESMAN인 사원들의 이름과 월급을 출력해서 result 변수에 담는다.

```
result <- emp[ emp$job=='SALESMAN', c("ename", "sal", "job")]
```

str(result) 만 따로보면

결과:

```
'data.frame': 4 obs. of 2 variables:  
 $ ename: Factor w/ 14 levels "ADAMS","ALLEN",..: 9 2 13 14  
 $ sal : int 1250 1600 1500 1250
```

2. result 변수의 월급을 높은 것부터 출력한다.

```
result[ order(result$sal, decreasing=T), c("ename", "sal", "job") ]
```

문제26. 부서번호가 30번인 사원들의 이름과 월급과 입사일을 출력하는데 먼저 입사한 사원부터 출력하시오!

답: result1 <- emp[emp\$deptno==30, c("ename", "sal", "hiredate")]
result1[order(result1\$hiredate, decreasing=F),]

문제27. doBy 패키지를 이용해서 이름과 월급을 출력하는데 월급이 높은 순서대로 출력하시오!

```
install.packages("doBy")  
library(doBy)  
orderBy(~-sal, emp[ , c("ename", "sal")])
```

설명 : emp[, c("ename", "sal")] 이름과 월급을 출력하는 결과를 orderBy 함수에 넣고 정렬하고자 하는 컬럼 sal 앞에 ~를 사용해서 정렬을 하면 됩니다.

물결 다음에 마이너스(-)를 붙여주면 높은 것부터 정렬됩니다.

문제28. 직업이 ANALYST가 아닌 사원들의 이름과 월급과 직업을 출력하는데 월급이 높은 사원부터 출력되게하시오! (doBy 패키지를 이용해서 하세요~)

```
library(doBy)  
orderBy(~-sal, emp[emp$job != 'ANALYST', c("ename", "sal", "job")])
```

문제29. 범죄 발생요일(crime_day.csv)를 R로 로드해서 토요일에 발생하는 범죄유형, 범죄건수를 출력하는데 범죄건수가 높은 것부터 출력하시오!

데이터 게시판 43에 crime_day.csv를 다운받으시오!

```
crime_day <- read.csv("crime_day.csv")
result <- crime.day[ crime_day$DAY=='SAT', c("C_T", "CNT") ]
library(doBy)
orderBy(~-CNT, result)
```

**문제30. 살인이 일어나는 장소와 건수를 출력하는데 건수가 높은것부터 출력하시오.
(crime_loc.csv 를 R로 로드하세요~)**

```
crime_loc <- read.csv("crime_loc.csv")
head(crime_loc)
result <- crime_loc[ crime_loc$범죄=='살인', c("장소", "건수") ]
library(doBy)
orderBy(~-건수, result)
```

문제31. 강도가 일어나는 장소와 건수를 출력하는데 건수가 높은것부터 출력하시오!

```
crime_loc <- read.csv("crime_loc.csv")
head(crime_loc)
result <- crime_loc[ crime_loc$범죄=='강도', c("장소", "건수") ]
library(doBy)
orderBy(~-건수, result)
```

문제32. 이름과 직업을 출력하는데 소문자로 출력하시오!

```
library(data.table)
data.table(이름=tolower(emp$ename), 직업=tolower(emp$job) )
```

문제33. 이름을 출력하고 그 옆에 이름의 첫번째 철자부터 세번째 철자까지 출력되게 하시오.

```
SQL > select ename, substr(ename, 1, 3)
      from emp;
```

```
R > library(data.table)
    data.table( 이름=emp$ename, 철자=substr(emp$ename,1,3) )
```

설명 : substr(변수, 시작, 끝)

문제34. 이름, 월급을 출력하는데 월급을 출력할 때에 숫자 0을 *로 출력하시오!

```
SQL> select, ename, replace( sal, 0, '*' )
      from emp;
```

```
R > data.table( emp$ename, gsub(0, '*', emp$sal) )
```

설명 : gsub(변경 전, 변경 후, 변수)

문제35. 이름, 월급을 출력하는데 월급을 출력할 때에 숫자 0, 1, 2를 *로 출력하시오!

```
SQL> select ename, regexp_replace( sal, '[0-2]', '*' )
      from emp;
```

```
R > data.table(이름=emp$ename, 월급=gsub('[0-2]', '*', emp$sal) )
```

문제36. 6의 9승을 출력하시오!

답 : 6^9

문제37. 10을 3으로 나눈 나머지값을 출력하시오!

답 : $10\%3$

문제38. 이름과 연봉을 출력하는데 연봉은 월급의 12를 곱해서 출력하고 컬럼명은 둘다 한글로 이름, 연봉이라고 출력되게하시오!

답 : data.table(이름=emp\$ename, 연봉=emp\$sal * 12)

문제39. 위의 결과를 다시 출력하는데 round 함수를 써서 백의 자리에서 반올림되게 하시오! (35700 ---> 36000)

답 : data.table(이름=emp\$ename, 연봉=round(emp\$sal * 12,-3))

3	5	7	0	0	.	7	8	8	
자릿수	-5	-4	-3	-2	-1	0	1	2	3

문제40. 위의 결과는 반올림을 한 것이고 반올림을 하지 않고 백자리 이후를 다 버려서 출력하시오!

답 : trunc는 소수점 이하만 버릴 수 있다. 소수점 이전은 못 버리므로 수행이 안된다.

data.table(이름=emp\$ename, 연봉=trunc(emp\$sal * 12,-3))

※ 설명 : R은 짹수를 좋아한다.

round(122.5) -----> 122

round(123.5) -----> 124

딱 중간에 있으면 짹수를 선택한다.

round(122.6) -----> 123

문제41. 오늘날짜를 출력하시오!

Sys.Date()

문제42. 이름, 입사한 날짜부터 오늘까지 총 몇일 근무했는지 출력하시오!

(힌트 : 오라클(to_date) = R (as.Date))

```
SQL > select ename, sysdate - hiredate  
      from emp;
```

```
R> data.table( emp$ename, Sys.Date() - emp$hiredate )  
    날짜      문자
```

날짜에서 문자를 뺄 수는 없다.

따라서 정답은:

```
R> data.table( emp$ename, Sys.Date() - as.Date(emp$hiredate) )
```

문제43. 오늘 날짜의 달의 마지막 날짜를 출력하시오!

```
SQL> select last_day( sysdate )  
      from dual;
```

```
R> install.packages("lubridate")  
library(lubridate)  
ceiling_date( Sys.Date(), "months" ) - days(1)
```

설명 : 오늘의 날짜의 달의 천장 날짜 (다음달 1일의 날짜)

따라서 ceiling_date의 날에서 하루를 빼면 이번달 마지막날의 날짜를 출력함.

floor_date(Sys.Date(), "months") -----> 2021-01-01

Sys.Date() -----> 2021-01-18

ceiling_date(Sys.Date(), "months") - days(1)

문제45. 11월에 입사한 사원들의 이름과 입사일을 출력하시오!

```
SQL> select ename, hiredate  
      from emp  
     where to_char(hiredate, 'mm') = '11';
```

```
R> emp[ format( as.Date(emp$hiredate), '%m' ) == '11', c("ename", "hiredate") ]
```

문제46. 오늘부터 100달 뒤에 돌아오는 날짜를 출력하시오!

```
SQL> select add_months( sysdate, 100 )
```

```
R > Sys.Date() + months(100)
```

설명 : days(숫자), months(숫자), years(숫자)

문제47. 오늘부터 100달뒤에 돌아오는 날짜의 요일을 출력하시오!

```
SQL> select to_char(add_months(sysdate, 100), 'day')  
      from dual;
```

```
R > format(Sys.Date() + months(100), '%A')
```

문제48. 내가 무슨요일에 태어났는지 출력하시오!

```
SQL> select to_char(to_date('1994/06/30', 'RRRR/MM/DD'), 'day')  
      from dual;
```

```
R> format(as.Date('1994/06/30'), '%A')
```

문제49. 이름, 월급, 등급을 출력하는데 월급이 1500이상이면 등급을 A라고 출력하고 아니면 B로 출력하시오!

```
R> data.table( emp$ename, emp$sal, ifelse(emp$sal >= 1500, 'A', 'B') )
```

설명 : ifelse는 결측치 데이터를 다른 데이터로 대체할 때 아주 유용한 함수.

머신러닝 이용시 모델이 학습할 때 좋은 학습 데이터를 제공하기 위하여 결측치에 값을 채워주는 것이 중요.

문제50. 이름, 월급, 보너스를 출력하는데 보너스가 입사한 년도가 1980년도이면 A로 출력하고

1981년도이면 B로 출력하고

1982년도이면 C로 출력하고

나머지 년도는 D로 출력되게 하시오!

```
data.table( emp$ename, emp$sal,
            ifelse( format(as.Date(emp$hiredate), '%Y')=='1980', 'A',
            ifelse( format(as.Date(emp$hiredate), '%Y')=='1981', 'B',
            ifelse( format(as.Date(emp$hiredate), '%Y')=='1982', 'C', 'D') ) ) )
```

문제51. (오늘의 마지막 문제 1/18) 신성이가 올린 리눅스 하둡 포트폴리오의 케글 데이터 중 student-mat.csv를 내려받아서 R로 로드하고 나이와 등급을 출력하는데 등급이 나이가 15 ~ 18은 A, 19~20은 B, 21~22은 C로 출력하시오!

```
setwd("d:WWWdata")
```

```
stu <- read.csv("student-mat.csv")
data.table( 나이=stu$age, 등급=
            ifelse( stu$age>=15 & stu$age<=18, 'A',
            ifelse( stu$age>=19 & stu$age<=20, 'B', 'C') ) )
```

문제52. 최대월급을 출력하시오!

1. 워킹 디렉토리를 지정한다.

```
setwd("d:WWWdata")
```

2. 지정된 워킹 디렉토리를 확인한다.

```
getwd()
```

#3. emp.csv를 로드하여 emp 데이터 프레임을 만든다.

```
emp <- read.csv("emp3.csv")
```

#4. 최대월급을 구한다.

```
max(emp$sal)
```

결과 : 5000

문제53. 직업이 SALESMAN인 사원들의 최대월급을 출력하시오!

두번에 나눠서 하는 방법:

```
> result <- emp[emp$job=='SALESMAN', "sal"]  
> result  
[1] 1250 1600 1500 1250  
> max(result)  
[1] 1600
```

한번에 하는 방법:

```
max( emp[emp$job=="SALESMAN", "sal"] )
```

문제54. 20번 부서번호를 가진 사원들 중에서의 최소월급을 출력하시오!

```
min( emp[emp$deptno==20, "sal"] )
```

또는

```
result <- emp[ emp$deptno==20, "sal" ]  
min(result)
```

문제55. 직업, 직업별 최대월급을 출력하시오!

```
SQL> select job, max(sal)  
      from emp  
      group by job;
```

```
R> aggregate( sal~job, emp, max )
```

설명 : aggregate(계산될 컬럼 ~ 기준될 컬럼, 테이블명, 그룹함수명)

 물결뒤의 컬럼명 별로 물결 앞에 컬럼을 나눠줌

문제56. 부서번호, 부서번호별 토탈월급을 출력하시오

```
SQL> select deptno, sum(sal)  
      from emp  
      group by deptno;
```

```
R> aggregate( sal~ deptno, emp, sum )
```

문제57. 위에서 출력되고 있는 컬럼명을 한글로 부서번호, 토탈월급으로 변경하시오!

```
> result <- aggregate( sal~ deptno, emp, sum )  
> names(result) <- c('부서번호', '토탈월급')  
> result
```

문제58. 위의 결과를 다시 출력하는데 토탈월급이 높은것부터 출력하시오!

```
> library(dplyr)  
> orderBy(~-토탈월급, result)
```

문제59. 직업, 직업별 인원수를 출력하시오!

```
SQL> select job, count(empno)  
      from emp  
      group by job;
```

답:

```
R> aggregate( empno ~ job, emp, length )
R> names(result) <- c("직업", "인원수")
R> result
```

문제60. 위의 결과를 다시 출력하는데 인원수가 높은것부터 출력하시오!

```
orderBy(~-인원수, result)
```

문제61. 직업, 직업별 인원수를 가로로 출력하시오!

```
> table(emp$job)
```

결과:

ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN
2	4	3	1	4

문제62. 문제61번의 결과를 원형 그래프로 시각화 하시오.

```
pie(table(emp$job), col=rainbow(5), density = 100)
```

문제63. 부서번호, 직업, 부서번호별 직업별 토탈 월급을 출력하시오!

```
SQL> select deptno, job, sum(sal)
      from emp
     group by deptno job;
```

```
R> aggregate( sal ~ deptno+job, emp, sum )
```

문제64. 입사한 년도(4자리), 입사한 년도별 평균월급을 출력하시오!

```
SQL> select to_char(hiredate, 'RRRR'), avg(sal)
      from emp
     group by to_char(hiredate, 'RRRR');
```

```
R> aggregate( sal ~ format(as.Date(emp$hiredate), '%Y'), emp, mean)
```

```
R> x <- aggregate( sal ~ format(as.Date(emp$hiredate), '%Y'), emp, mean)
> names(x) <- c("입사한 년도", "평균월급")
> x
```

문제65. 위의 결과에서 소수점 이하는 안나오게 하시오

```
R> x$평균월급 <- trunc(x$평균월급)
> x
```

설명 : x 데이터 프레임의 평균월급의 값의 소수점 이하를 버리고 x 데이터 프레임의 평균월급에 넣는다.

문제66. 머신러닝 때 사용할 유방암 데이터를 R로 로드하고 전체 건수가 몇건인지 확인하시오!
(데이터 게시판 54번)

책 129페이지에 유방암 데이터에 대한 설명 참고

이 데이터는 위스콘신 대학교의 연구원이 기부했으며 유방 종양의 미세침습인물 디지털 이미지에서 측정한 값

이 들어있다. 이 값은 디지털 이미지에 존재하는 세포핵의 특성을 나타낸다. 569개의 암조직 검사 예시가 들어있으며 각 예시는 32개의 특징을 갖는다.

갑상선 초음파 검사할 때 나오는 디지털 이미지 데이터

1. 반지름
2. 질감
3. 둘레
4. 넓이
5. 매끄러움
6. 조밀성
7. 오목함

```
wisc <- read.csv("wisc_bc_data.csv")
```

```
> nrow(wisc) # 전체 건수 확인
```

```
[1] 569
```

```
> ncol(wisc) # 컬럼의 개수 확인
```

```
[1] 32
```

```
> str(wisc) # 데이터 프레임 구조 확인
```

```
$ diagnosis      : chr "B" "B" "B" "B" ...
$ radius_mean    : num 12.3 10.6 11 11.3 15.2 ...
$ texture_mean   : num 12.4 18.9 16.8 13.4 13.2 ...
$ perimeter_mean : num 78.8 69.3 70.9 73 97.7 ...
$ area_mean      : num 464 346 373 385 712 ...
$ smoothness_mean: num 0.1028 0.0969 0.1077 0.1164 0.0796 ...
$ compactness_mean: num 0.0698 0.1147 0.078 0.1136 0.0693 ...
$ concavity_mean : num 0.0399 0.0639 0.0305 0.0464 0.0339 ...
$ points_mean    : num 0.037 0.0264 0.0248 0.048 0.0266 ...
$ symmetry_mean  : num 0.196 0.192 0.171 0.177 0.172 ...
$ dimension_mean : num 0.0595 0.0649 0.0634 0.0607 0.0554 ...
$ radius_se       : num 0.236 0.451 0.197 0.338 0.178 ...
$ texture_se      : num 0.666 1.197 1.387 1.343 0.412 ...
$ perimeter_se    : num 1.67 3.43 1.34 1.85 1.34 ...
$ area_se         : num 17.4 27.1 13.5 26.3 17.7 ...
$ smoothness_se   : num 0.00805 0.00747 0.00516 0.01127 0.00501 ...
$ compactness_se  : num 0.0118 0.03581 0.00936 0.03498 0.01485 ...
$ concavity_se   : num 0.0168 0.0335 0.0106 0.0219 0.0155 ...
$ points_se       : num 0.01241 0.01365 0.00748 0.01965 0.00915 ...
$ symmetry_se     : num 0.0192 0.035 0.0172 0.0158 0.0165 ...
$ dimension_se    : num 0.00225 0.00332 0.0022 0.00344 0.00177 ...
$ radius_worst    : num 13.5 11.9 12.4 11.9 16.2 ...
$ texture_worst   : num 15.6 22.9 26.4 15.8 15.7 ...
$ perimeter_worst : num 87 78.3 79.9 76.5 104.5 ...
$ area_worst       : num 549 425 471 434 819 ...
$ smoothness_worst: num 0.139 0.121 0.137 0.137 0.113 ...
$ compactness_worst: num 0.127 0.252 0.148 0.182 0.174 ...
$ concavity_worst : num 0.1242 0.1916 0.1067 0.0867 0.1362 ...
$ points_worst    : num 0.0939 0.0793 0.0743 0.0861 0.0818 ...
```

```
$ symmetry_worst : num  0.283 0.294 0.3 0.21 0.249 ...
$ dimension_worst : num  0.0677 0.0759 0.0788 0.0678 0.0677 ..
```

설명 : diagnosis만 문자이고 나머지는 다 숫자인데 diagnosis가 유방암 환자가 양성(B)인지 악성인지(M)를 나타내는 라벨(정답) 컬럼 / 종속변수

문제67. 유방암 데이터에서 양성 환자가 몇명이고 악성환자가 몇명인지 확인하시오!

(세로)

```
> aggregate(id~diagnosis, wisc, length)
  diagnosis   id
  1           B 357
  2           M 212
```

(가로)

```
> table(wisc$diagnosis)
  B   M
357 212
```

문제68. 위의 결과를 원형 그래프로 시각화 하시오!

```
pie(table(wisc$diagnosis), col=c('green', 'pink'))
```

문제69. 직업, 직업별 토탈월급을 가로로 출력하시오!

세로 :

```
aggregate( sal~ job, emp, sum)
```

가로:

```
tapply(emp$sal, emp$job, sum)
```

문제70. 위의 결과를 막대 그래프로 시각화 하시오!

```
x <- tapply( emp$sal, emp$job, sum )
barplot( x, main="직업별 토탈월급", col=rainbow(5), density=50)
```

문제71. 입사한 년도(4자리), 입사한 년도별 토탈월급을 출력하시오!

세로:

```
> aggregate( sal~ format(as.Date(emp$hiredate), '%Y'), emp, sum)
  format(as.Date(emp$hiredate), "%Y")   sal
  1                      1980    800
  2                      1981 22825
  3                      1982  4300
```

가로:

```
> tapply(emp$sal, format(as.Date(emp$hiredate),'%Y'), sum)
1980 1981 1982 1983
800 22825 4300 1100
```

문제72. 위의 결과를 막대 그래프로 시각화 하시오!

```
> x <- tapply(emp$sal, format(as.Date(emp$hiredate),'%Y'), sum)
```

```
> barplot(x, main="년도별 토탈월급", col=rainbow(4), density=50)
```

문제73. 아래의 SQL의 결과를 R로 구현하시오!

```
SQL> select job, sum( decode( deptno, 10, sal) ) as "10",
      sum( decode( deptno, 20, sal) ) as "20",
      sum( decode( deptno, 30, sal) ) as "30"
    from emp
   group by job;
```

```
R> attach(emp)
R> tapply( sal, list(job, deptno), sum )
```

설명 : attach(emp)를 사용했기 때문에 emp\$sal, emp\$job, emp\$deptno 라고 안쓰고 sal, job, deptno 로만 작성할 수 있음을

결과:

	10	20	30
ANALYST	NA	6000	NA
CLERK	1300	1900	950
MANAGER	2450	2975	2850
PRESIDENT	5000	NA	NA
SALESMAN	NA	NA	5600

설명 : 위의 데이터를 가지고 그래프를 그리려면 NA가 있으면 안됩니다. NA값을 0으로 변경해줘야 합니다.

문제74. 위의 결과의 NA를 0으로 출력되게하시오!

```
R> attach(emp)
>x <- tapply( sal, list(job, deptno), sum )
설명 : list(job, deptno)를 써서 직업별 부서번호별 집계결과를 출력
```

```
> x[ is.na(x)] <- 0
> x
```

문제75. 위의 결과에서 컬럼명만 출력하고 로우명만 출력하시오!

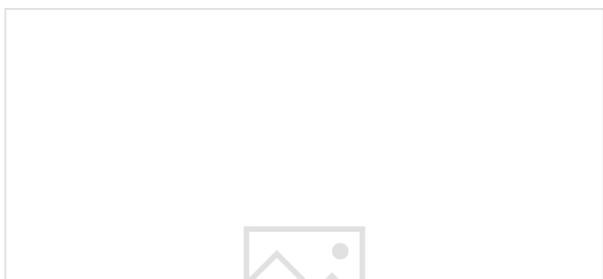
```
R> colnames(x)
R> rownames(x)
```

문제76. 위의 결과 데이터를 막대그래프로 시각화하시오!

```
barplot(x, col=rainbow(5), legend=rownames(x), beside = T, density = 50)
```

설명: legend는 그래프의 설명 박스

beside=T로 해야 각각의 막대그래프가 그려짐





beside=F로 하면 한 막대그래프 안에 원하는 데이터가 그려짐



```
barplot(x, col=rainbow(5), legend=("topright",rownames(x)), beside = T, density = 50)
```

문제77. 입사한 년도(4자리), 입사한 년도별 직업별 토탈월급을 출력하시오!

```
R > tapply( emp$sal, list(format( as.Date(emp$hiredate), '%Y'), emp$job), sum )
```

문제78. 위의 결과에서 NA를 0으로 출력되게 하시오!

```
> x <- tapply( emp$sal, list(emp$job,format( as.Date(emp$hiredate), '%Y') ), sum )
> x[is.na(x)] <-0
> x
```

	1980	1981	1982	1983
ANALYST	0	3000	3000	0
CLERK	800	950	1300	1100
MANAGER	0	8275	0	0
PRESIDENT	0	5000	0	0
SALESMAN	0	5600	0	0

문제79. 위의 결과를 막대 그래프로 그리시오~

```
> barplot(x, col=rainbow(5), legend=rownames(x), beside=T, density = 50)
```

문제80. (점심시간 문제) 입사한 년도(4자리)를 막대 그래프의 x축으로 구성하고 막대그래프의 y축을 부서번호의 평균월급으로 구성되게 하는데 입사한 년도별로 부서번호 10, 20, 30번이 각각 그려지게 하시오~

```
> x <- tapply( emp$sal, list(emp$deptno,format( as.Date(emp$hiredate), '%Y' )), mean )  
> x[is.na(x)] <- 0  
> barplot(x, col=rainbow(5), legend=rownames(x), beside=T, density = 50)
```



문제81. 직업과 직업별 토탈월급을 가지고 원형(pie)그래프를 그리시오!

답:

```
x <- tapply( emp$sal, emp$job, sum)  
pie( x, col=rainbow(5), density=80 )
```

문제82. 위의 그래프를 3D로 그리시오~

```
install.packages("plotrix")  
library( plotrix )  
pie3D(x, explode=0.1, labels=rownames(x) )
```

설명 : explode는 벌어진 정도를 나타낸다.



문제83. 위의 그래프의 결과에 직업 옆에 비율도 같이 출력되게 하시오!

```
x <- tapply( emp$sal, emp$job, sum ) #직업별 토탈월급을 가로로 출력  
x2 <- aggregate( sal~job, emp, sum ) #직업별 토탈월급을 세로로 출력  
pct <- round( x2$sal / sum(emp$sal) * 100, 1)  
pct
```

결과:

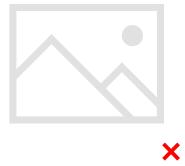
```
[1] 20.7 14.3 28.5 17.2 19.3
```

```
job_label <- paste( x2$job, ':', pct, '%' )  
job_label
```

결과:

```
"ANALYST : 20.7 %" "CLERK : 14.3 %" "MANAGER : 28.5 %"  
"PRESIDENT : 17.2 %" "SALESMAN : 19.3 %"
```

```
pie3D(x, explode=0.1, labels=job_label ) #3D 원형그래프
```



```
pie(x, labels=job_label, col = rainbow(5) ) # 그냥 원형그래프 그리기
```



문제84. 유방암 데이터의 diagnosis의 양성과 악성의 비율을 원형 그래프로 그리시오.

```
x <- tapply(wisc$id, wisc$diagnosis, length) # diagnosis별 건수를 가로로 출력  
x2 <- aggregate(id~diagnosis, wisc, length) # diagnosis별 건수를 세로로 출력
```

```
pct <- round(x2$id/ nrow(wisc) *100, 1)  
pct
```

```
w_label <- paste( x2$diagnosis, ':', pct, '%')  
w_label
```

```
pie(x, labels=w_label, col=c('green','red') )
```

설명: 그래프를 그릴때는 가로로 출력되는 결과 데이터를 가지고 그려야 합니다.
비율을 나타내는 라벨을 만들 때는 세로로 출력되는 데이터를 가지고 만듭니다.

문제85. 이름과 부서위치를 출력하시오!

```
SQL> select e.ename, d.loc  
      from emp e, dept d  
     where e.deptno = d.deptno;
```

#emp와 dept를 조인하는데 deptno에 의해서 조인해라

```
R> x <- merge( emp, dept, by="deptno")
```

```
R> x[행, 열]  
> x[, c("ename", "loc")]
```

설명 : x 데이터 프레임에서 ename과 loc만 출력

문제86. 부서위치가 DALLAS인 사원들의 이름과 월급과 부서위치를 출력하시오!

```
x[x$loc=='DALLAS', c("ename", "sal", "loc")]
```

문제87. 커미션이 NA인 사원들의 이름과 부서위치와 커미션을 출력하시오!

```
x[is.na(x$comm), c("ename", "loc", "comm")]
```

문제88. outer join으로 이름과 부서위치를 출력하는데 아래의 SQL과 동일한 결과가 출력되게 하시오!

```
SQL> select e.ename, d.loc  
      from emp e, dept d  
     where e.deptno(+) = d.deptno;
```

```
R> x <- merge( emp, dept, by="deptno", all.y=T )
```

↑ ↑
x y

```
> x[, c("ename", "loc")]
```

설명: all.y=T : dept 테이블 쪽에 데이터가 모두 나오게 해라~

문제89. 아래의 SQL의 결과를 R로 구현하시오!

```
SQL> select e.ename, d.loc  
      from emp e, dept d  
     where e.deptno = d.deptno(+);
```

```
R> x <- merge( emp, dept, by="deptno", all.x=T )  
> x[, c("ename", "loc")]
```

문제90. 아래의 SQL의 결과를 R로 구현하시오! full outer join

```
SQL> select e.ename, d.loc  
      from emp e full outer join dept d  
     on (e.deptno = d.deptno)
```

```
R> x <- merge( emp, dept, by="deptno", all=T)
> x[, c("ename", "loc")]
```

문제91. (self join) 이름을 출력하고 그 옆에 자기의 직속상사의 이름을 출력하시오!

```
SQL> select 사원.ename, 관리자.ename
   from emp 사원, emp 관리자
  where 사원.mgr = 관리자.empno;
```

```
R> x <- merge( emp, emp, by.x="mgr", by.y = "empno")
> x[,c("ename.x", "ename.y")]
```

결과:

	ename.x	ename.y
1	FORD	JONES
2	SCOTT	JONES
3	MARTIN	BLAKE
4	ALLEN	BLAKE
5	TURNER	BLAKE
6	JAMES	BLAKE
7	WARD	BLAKE
8	MILLER	CLARK
9	ADAMS	SCOTT
10	BLAKE	KING
11	CLARK	KING
12	JONES	KING
13	SMITH	FORD

문제92. 위의 결과를 다시 출력하는데 자기의 월급이 자기의 직속상사의 월급보다 더 큰 사원들만 출력하시오!

```
> x[x$sal.x > x$sal.y,c("ename.x", "ename.y")]
```

결과:

	ename.x	ename.y
1	FORD	JONES
2	SCOTT	JONES

문제93. 위의 결과 데이터인 사원이름과 직속상사의 이름을 출력하는 데이터를 가지고 사원 테이블의 조직도를 그리시오~

```
install.packages("igraph")
library(igraph)
> x <- merge(emp, emp, by.x="mgr", by.y="empno")
```

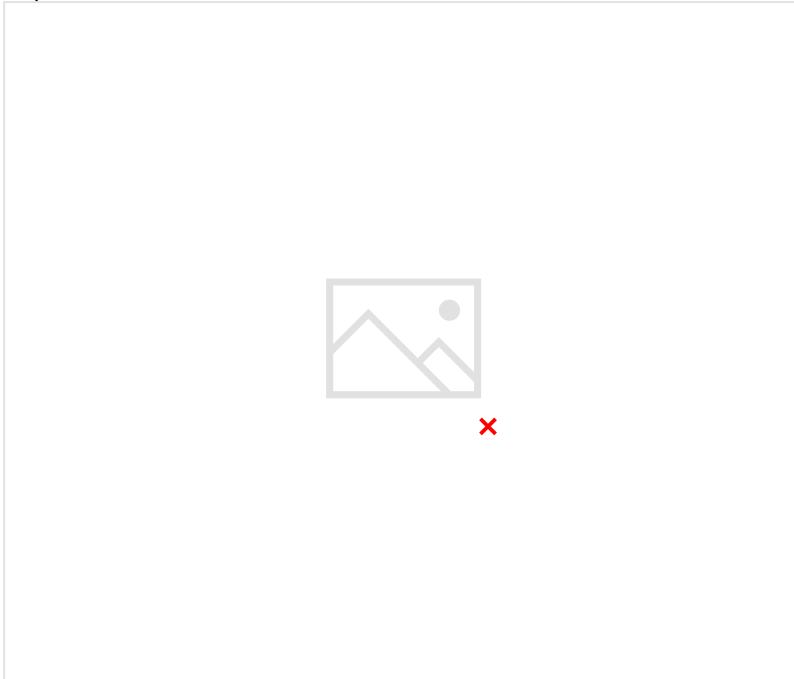
```
> a <- x[ , c("ename.x", "ename.y")]
> a
```

결과:

	ename.x	ename.y
1	FORD	JONES
2	SCOTT	JONES

```
3 MARTIN BLAKE
4 ALLEN BLAKE
5 TURNER BLAKE
6 JAMES BLAKE
7 WARD BLAKE
8 MILLER CLARK
9 ADAMS SCOTT
10 BLAKE KING
11 CLARK KING
12 JONES KING
13 SMITH FORD
```

```
> b <- graph.data.frame(a,directed=T)
> plot(b)
```



문제94. 위의 그래프를 구글의 googleVis를 이용해서 시각화 하시오!

답:

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

a <- merge(emp,emp, by.x="empno",by.y="mgr", all.y=T)

org <- gvisOrgChart(a, idvar="ename.y",parentvar="ename.x",
                      options=list(width=600, height=250, size='middle',allowCollapse=T))

plot(org)
```

문제95. 부서위치, 부서위치별 토탈월급을 출력하시오~

세로:

```
> x <- merge(emp, dept, by="deptno", all=T)
> aggregate(sal~loc,x, sum, na.action = na.pass)
```

가로:

```
> tapply(x$sal, x$loc, sum)
```

문제96. 위의 결과를 막대그래프로 시각화 하시오!

```
x <- merge(emp, dept, by="deptno", all=T)
x2 <- tapply(x$sal, x$loc, sum)
barplot( x2, col=rainbow(4), ylim=c(0,12000) )
```



✗

문제97. 아래와 같이 부서위치별 년도별 토탈월급을 출력하시오!

```
install.packages("lubridate")
library(lubridate)
year( emp$hiredate ) # lubridate를 이용해 hiredate에서 연도만 추출 가능.
```

답 : x <- merge(emp, dept, by="deptno", all=T)
tapply(x\$sal, list(x\$loc, year(x\$hiredate)), sum)

	1980	1981	1982	1983
BOSTON	NA	NA	NA	NA
CHICAGO	NA	9400	NA	NA
DALLAS	800	5975	3000	1100
NEW YORK	NA	7450	1300	NA

문제98. 위의 결과를 다시 출력하는데 NA대신에 0으로 출력되게 하시오!

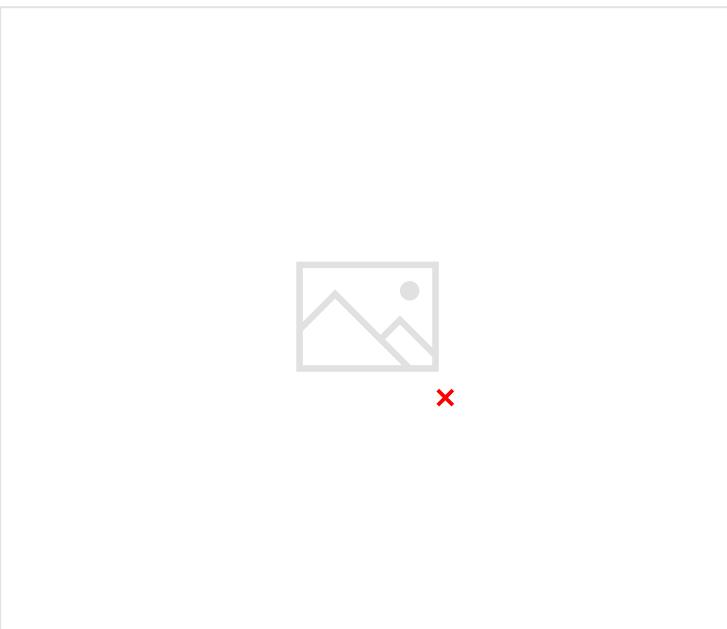
```
> y <- tapply( x$sal, list( year(x$hiredate) , x$loc), sum )
> y[is.na(y)] <-0
> y
```

결과:

	BOSTON	CHICAGO	DALLAS	NEW YORK
1980	0	0	800	0
1981	0	9400	5975	7450
1982	0	0	3000	1300
1983	0	0	1100	0

문제99. 위의 결과를 막대 그래프로 시각화 하시오!

```
답 : x <- merge( emp, dept, by="deptno", all=T)
> y <- tapply( x$sal, list( year(x$hiredate) , x$loc), sum )
> y[is.na(y)] <-0
> barplot( y, col=rainbow(4), beside=T, legend=rownames(y), ylim=c(0,10000) )
```



문제100. 지하철 1-4호선 승하차 승객수.csv를 R로 로드해서 line이라는 이름으로 데이터 프레임을 생성하시오!
(데이터 게시판 127번)

```
line <- read.csv( "1-4호선승하차승객수.csv", header=T )
head(line)
```

설명 : line_no : 몇호선인지 (1~4호선)

time : 시간

in : 승차 인원수

out : 하차 인원수

문제101. 위의 라인에서 line_no를 중복 제거해서 출력하시오!

```
line <- read.csv( "1-4호선승하차승객수.csv", header=T )
unique(line$line_no)
```

결과:

```
[1] "line_1" "line_2" "line_3" "line_4"
```

문제102. 위의 지하철 승하차수 정보를 가지고 구글 모션차트를 그리시오!

```
line <- read.csv( "1-4호선승하차승객수.csv", header=T )
t1 <- gvisMotionChart( line, idvar="line_no", timevar="time" )
plot(t1)
```

설명: 결제를 해야하므로 할 수 없습니다.

문제103. 부서번호, 부서번호별 토탈월급을 출력하는데 아래의 SQL처럼 전체 토탈월급이 출력되게하시오!

```
SQL> select deptno, sum(sal)
      from emp
      group by rollup(deptno);
```

```
R> rbind( aggregate( sal ~ deptno, emp, sum ), c(" ", sum(emp$sal) ) )
  deptno  sal
1     10  8750
2     20 10875
3     30  9400
4          29025
```

```
> aggregate( sal ~ deptno, emp, sum )
  deptno  sal
1     10  8750
2     20 10875
3     30  9400
```

```
> c(" ", sum(emp$sal) )
[1] " "    "29025"
```

설명 : rbind는 두개의 결과를 위아래로 출력하고 싶을 때 사용하는 함수
cbind는 두개의 결과를 양 옆으로 출력하고 싶을 때 사용하는 함수

문제104. cbind를 사용해서 아래의 두개의 결과를 하나로 출력하시오!

```
aggregate( sal~deptno, emp, sum )
aggregate( sal~deptno, emp, mean )
```

답:

```
cbind( aggregate( sal~deptno, emp, sum ), aggregate( sal~deptno, emp, mean ) )
```

좀더 깔끔하게 하는법:

```
x2 <- aggregate( sal~deptno, emp, mean )
cbind( aggregate( sal~deptno, emp, sum ), x2$sal )
  deptno  sal  x2$sal
1     10  8750 2916.667
2     20 10875 2175.000
3     30  9400 1566.667
```

문제105. 위의 결과의 컬럼명을 아래와 같이 출력되게하시오!

```
a <- aggregate( sal~deptno, emp, sum )
```

```

b <- aggregate( sal~deptno, emp, mean )
c <- cbind(a,b$sal)
names( c) <- c("부서번호", "임금", "평균임금")
c

```

결과:

	부서번호	임금	평균임금
1	10	8750	2916.667
2	20	10875	2175.000
3	30	9400	1566.667

문제106. (오늘의 마지막 문제 1/19) 아래의 SQL의 결과를 R로 구현하시오!

```

SQL> select job, sum(sal) as 토탈월급,
       max(sal) as 최대월급,
       min(sal) as 최소월급,
       avg(sal) as 평균월급,
       count(*) as 인원수
  from emp
 group by job;

```

```

a <- aggregate( sal~job, emp, sum)
b <- aggregate( sal~job, emp, max)
c <- aggregate( sal~job, emp, min)
d <- aggregate( sal~job, emp, mean)
e <- aggregate( sal~job, emp, length)
f <- cbind(a, b$sal, c$sal, d$sal, e$sal)
names( f ) <- c("직업", "임금", "최대월급", "최소월급", "평균월급", "인원수")

```

결과:

	직업	임금	최대월급	최소월급	평균월급	인원수
1	ANALYST	6000	3000	3000	3000.000	2
2	CLERK	4150	1300	800	1037.500	4
3	MANAGER	8275	2975	2450	2758.333	3
4	PRESIDENT	5000	5000	5000	5000.000	1
5	SALESMAN	5600	1600	1250	1400.000	4

문제107. 아래의 SQL의 결과를 R로 구현하시오!

```

SQL> select ename, sal, deptno
      from emp
      where deptno in (10, 20)
  union all
  select ename, sal, deptno
      from emp
      where deptno = 20;

```

```

R> emp[ emp$deptno %in% c(10,20), c("ename", "sal", "deptno")]
> emp[ emp$deptno ==20, c("ename", "sal", "deptno") ]

```

최종 답:

```
rbind(emp[ emp$deptno %in% c(10,20), c("ename", "sal", "deptno")],  
      emp[ emp$deptno ==20, c("ename", "sal", "deptno") ])
```

설명 : rbind는 두개의 결과를 위아래로 둑어서 출력하는 함수

문제108. 아래의 SQL을 R로 구현하시오!

```
SQL > select job, sum(sal)  
      from emp  
      group by job  
union all  
select null as job, sum(sal)  
      from emp;
```

답: 2개의 문장 합치기

1. aggregate(sal~job, emp, sum)
2. cbind(" ", sum(emp\$sal))

```
rbind( aggregate(sal~job, emp, sum), c(" ", sum(emp$sal)) )
```

문제109. 아래의 SQL의 결과를 R로 구현하시오!

```
SQL> select ename, sal, deptno  
      from emp  
      where deptnp in (10, 20)  
union  
select ename, sal, deptno  
      from emp  
      where deptno=10;
```

설명 : SQL의 union의 경우는 두 SQL의 결과를 위아래로 연결하면서 첫번째 컬럼인 ename을 기준으로 데이터 정렬을 하고 중복된 데이터를 제거합니다.

문제110. 아래의 SQL의 결과를 R로 구현하시오!

```
SQL> select ename, sal, deptno  
      from emp  
      where deptno in (10, 20)  
minus  
select ename, sal, deptno  
      from emp  
      where deptno = 10;
```

```
setdiff(emp[ emp$deptno %in% c(10,20), c("ename", "sal", "deptno")],  
        emp[ emp$deptno ==10, c("ename", "sal", "deptno") ])
```

설명 : R에 내장되어있는 setdiff 함수를 사용하면 안되고 **dplyr 패키지의 setdiff**를 이용해야 합니다.

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

```
setdiff(emp[ emp$deptno %in% c(10,20), c("ename", "sal", "deptno")],  
        emp[ emp$deptno ==10, c("ename", "sal", "deptno") ])
```

---- union 처럼 minus도 정렬을 하므로 정렬된 결과로 출력하시오!!

```
library(doBy)  
x <- setdiff(emp[ emp$deptno %in% c(10,20), c("ename", "sal", "deptno")],  
             emp[ emp$deptno ==10, c("ename", "sal", "deptno") ])  
orderBy(~ename, x)
```

문제111. 아래의 SQL의 결과를 R로 구현하시오!

```
SQL> select ename, sal, deptno  
      from emp  
     where deptno in ( 10, 20 )  
intersect  
select ename, sal, deptno  
      from emp  
     where deptno=10;
```

결과:

ename	sal	deptno
KING	5000	10
CLARK	2450	10
MILLER	1300	10

```
R>  
x <- intersect(emp[ emp$deptno %in% c(10,20), c("ename", "sal", "deptno")],  
                emp[ emp$deptno ==10, c("ename", "sal", "deptno") ])  
  
library(doBy)  
orderBy(~ename, x)
```

문제112. JONES의 월급보다 더 많은 월급을 받는 사원들의 이름과 월급을 출력하시오!

```
SQL> select ename, sal  
      from emp  
     where sal> (select sal  
                  from emp  
                 where ename='JONES');
```

```
R> jones_sal <- emp[ emp$ename=='JONES', c("sal") ]  
> emp[ emp$sal > jones_sal, c("ename", "sal")]
```

문제113. 아래의 SQL의 R로 구현하시오!

```
SQL> select ename, sal  
      from emp  
     where sal = ( select max(sal)  
                   from emp );
```

답:

```
R> max_sal <- emp(max$sal)
  > emp[ emp$sal == max_sal, c("ename", "sal")]
```

또는

```
max_sal<-max(emp[, "sal"])
emp[emp$sal==max_sal,c("ename","sal")]
```

문제114. 전국에서 등록금이 가장 비싼 학교이름과 등록금을 출력하시오!

```
univ <- read.csv("전국_대학별등록금통계_현황.csv")
```

```
> max_tuition<- max(univ[, "등록금"])
> univ[univ$등록금==max_tuition, c("학교명", "등록금")]
```

결과:

학교명	등록금
86 명지대학교 자연캠퍼스	9117

문제115. KING에게 보고하는 사원들의 이름과 월급을 출력하시오!

```
SQL> select ename, sal
      from emp
     where mgr = (select empno
                   from emp
                  where ename='KING');
```

```
R> king_empno <- emp[ emp$ename=='KING', "empno"]
  > emp[ emp$mgr == king_empno, c("ename", "sal") ]
```

결과 :

ename	sal
NA <NA>	NA
2 BLAKE	2850
3 CLARK	2450
4 JONES	2975

맨 위의 NA 부분을 제거하려면? na.omit을 사용

```
R> na.omit( emp[ emp$mgr == king_empno, c("ename", "sal") ] )
```

문제116. 관리자인 사원들의 이름을 출력하시오!

```
SQL> select ename
      from emp
     where empno in (select mgr
                      from emp );
```

설명: 관리자 번호(mgr)가 사원번호인 사원들을 출력

```
R> emp[ emp$empno %in% emp$mgr, "ename" ]  
결과: "KING" "BLAKE" "CLARK" "JONES" "FORD" "SCOTT"
```

세로로 나오게 하기 위한 방법:

```
R> library(data.table)  
> data.table( 이름=emp[ emp$empno %in% emp$mgr, "ename" ])
```

결과:

```
이름  
1: KING  
2: BLAKE  
3: CLARK  
4: JONES  
5: FORD  
6: SCOTT
```

문제117. 관리자가 아닌 사원들의 이름을 출력하시오!

```
SQL> select ename  
      from emp  
     where empno not in (select mgr  
                           from mgr  
                          where mgr is not null);
```

```
R> data.table( 이름=emp[! emp$empno %in% emp$mgr, "ename" ])
```

문제118. 아파트에서 가장 많이 발생한 범죄유형이 무엇인지 출력하시오!

(crime_loc.csv 를 사용)

```
> x <- crime_loc[crime_loc$장소=='아파트', c("범죄", "건수")]  
> x$x$건수 == max(x$건수),]
```

결과:

```
범죄 건수  
절도 25389
```

문제119. 지하철에서 가장 많이 발생하는 범죄유형이 무엇인지 출력하시오!

```
> x <- crime_loc[crime_loc$장소=='지하철', c("범죄", "건수")]  
> x$x$건수 == max(x$건수),]
```

결과:

```
범죄 건수  
강간 1339
```

문제120. 지하철에서 발생하는 범죄유형과 건수를 출력하는데 건수가 높은것부터 출력하시오!

```
> x <- crime_loc[crime_loc$장소=='지하철', c("범죄", "건수")]
library(dplyr)
orderBy(~-건수, x)
```

문제121. 위의 결과에서 첫번째 행만 출력하시오!

```
library(dplyr)
x2 <- orderBy(~-건수, x)
x2[1, ]
```

문제122. 문제 120번의 결과에서 첫번째 행부터 세번째 행까지 출력하시오!

```
> x2[c(1:3), ]
```

문제123. 강력범죄가 가장 많이 발생하는 요일은 언제인가?

(*crime_day.csv*를 이용)

```
crime_day <- read.csv("crime_day.csv")
head(crime_day)
crime_day[ crime_day$C_C=="강력범죄", ]
```

>>> 앞에 공백이 있어서 결과가 안나옴

설명 : trimws 함수 ---> 양쪽 공백 제거하는 함수 (데이터 전처리할 때 중요!!!!)

따라서..

```
x <- crime_day[ trimws(crime_day$C_C)=="강력범죄", ]
library(dplyr)
x2 <- orderBy(~-CNT, x )
x2[ 1, "DAY" ]
```

문제124. (점심시간 문제) 살인기수가 많이 발생하는 요일을 1위부터 3위까지 출력하시오!

```
> x <- crime_day[ trimws(crime_day$C_T)=="살인기수", ]
> x2 <- orderBy(~-CNT, x )
> x2[c(1:3), "DAY"]
```

결과:

"MON" "WED" "TUE"

문제125. 월요일에 많이 발생하는 범죄, 건수, 순위를 출력하시오.

```
library(dplyr)
library(dplyr)
crime_day <- read.csv("crime_day.csv")
c_mon <- crime_day[ crime_day$DAY=='MON', ]
head(crime_day)
```

```
x <- data.table( 범죄=c_mon$C_T, 건수=c_mon$CNT,
                 순위= dense_rank(-c_mon$CNT) )
orderBy(~순위, x )
```

문제126. 여자들이 많이 걸리는 암과 그 건수와 순위를 출력하시오!

(암종이 모든 암은 제외하는 조건을 주세요~)
(cancer2.csv를 사용하세요)

```
c <- cancer2[cancer2$성별=='여자'&cancer2$암종 !='모든암', ]
View( c ) #이렇게 하면 데이터를 예쁘게 볼 수 있음
```

```
> x <- data.table(암종=c$암종, 환자수=c$환자수, 순위=dense_rank(-c$환자수))
> x2 <- orderBy(~순위,x)
> x3 <- unique(x2)
> View(x3)
```

문제127. emp 테이블의 월급으로 기본적인 막대 그래프를 그리시오!

답: barplot(emp\$sal)

문제128. 위의 그래프의 제목을 Salary Bar Chart 라고 이름을 붙이시오!

답: barplot(emp\$sal, main="Salary Bar Chart")

문제129. 막대 그래프 x축에 사원이름을 붙이시오!

답: barplot(emp\$sal, main="Salary Bar Chart", names.arg=emp\$ename)

문제130. 막대 그래프의 x축과 y축의 이름을 각각 이름, 월급 이라고 하시오!

답: barplot(emp\$sal, main="Salary Bar Chart", names.arg=emp\$ename, xlab="이름", ylab="월급")

문제131. 막대 그래프의 색깔을 파란색으로 출력하시오!

답: barplot(emp\$sal, main="Salary Bar Chart", names.arg=emp\$ename, xlab="이름", ylab="월급", col="Green Yellow", density=80)



설명 : 카페에 R 색상표 확인

문제132. 창업건수.csv를 R로 로드하고 치킨집의 창업건수를 막대그래프로 시각화 하시오!
(데이터 게시판 50번)

```
create_cnt <- read.csv("창업건수.csv", header=T)
drop_cnt <- read.csv("폐업건수.csv", header=T)
View(create_cnt)
```

```
barplot( create_cnt$치킨집, main="년도별 치킨집 창업건수", names.arg=create_cnt$년도, col="Green Yellow",
density=80, ylim=c(0,1600) )
```



문제133. 폐업건수.csv를 R로 로드하고 치킨집의 창업건수를 막대그래프로 시각화 하시오!

```
barplot( drop_cnt$치킨집, main="년도별 치킨집 폐업건수", names.arg=drop_cnt$년도, col="Green Yellow", density=80, ylim=c(0,1600) )
```

문제134. 치킨집의 창업건수와 폐업건수를 같이 막대 그래프로 시각화하시오.

```
create_cnt <- read.csv("창업건수.csv", header=T)
drop_cnt <- read.csv("폐업건수.csv", header=T)
x <- rbind(create_cnt$치킨집, drop_cnt$치킨집)
x
barplot( x , main="년도별 치킨집 창업과 폐업건수",
         names.arg=drop_cnt$년도, col= c("Green Yellow","Hot pink") ,
         density=80, ylim=c(0,4000) , beside=T )
```



문제135. 카페(커피음료)가 얼마나 창업을 하고 얼마나 폐업을 하는지 막대 그래프로 시각화하시오!

```
create_cnt <- read.csv("창업건수.csv", header=T)
drop_cnt <- read.csv("폐업건수.csv", header=T)
x <- rbind(create_cnt$커피음료, drop_cnt$커피음료)
x
barplot( x , main="년도별 카페 창업과 폐업건수",
         names.arg=drop_cnt$년도, col= c("Green Yellow","Hot pink") ,
         density=80, ylim=c(0,4000) , beside=T )
```



문제136. 위의 막대 그래프를 앞으로 편하게 그릴 수 있도록 함수로 만들어서 실행되게 하시오!

R 함수 생성 문법:

함수명 <- function(){ 함수코드 }

실행: 함수명 적기

해보기:

```
ms_auto <-function() {  
  create_cnt <- read.csv("창업건수.csv", header=T)  
  drop_cnt <- read.csv("폐업건수.csv", header=T)  
  x <- rbind(create_cnt$커피음료, drop_cnt$커피음료)  
  barplot( x , main="년도별 카페 창업과 폐업건수",  
           names.arg=drop_cnt$년도, col= c("Green Yellow","Hot pink") ,  
           density=80, ylim=c(0,4000) , beside=T )  
}  
  
ms_auto()
```

문제137. 창업건수.csv를 로드해서 치킨집의 창업건수를 막대 그래프로 그리는 함수를 생성하시오~

```
a_bar <- function(){  
  create_cnt <- read.csv("창업건수.csv", header=T)  
  
  barplot( create_cnt$치킨집, main="년도별 치킨집 창업건수", names.arg=create_cnt$년도, col="Green Yellow",  
           density=80, ylim=c(0,1500) )  
}
```

문제138. 사원 테이블의 월급을 원형 그래프로 그리시오!

```
pie(emp$sal)
```



×

문제139. 위의 그래프를 다시 그리는데 누구의 월급인지가 명시되게 하시오!

```
pie(emp$sal, main="Salary Pie Chart", labels=emp$ename, col=rainbow(15) )
```

```
sal_labels <- round( emp$sal/sum(emp$sal)*100, 1)
```

```
sal_labels2 <- paste( emp$ename, sal_labels, '%' )
```

```
sal_labels2
```

```
pie(emp$sal, main="Salary Pie Chart", labels=sal_labels2, col=rainbow(15) )
```



×

문제141. 아래의 데이터로 plot(점) 그래프를 그리시오!

```
cars <- c(1, 3, 6, 4, 9)
```

```
cars
```

```
plot(cars)
```

```
plot(cars, type='o', col='blue')
```



설명 : type='o' : 선을 그어라~

문제143. 차와 트럭의 판매된 차량 수를 라인 그래프로 시각화 하시오!

```
cars <- c(1,3,6,4,9)  
trucks <- c(2, 5, 4, 5, 12)
```

```
plot(cars, type='o', col='blue', ylim=c(0,12) )
```

#그래프 창을 닫지 말고 바로 이어서 아래의 코드를 실행

```
lines(trucks, type='o', pch=22, lty=2, col='red', ylim=c(0,12) )
```



설명: pch=21 : 동그라미 , lty=1 : 직선

pch=22 : 네모, lty=2: 점선

문제144. 가로축을 월, 화, 수, 목, 금으로 변경하세요

```
cars <- c(1,3,6,4,9)
```

```

trucks <- c(2, 5, 4, 5, 12)

plot(cars, type='o', col='blue', ylim=c(0,12), axes=FALSE, ann=FALSE )

#그래프 창을 닫지 말고 바로 이어서 아래의 코드를 실행
lines(trucks, type='o', pch=22, lty=2, col='red', ylim=c(0,12) )

```

설명: axes=FALSE: x 축과 y축을 지워라~

ann=FALSE: 축 이름을 지워라

- 새로운 축을 생성하는 방법
`axis(1, at=1:5, lab=c("mon", "tue", "wed", "thu", "fri"))`

↑

x축

`axis(2) #y축생성`



- 레전드 붙이기
`legend(2, 10, c('car', 'truck'), col=c('blue', 'red'), cex=0.8, pch=32:22, lty=1:2)`

설명: cex는 글씨 크기, 레전드 안에 pch 21(동그라미), pch22(네모), lty1 (직선), lty2(점선)을 표시한다.

문제145. 치킨집의 창업건수를 이용해서 라인 그래프를 그리시오! x축을 연도로 두고 y축을 창업건수로 두고 만드시오!

```
create_cnt <- read.csv("창업건수.csv", header=T)
```

```
plot(create_cnt$치킨집, type='o', col='blue', axes=FALSE, ann=FALSE )
```

```
axis( 1, at=1:10, lab=create_cnt$년도 )
```

```
axis(2) #y축생성
```

```
legend(2, max(create_cnt$치킨집), '창업', col='blue' , cex=0.8, pch=22, lty=1)
```

ylim 안쓰면 자동으로 위치 찾아줌!!



문제146. 치킨집의 창업건수와 폐업건수를 같이 출력되게 하시오!
(라인 그래프로 그리시오!)

```
create_cnt <- read.csv("창업건수.csv", header=T)
drop_cnt <- read.csv("폐업건수.csv", header=T)
```

```
plot(drop_cnt$치킨집, type='o', col='red', axes=FALSE, ann=FALSE, pch=22, lty=2)
lines(create_cnt$치킨집, type='o', pch=21, lty=1, col='blue' )
```

```
axis( 1, at=1:10, lab=create_cnt$년도 )
axis(2) #y축생성
legend(8, max(create_cnt$치킨집), c('폐업', '창업') , col=c('red','blue') , cex=0.8, pch=22:21, lty=2:1)
```



문제147. (오늘의 마지막 문제 1/20) 막대그래프, 원형 그래프, 라인그래프 3가지를 그렸는데 이 세가지 그래프를 쉽게 실행할 수 있도록 함수로 만들자

auto_bar() : 창업/ 폐업 막대그래프 실행
auto_pie() : 사원테이블의 월급 비율 원형 그래프 실행
auto_line() : 창업/ 폐업 라인 그래프 실행

```
auto_bar <- function() {  
  create_cnt <- read.csv("창업건수.csv", header=T)  
  drop_cnt <- read.csv("폐업건수.csv", header=T)  
  x <- rbind(create_cnt$치킨집, drop_cnt$치킨집)  
  barplot( x , main="년도별 치킨집 창업과 폐업건수",  
           names.arg=drop_cnt$년도, col= c("Green Yellow","Hot pink") ,  
           density=80, ylim=c(0,4000) , beside=T )  
}  
  
auto_pie <- function() {  
  sal_labels <- round( emp$sal/sum(emp$sal)*100, 1)  
  sal_labels2 <- paste( emp$ename, sal_labels, '%' )  
  sal_labels2  
  pie(emp$sal, main="Salary Pie Chart", labels=sal_labels2, col=rainbow(15) )  
}  
  
auto_line <- function() {  
  create_cnt <- read.csv("창업건수.csv", header=T)  
  drop_cnt <- read.csv("폐업건수.csv", header=T)  
  
  plot(drop_cnt$치킨집, type='o', col='red', axes=FALSE, ann=FALSE, pch=22, lty=2)  
  lines(create_cnt$치킨집, type='o', pch=21, lty=1, col='blue' )  
  
  axis( 1, at=1:10, lab=create_cnt$년도 )
```

```

axis(2) #y축생성
legend(8, max(create_cnt$치킨집), c('폐업', '창업') , col=c('red','blue') , cex=0.8, pch=22:21, lty=2:1)

}

```

문제148. 막대 그래프 자동화 코드를 이용해서 원형 그래프 자동화 함수를 생성하시오!

```

pie2 <- function() {      # R 에서 함수 만드는 코드

  fname <- file.choose() # 원도우 탐색기 여는 코드

  table <- read.csv(fname, header=T, stringsAsFactor=F )

  print(data.table(colnames(table))) # 컬럼명을 번호와 함께 출력하는 코드

  xcol_num <- as.numeric(readline('x축 컬럼 번호: ')) # 번호 물어보기
  ycol_num <- as.numeric(readline('y축 컬럼 번호: ')) # 번호 물어보기

  xcol <- colnames(table[xcol_num])  # x축 컬럼명을 담는 코드
  ycol <- colnames(table[ycol_num])  # y축 컬럼명을 담는 코드

  xcol2 <- table[,xcol] # x 축 컬럼의 데이터를 xcol2에 넣는다.
  ycol2 <- table[,ycol] # y 축 컬럼의 데이터를 ycol2에 넣는다.

  y_labels <- round( ycol2/ sum(ycol2) *100, 1)
  y_labels2 <- paste(xcol2, y_labels, '%')
  pie( ycol2 , main=paste(ycol , '의 원형 그래프') , labels=y_labels2, col=rainbow(15) )

}

```

문제149. 아래의 메뉴 코드를 실행하시오!

```

x1 <- menu(c('막대 그래프', '원형 그래프'),
           title="숫자를 선택하세요~" )

```

문제150. 아래의 switch코드를 실행하시오!

```

my_func <- function() {

  bar <- function() {      # R 에서 함수 만드는 코드

    fname <- file.choose() # 원도우 탐색기 여는 코드

    table <- read.csv(fname, header=T, stringsAsFactor=F )

    print(data.table(colnames(table))) # 컬럼명을 번호와 함께 출력하는 코드
  }
}

```

```

xcol_num <- as.numeric(readline('x축 컬럼 번호: ')) # 번호 물어보기
ycol_num <- as.numeric(readline('y축 컬럼 번호: ')) # 번호 물어보기

xcol <- colnames(table[xcol_num]) # x축 컬럼명을 담는 코드
ycol <- colnames(table[ycol_num]) # y축 컬럼명을 담는 코드

xcol2 <- table[,xcol] # 예: emp[, "empno"]
ycol2 <- table[,ycol] # y 축 컬럼의 데이터를 ycol2에 넣는다.
y_max <- max(ycol2) + 100

barplot(ycol2, main=paste(xcol,'과',ycol,'의 막대 그래프'), names.arg=xcol2,
        col=c('Green Yellow'), density=80, ylim= c(0,y_max) , beside=T)
}

```

```

pie2 <- function() {      # R에서 함수 만드는 코드

  fname <- file.choose() # 윈도우 탐색기 여는 코드

  table <- read.csv(fname, header=T, stringsAsFactor=F )

  print(data.table(colnames(table))) # 컬럼명을 번호와 함께 출력하는 코드

  xcol_num <- as.numeric(readline('x축 컬럼 번호: ')) # 번호 물어보기
  ycol_num <- as.numeric(readline('y축 컬럼 번호: ')) # 번호 물어보기

  xcol <- colnames(table[xcol_num]) # x축 컬럼명을 담는 코드
  ycol <- colnames(table[ycol_num]) # y축 컬럼명을 담는 코드

  xcol2 <- table[,xcol] # x 축 컬럼의 데이터를 xcol2에 넣는다.
  ycol2 <- table[,ycol] # y 축 컬럼의 데이터를 ycol2에 넣는다.

  y_labels <- round( ycol2/ sum(ycol2) *100, 1)
  y_labels2 <- paste(xcol2, y_labels, '%')

  pie( ycol2 , main=paste(ycol , '의 원형 그래프') , labels=y_labels2, col=rainbow(15) )

}

x1 <- menu( c('막대 그래프', '원형 그래프'),
            title=" 숫자를 선택하세요 ~ " )
switch( x1,
        gp1 = { bar() },
        gp2 = { pie2() }
      )
my_func()

```

문제151. cars2.csv를 생성하고 cars2.csv의 시계열 데이터로 라인 그래프를 그리시오!

```

cars <- read.csv("cars.csv")
y_max <- max(cars$cnt2) + 5
plot(cars$cnt2, type='o', col='blue', ylim=c(0,y_max), axes=FALSE, ann=FALSE )
axis( 1, at=1:5, lab=cars$date2 )
axis(2) #y축생성
legend(2, 10, 'car', col='blue', cex=0.8, pch=21, lty=1)

```



문제152. 위의 스크립트를 이용해서 윈도우에서 csv 파일을 불러와서 라인 그래프를 그리는 함수를 line2라는 이름으로 생성하시오!

```

line2 <- function() {      # R 에서 함수 만드는 코드

  fname <- file.choose() # 윈도우 탐색기 여는 코드

  table <- read.csv(fname, header=T, stringsAsFactor=F )

  print(data.table(colnames(table))) # 컬럼명을 번호와 함께 출력하는 코드

  xcol_num <- as.numeric(readline('x축 컬럼 번호: ')) # 번호 물어보기
  ycol_num <- as.numeric(readline('y축 컬럼 번호: ')) # 번호 물어보기

  xcol <- colnames(table[xcol_num])  # x축 컬럼명을 담는 코드
  ycol <- colnames(table[ycol_num])  # y축 컬럼명을 담는 코드

  xcol2 <- table[,xcol] # x 축 컬럼의 데이터를 xcol2에 넣는다.
  ycol2 <- table[,ycol] # y 축 컬럼의 데이터를 ycol2에 넣는다.

  y_max <- max(ycol2) + 5
  plot(ycol2, type='o', col='blue', ylim=c(0,y_max), axes=FALSE, ann=FALSE )
  axis( 1, at=1:5, lab=xcol2 )
  axis(2) #y축생성
  legend(2, 10, table, col='blue', cex=0.8, pch=21, lty=1)
}

```

문제153. my_func 함수(그래프 통합 코드 함수)에 line2 함수로 만든 라인 그래프를 세번째로 추가시키시오!

```
my_func <- function() {  
  
  bar <- function() {    # R 에서 함수 만드는 코드  
  
    fname <- file.choose() # 윈도우 탐색기 여는 코드  
  
    table <- read.csv(fname, header=T, stringsAsFactor=F )  
  
    print(data.table(colnames(table))) # 컬럼명을 번호와 함께 출력하는 코드  
  
    xcol_num <- as.numeric(readline('x축 컬럼 번호: ')) # 번호 물어보기  
    ycol_num <- as.numeric(readline('y축 컬럼 번호: ')) # 번호 물어보기  
  
    xcol <- colnames(table[xcol_num])  # x축 컬럼명을 담는 코드  
    ycol <- colnames(table[ycol_num])  # y축 컬럼명을 담는 코드  
  
    xcol2 <- table[,xcol] # 예: emp[, "empno"]  
    ycol2 <- table[,ycol] # y 축 컬럼의 데이터를 ycol2에 넣는다.  
    y_max <- max(ycol2) + 100  
  
    barplot(ycol2, main=paste(xcol,'과',ycol,'의 막대 그래프'), names.arg=xcol2,  
            col=c('Green Yellow'), density=80, ylim= c(0,y_max) , beside=T)  
  }  
  
  pie2 <- function() {    # R 에서 함수 만드는 코드  
  
    fname <- file.choose() # 윈도우 탐색기 여는 코드  
  
    table <- read.csv(fname, header=T, stringsAsFactor=F )  
  
    print(data.table(colnames(table))) # 컬럼명을 번호와 함께 출력하는 코드  
  
    xcol_num <- as.numeric(readline('x축 컬럼 번호: ')) # 번호 물어보기  
    ycol_num <- as.numeric(readline('y축 컬럼 번호: ')) # 번호 물어보기  
  
    xcol <- colnames(table[xcol_num])  # x축 컬럼명을 담는 코드  
    ycol <- colnames(table[ycol_num])  # y축 컬럼명을 담는 코드  
  
    xcol2 <- table[,xcol] # x 축 컬럼의 데이터를 xcol2에 넣는다.  
    ycol2 <- table[,ycol] # y 축 컬럼의 데이터를 ycol2에 넣는다.  
  
    y_labels <- round( ycol2/ sum(ycol2) *100, 1)  
    y_labels2 <- paste(xcol2, y_labels, '%')  
  
    pie( ycol2 , main=paste(ycol , '의 원형 그래프') , labels=y_labels2, col=rainbow(15) )  
  }  
}
```

```

}

line2 <- function() {      # R 에서 함수 만드는 코드

  fname <- file.choose() # 윈도우 탐색기 여는 코드

  table <- read.csv(fname, header=T, stringsAsFactor=F )

  print(data.table(colnames(table))) # 컬럼명을 번호와 함께 출력하는 코드

  xcol_num <- as.numeric(readline('x축 컬럼 번호: ')) # 번호 물어보기
  ycol_num <- as.numeric(readline('y축 컬럼 번호: ')) # 번호 물어보기

  xcol <- colnames(table[xcol_num]) # x축 컬럼명을 담는 코드
  ycol <- colnames(table[ycol_num]) # y축 컬럼명을 담는 코드

  xcol2 <- table[,xcol] # x 축 컬럼의 데이터를 xcol2에 넣는다.
  ycol2 <- table[,ycol] # y 축 컬럼의 데이터를 ycol2에 넣는다.

  y_max <- max(ycol2) + 5
  plot(ycol2, type='o', col='blue', ylim=c(0,y_max), axes=FALSE, ann=FALSE )
  axis( 1, at=1:5, lab=xcol2 )
  axis(2) #y축생성
  legend(2, 10, table, col='blue', cex=0.8, pch=21, lty=1)
}

```

```

x1 <- menu( c('막대 그래프', '원형 그래프', '라인 그래프'),
            title=" 숫자를 선택하세요 ~ " )
switch( x1,
        gp1 = { bar() },
        gp2 = { pie2() },
        gp3 = { line2() }
      )

```

문제154. 중고차의 마일리지로 히스토그램 그래프를 그리시오!
(그레프와 코드를 카페에 올리고 검사받으세요~)

```
> hist(usedcars$mileage, breaks= seq(0, 200000, by=20000) )
```



권준환씨 코드:

```
max_m = max(usedcars$mileage)*1.1  
주행거리=usedcars$mileage  
hist( 주행거리,  
, breaks=seq(0, max_m , by=10000) , at=seq(0, max_m ,by=10000), main="중고차 주행거리", col='orange',  
density=80 )
```

문제155. 위의 그래프 코드를 함수로 생성해서 윈도우 탐색기에서 데이터를 불러와 히스토그램 그래프를 그려
지게 하시오!

```
hist2 <- function() {      # R 에서 함수 만드는 코드  
  
  fname <- file.choose() # 원도우 탐색기 여는 코드  
  
  table <- read.csv(fname, header=T, stringsAsFactor=F )  
  
  print(data.table(colnames(table))) # 컬럼명을 번호와 함께 출력하는 코드  
  
  xcol_num <- as.numeric(readline('x축 컬럼 번호: ')) # 번호 물어보기  
  
  xcol <- colnames(table[xcol_num])  # x축 컬럼명을 담는 코드  
  
  xcol2 <- table[,xcol] # x 축 컬럼의 데이터를 xcol2 에 넣는다.  
  
  max_m = max(xcol2)*1.1  
  주행거리=xcol2  
  hist(xcol2, ,at=seq(0, max_m ,by=10000), main=paste(xcol,'의 히스토그램 그래프') , col='orange', density=80 )  
}
```

문제156. 다음 농구선수 3명의 득점 점수의 평균값, 중앙값, 최빈값을 각각 구하시오!

■■■ 농구 선수 3명이 각각의 게임당 득점한 점수

```
x1 <- c(7,8,9,9,10,10,11,11,12,13)
x2 <- c(7,9,9,10,10,10,10,11,11,13 )
x3 <- c(1,1,7,7,10,10,10,11,13,30 )
```

평균:

```
mean(x1)
mean(x2)
mean(x3)
```

중앙값:

```
median(x1)
median(x2)
median(x3)
```

결과가 모두 10이다.

최빈값:

```
> names(table(x1))[table(x1)==max(table(x1))] #9,10,11
> names(table(x2))[table(x2)==max(table(x2))] #10
> names(table(x3))[table(x3)==max(table(x3))] #10
```

설명 : 보통은 위의 3개의 통계치로 한명의 선수를 선택할 수 있는데 위와 같은 경우는 평균값, 중앙값, 최빈값으로는 특정선수를 선택하기가 곤란하다.

* 이런 경우에 필요한 데이터 분석방법은?

-> 각 선수의 점수 데이터의 분포가 어떻게 분포 되었는지 분포 방식을 측정할 수 있으면 결정을 내릴 때 크게 도움을 줄 수 있다.

문제157. 이상치 데이터(30점)를 가지고 있는 3번째 선수의 점수 데이터로 4분위수 그래프를 그리시오!

```
a <- boxplot(x3)
a
a$stats
```



- [,1]
[1,] 1 <- 하한값
[2,] 7 <- 하한 사분위수
[3,] 10 <- 중앙값
[4,] 11 <- 상한 사분위수
[5,] 13 <- 상한값

이상치: 가운데의 50% 데이터에만 집중을 함

문제158. 위와 같이 사분위수 그래프를 그리면서 위의 통계치를 확인하는 이유가 무엇인가?

답 : 이상치 때문, 이상치를 제거하고 가운데 50%의 데이터에만 집중함으로써 문제를 우회할 수 있다.

머신러닝 모델을 학습 시킬때도 이상치를 제거하고 머신러닝 모델을 학습시킨 결과가 훨씬 더 학습률이 좋아서 모델의 판별 정확도가 더 높은 경우가 많습니다.

문제159. 3명의 농구 선수들의 점수를 가지고 사분위수 그래프를 그리는데 1번, 2번, 3번 선수의 그래프를 하나의 결과로 나오게 출력하시오!

boxplot(x1, x2,x3)



설명 : 그래프를 보면 1, 2번 선수가 3번 선수보다 상대적으로 좁은 범위를 가지고 있다. 3번 선수는 넓은 범위를 가지고 있고 3번 선수는 2번 선수에 비해 훨씬 높은 점수(30점)을 득점했지만 다른 경우에는 훨씬 낮은 점수에 대한 기록도 보인다. 그래서 1, 2번 선수가 더 일관성이 있고 대부분의 경우에 3번 선수보다 더 높은 점수를 기록했다.

그래서 만약 세 명의 선수 중에 한 명의 선수를 고른다면 1번과 2번 선수에서 골라야 한다.

문제160. 위의 3명의 선수들 중에 3번째 선수는 제외했고 1번과 2번 선수 중 한 명을 선택해야 한다면 어떻게 해야 할까?

분산: 데이터의 퍼짐 정도를 수치화 한 것

확률 변수가 기댓값으로 얼마나 떨어진 곳에 분포하는지 가늠하는 숫자

표준편차: 분산의 제곱근 값으로 이 값을 통하여 평균에 흘어진 정도를 확인할 수 있다.

- 빅데이터 기사시험 수제비 2권 4-16 참조

```
var(x1) # 3.333333  
var(x2) # 2.444444
```

```
sd(x1) # 1.825742  
sd(x2) # 1.563472
```

설명 : 두 번째 선수가 첫 번째 선수보다 점수의 폭이 크지 않고 일관된 형태를 보이고 있다.

표준편차는 데이터가 정규분포를 따른다는 가정하에 주어진 값이 얼마나 평균으로부터 멀리 떨어졌는지를 빠르게 평가할 때 사용한다.

정규분포 68% ---> 평균으로부터 1표준편차

정규분포 95% ---> 평균으로부터 2표준편차

정규분포 99.7% ----> 평균으로부터 3표준편차

문제161. 중고차 가격(price)에 대한 사분위수 그래프를 그리고 사분위수 통계치를 확인하시오!

```
usedcar <- read.csv("usedcars.csv")  
a = boxplot(usedcar$price)
```

```
a$stats
```

결과:

[,1]

[1,] 5980.0 ---> 하한값

[2,] 10995.0 ----> 하한 사분위수

[3,] 13591.5 ----> 중앙값 (가운데 굽은 줄)

[4,] 14906.0 ---> 상한 사분위수

[5,] 19995.0 ----> 상한값

attr("class")

1



```
a$out
```

[1] 21992 20995 4899 3800 ---> 이상치 확인

문제162. 중고차 가격의 가운데 50%에 해당하는 데이터에 집중하기 위해서 Q1과 Q3 값을 확인하시오!

```
quantile( usedcar$price )
```

```
> quantile( usedcars$price )
```

0% 25% 50% 75% 100%

3800.0 10995.0 13591.5 14904.5 21992.0

↑ ↑ ↑

Q1 Q2 Q3

하한사분위수 중앙값 상한 사분위수

문제163. 미국지도를 시각화 하시오!

```
map("world", "USA")
```

문제164. (오늘의 마지막 문제 1/21) jobs.txt을 워드 클라우드로 시각화하시오!

```
library(rJava)
```

```
library(KoNLP)
```

```

setwd("d:\data") # 워킹디렉토리를 소환
jobs <- readLines('jobs.txt')

nouns <- extractNoun(jobs) # 명사 단어만 추출
nouns <- unlist(nouns)
nouns <- nouns[nchar(nouns)>=2] # 단어중에 2철자 이상인것만
cnouns <- count(nouns) # 단어별 건수를 출력한다.

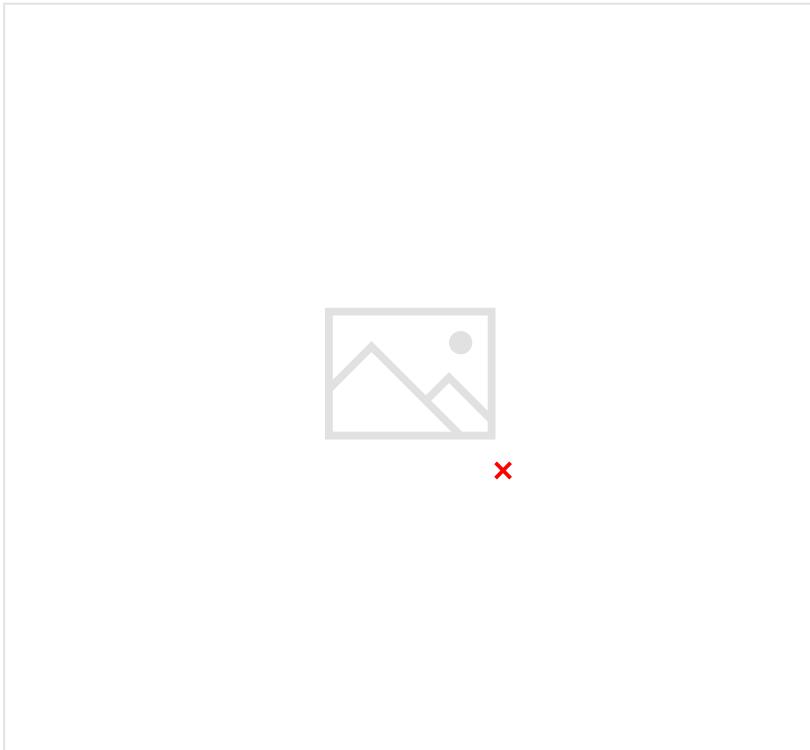
# 색깔 추가
pal <- brewer.pal(6,"Dark2")
pal <- pal[-(1)]

# 글씨 폰트 설정
windowsFonts(malgun=windowsFont("맑은 고딕"))

wordcloud( words=cnouns$x, # 단어
           freq=cnouns$freq, # 단어 빈도수
           colors=pal, # 색깔
           min.freq=3, # 빈도수가 3개 이상인것만 시각화
           random.order=F, # F로 하게 되면 큰글씨부터 출력이 되면서
           family="malgun") # 중앙에서 퍼지게 한다.

# 맑음 글씨체로 시각화 하겠다.

```



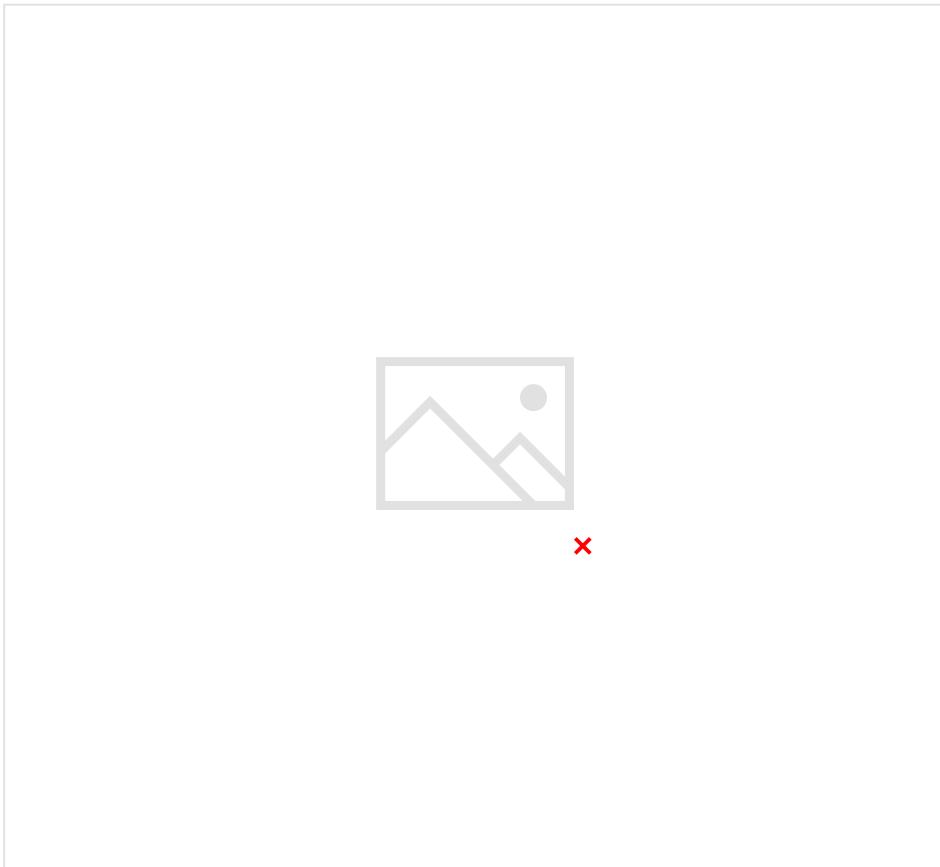
문제165. (점심시간 문제) 아래의 문자 3개를 factor로 만드는데 순서를 SEVERE> MODERATE> MILD 이 순서가 되게 만드시오. factor 이름은 b라고 하시오!

SEVERE MILD MODERATE

```
a <- c("SEVERE", "MILD", "MODERATE")  
a1<- factor( a, order=TRUE, level=c("SEVERE", "MODERATE", "MILD") )  
a1
```

문제166. 중고차의 주행거리가 높으면 중고차의 가격이 낮아지는지 산포도 그래프로 확인하시오!

```
plot(car$mileage, car$price, pch=21, col='blue', bg='blue')
```



설명 : 주행거리가 높을수록 가격이 낮아지는 음의 상관관계를 보이고 있습니다.

```
cor( car$mileage, car$price ) # 결과: -0.8061494
```

문제167. 산포도 그래프를 자동화 스크립트에 7번에 추가하시오 !

오전에 나눠준 my_func.R 스크립트 중에 산포도 그래프 함수를 가져다가 my_func2.R 에 추가하시오 !

```
sanpodo <- function() {  
  
  fname <- file.choose()  
  table <- read.csv(fname, header=T, stringsAsFactor=F )  
  
  print(data.table(colnames(table)))  
  
  xcol_num <- as.numeric(readline('x축 컬럼 번호: '))  
  ycol_num <- as.numeric(readline('y축 컬럼 번호: '))
```

```

xcol <- colnames(table[xcol_num])
ycol <- colnames(table[ycol_num])

xcol2 <- table[,xcol]
ycol2 <- table[,ycol]

plot(xcol2,ycol2,
main=paste(xcol,'과',ycol,'의 산포도 그래프'),lwd=2, xlab=xcol,ylab=ycol,col='blue',pch=21,bg='blue')
}

```

문제168. 관계를 관찰 할 수 있는 또 다른 척도인 이원 교차표를 이용해서 직업과 월급과의 관계를 관찰하여 직업별로 월급의 차이가 존재하는지 확인하시오!

월급을 2500을 기준으로 직업별로 각각 월급이 2500 이상인 사원들과 월급이 2500보다 작은 사원들의 분포를 확인합니다.

답: library(data.table)
 data.table(emp\$sal, emp\$sal>=2500)

결과:

	V1	V2
1:	5000	TRUE
2:	2850	TRUE
3:	2450	FALSE
4:	2975	TRUE
5:	1250	FALSE
6:	1600	FALSE
7:	1500	FALSE
8:	950	FALSE
9:	1250	FALSE
10:	3000	TRUE
11:	800	FALSE
12:	3000	TRUE
13:	1100	FALSE
14:	1300	FALSE

emp\$sal_tf <- emp\$sal >=2500 # 이 컬럼이 추가됨
 emp

	index	empno	ename	job	mgr	hiredate	sal	comm	deptno	sal_tf
1	1	7839	KING	PRESIDENT	NA	1981-11-17 0:00	5000	NA	10	TRUE
2	2	7698	BLAKE	MANAGER	7839	1981-05-01 0:00	2850	NA	30	TRUE
3	3	7782	CLARK	MANAGER	7839	1981-05-09 0:00	2450	NA	10	FALSE
4	4	7566	JONES	MANAGER	7839	1981-04-01 0:00	2975	NA	20	TRUE
5	5	7654	MARTIN	SALESMAN	7698	1981-09-10 0:00	1250	1400	30	FALSE
6	6	7499	ALLEN	SALESMAN	7698	1981-02-11 0:00	1600	300	30	FALSE
7	7	7844	TURNER	SALESMAN	7698	1981-08-21 0:00	1500	0	30	FALSE
8	8	7900	JAMES	CLERK	7698	1981-12-11 0:00	950	NA	30	FALSE
9	9	7521	WARD	SALESMAN	7698	1981-02-23 0:00	1250	500	30	FALSE

```

10 10 7902 FORD ANALYST 7566 1981-12-11 0:00 3000 NA 20 TRUE
11 11 7369 SMITH CLERK 7902 1980-12-09 0:00 800 NA 20 FALSE
12 12 7788 SCOTT ANALYST 7566 1982-12-22 0:00 3000 NA 20 TRUE
13 13 7876 ADAMS CLERK 7788 1983-01-15 0:00 1100 NA 20 FALSE
14 14 7934 MILLER CLERK 7782 1982-01-11 0:00 1300 NA 10 FALSE

```

설명 : emp 테이블에 sal_tf라는 새로운 컬럼과 데이터를 만들었는데 바로 이 sal_tf는 기존 데이터로 만든 새로운 파생변수입니다.

```

library(gmodels)
CrossTable( emp$job, emp$sal_tf)

```

문제169. (오늘의 마지막 문제 1/22) 중고차 모델별로 색까링 보수적인 색깔이 많은지 아닌지 이원교차표로 확인하시오!

보수적인 색깔: Black, Gray, Silver, White

```

car$col_tf <- car$color %in% c("Black", "Gray", "Silver", "White")
CrossTable( car$model, car$col_tf)

```

		car\$conservative		Row Total
car\$model	FALSE	TRUE		
SE	27	51	78	
	0.009	0.004		
	0.346	0.654	0.520	
	0.529	0.515		
	0.180	0.340		
SEL	7	16	23	
	0.086	0.044		
	0.304	0.696	0.153	
	0.137	0.162		
	0.047	0.107		
SES	17	32	49	
	0.007	0.004		
	0.347	0.653	0.327	
	0.333	0.323		
	0.113	0.213		
Column Total	51	99	150	
	0.340	0.660		

문제170. plotly 패키지로 박스 그래프를 시각화하는데 농구선수 2번과 농구선수 3번을 같이 시각화하시오!

```

x1 = c(7,8,9,9,10,10,11,11,12,13)
x2 = c(7,9,9,10,10,10,11,11,13)

```

```
x3 = c(1,1,7,7,10,10,10,11,13,30)

library(plotly)
subplot(
  add_markers = plot_ly(y=~x2,type='box', name='선수2'),
  add_markers = plot_ly(y=~x3,type='box', name='선수3')
)
```

문제171. 선수 2번의 데이터만 박스 그래프로 시각화 하시오!

```
x2 = c(7,9,9,10,10,10,11,11,13)
```

```
library(plotly)
plot_ly(y=~x2,type='box', name='선수2')
```

설명 : type의 옵션들>>

bar: 막대 그래프

pie : 원형 그래프

scatter : 산포도

histogram: 히스토그램 그래프

문제172. plot_ly로 라인그래프를 그리는데 농구선수 2번의 점수 데이터로 그리시오.

```
x2 = c(7,9,9,10,10,10,11,11,13)
```

```
library(plotly)
plot_ly(y=~x2,type='scatter', name='선수2', mode="dot")
```

설명 : type='scatter', mode="dot"로 하면 라인 그래프로 그려집니다.

문제173. 2019년 특정 기업의 주가데이터를 라인 그래프로 그리시오!

```
stock <- read.csv("000080.csv")
head(stock)

library(plotly)
subplot(
  add_markers = plot_ly(x= ~stock$Date, #날짜
                        y=~stock$Open, # 시가
                        type='scatter', # 그래프의 종류
                        name='주가', # 레전드
                        mode="dot") # 라인 그래프
)
```

문제174. (점심시간 문제) 아래와 같이 별이 출력되게 함수를 생성하시오

```
star(5)
```

```
aaa <- function(x) { for ( i in 1:x ) {print (i) }
    }
```

```
for ( i in 1:5 ) {print ("★" *i) }
```

```
"★" "★" "★"
```

문제175. (빅데이터 기사 문제) 우리나라에 알려진 흡연율이 20%일 경우, 300명을 표본추출하여 300명의 설문 조사를 통하여 관측된 흡연자의 비율이 22.4%인지를 검정한다.

설문결과 비흡연자는 232명이고 흡연자는 68명이라 가정하면,

귀무가설 : "흡연율은 22.4% 이다" 이고

대립가설 : "흡연율은 22.4%가 아니다"이다. 둘중에 어떤 것을 채택해야할지 카이제곱 검정결과로 도출하시오!

```
chisq.test( c(232,68), p=c(0.776, 0.224) )
```

```
data: c(232, 68)
X-squared = 0.012273, df = 1, p-value = 0.9118
      ↑      ↑      ↑
    카이제곱값   자유도   p-value 값
```

답: 카이제곱 검정 결과가 도출된 p-value값이 0.9118로서 0.05보다 크므로 귀무가설을 채택한다. 설문결과 관측된 흡연율을 가정된 흡연율(22.4%)를 따른다고 할 수 있습니다.

문제176. (빅데이터 기사 문제) 다음중 관찰된 빈도가 기대되는 빈도와 유의미하게 다른지 검정하기 위해 사용되는검정으로 가장 옳은 것은? 객관식

1. Z -검정

2. T-검정

3. ANOVA

4. 카이제곱 검정

문제177. 아래의 3차원 데이터의 a와 b 지점 사이의 거리를 구하시오!

```
a = c( 0, 3, 2 )
```

```
b = c( 2, 0, 0 )
```

답: distance(a,b)

4.123106

문제178. 위에서 만든 distance 함수를 이용해서 여러개의 지점과 c(4,4) 지점과의 거리를 각각 비교하시오!
(책 120페이지의 그림 / 구글에서 knn 토마토 검색)

a = c(1,5) 와 토마토 c(4,4) 와의 거리 ?

a = c(2,6) 와 토마토 c(4,4) 와의 거리 ?

a = c(4,5) 와 토마토 c(4,4) 와의 거리 ?

a = c(5,2) 와 토마토 c(4,4) 와의 거리 ?

a = c(6,3) 와 토마토 c(4,4) 와의 거리 ?

a = c(1,7) 와 토마토 c(4,4) 와의 거리 ?

```
x = c( 1,2,4,5,6,1 )
y = c( 5,6,5,2,3,7 )
```

```
temp <- c( ) #거리를 담을 변수
for ( i in 1:6 ) {
    temp <- append( temp, distance( c(x[i], y[i]) , c(4,4) ) )
}
temp
```

결과:

```
3.162278 2.828427 1.000000 2.236068 2.236068 4.242641
```

문제179. 위의 거리중에서 가장 작은 값을 출력하시오!

답: min(temp)

문제180. 토마토와 가장 가까운 거리에 있는 음식 종류가 무엇인지 출력하시오!

```
fruits <- data.frame(
    '재료'= c('사과', '베이컨', '바나나', '당근', '셀러리', '치즈'),
    '단맛' = c( 10, 1, 10, 7, 3, 1),
    '아삭한맛' = c( 9, 4, 1, 10, 10, 1),
    '음식종류' = c('과일', '단백질', '과일', '채소', '채소', '단백질; ')
)
```

View(fruits)

토마토 = c(6,4)

답:

```
temp <- c( ) #거리를 담을 변수
for ( i in 1:6 ) {
    temp <- append( temp, distance( c( fruits$단맛[i], fruits$아삭한맛[i] ) , 토마토 ) )
}
```

temp

결과:

```
6.403124 5.000000 5.000000 6.082763 6.708204 5.830952
```

```
fruits$dist <- temp
View( fruits )
```

문제181. 위에서 만든 fruits의 파생변수인 dist를 이용해서 거리에 대한 순위 파생변수를 추가하시오!

```
library(dplyr)
fruits$rnk <- dense_rank( fruits$dist )
fruits
```

문제182. 위의 결과에서 순위를 3위까지만 출력하시오!

```
fruits[ fruits$rnk <= 3, '음식종류' ]
```

문제183. 위의 결과중에서 최빈값을 출력하시오!

```
a <- fruits[ fruits$rnk <= 3, '음식종류' ]
table(a)[ table(a) == max( table(a) ) ]
```

설명 : 여기서 숫자 3이 바로 knn의 k의 개수가 된다.

문제184. (오늘의 마지막 문제 1/25) 아래의 결과를 캡쳐해서 올리세요.

```
x <- data.frame( 실제=wbcd_test_label, '예측'=result1 )
table(x)
```



설명 : shuffle이 다 다르기 때문에 위의 결과도 자리마다 다 다르다.

문제185. 다음은 분석 모형을 정의할 때 설정하는 것이다. 이를 설명한 것으로 가장 적절한 것은 무엇인가?
(수제비 책 3-24)

- * 모델 내부에서 확인이 가능한 변수로 데이터를 통해서 산출이 가능한 값이다.
- * 예측을 수행할 때 모델에 의해 요구되는 값들이다.
- * 주로 예측자에 의해 수작업으로 측정되지 않는다.

- | | |
|-------------------|-------------|
| 1. 신경망 학습에서의 학습률 | 2. 파라미터 |
| 3. 신경망 학습의 배치 사이즈 | 4. 정규화 파라미터 |

정답 : 2번

문제186. seed값을 설정한 상태에서 모든 자리에서 동일한 결과가 나오도록 모델을 생성하시오!

1. 데이터를 로드합니다.

```
wbcd <- read.csv("wisc_bc_data.csv", header=T, stringsAsFactors=FALSE)
```

```
wbcd$diagnosis <- factor(wbcd$diagnosis,
                           levels =c("B","M"),
```

```
labels = c("Benign","Malignant")
```

2. 데이터를 shuffle 합니다.

```
set.seed(1)  
wbcn_shuffle <- wbcn[sample(nrow(wbcn)), ]
```

#3. 환자번호를 제외합니다.

```
wbcn2 <- wbcn_shuffle[-1]
```

4. 데이터를 min/max 정규화 합니다. (0~1사이의 숫자로 다 변환합니다.)

```
normalize <- function(x) {  
  return ( (x-min(x)) / (max(x) - min(x)) )  
}
```

```
wbcn_n <- as.data.frame(lapply(wbcn2[2:31],normalize))
```

5. 훈련데이터와 테스트 데이터를 9대 1로 나눕니다.

```
train_num<-round(0.9*nrow(wbcn_n),0)  
wbcn_train<-wbcn_n[1:train_num,  
wbcn_test<-wbcn_n[(train_num+1):nrow(wbcn_n),]
```

```
wbcn_train_label <- wbcn2[1:train_num,1]  
wbcn_test_label <- wbcn2[(train_num+1):nrow(wbcn_n),1]
```

```
wbcn_test_label
```

6. 모델을 훈련시켜 테스트 데이터의 라벨을 예측합니다.

```
library(class)  
set.seed(1)  
result1 <- knn(train=wbcn_train, test=wbcn_test, cl=wbcn_train_label, k=21)
```

#7. 테스트 데이터에 대한 실제값과 예측값을 확인합니다.

```
x <- data.frame('실제'=wbcn_test_label, '예측'=result1)  
table(x)
```

k = 2 일때 :

예측

실제	Benign	Malignant
Benign	33	1
Malignant	0	23

k = 21일때 :

예측

실제	Benign	Malignant
Benign	34	0
Malignant	0	23

설명 : seed값과 k 값이 결국 무엇일까? 하이퍼 파라미터 / / seed 값은 사용자가 알아내야 함

문제187. 아이리스의 종, 아이리스 종별 건수를 출력하시오!

```
table(iris$Species)
```

결과:

setosa	versicolor	virginica
50	50	50

문제188. `createDataPartition` 함수를 이용해서 아이리스 데이터를 훈련 데이터와 테스트 데이터로 나누시오!

훈련 데이터는 80%, 테스트 데이터는 20%로 하세요!

(훈련 데이터는 `train_data`, 테스트 데이터는 `test_data`로 하시오!)

```
train_num <- createDataPartition(iris$Species, p=0.8, list=F)
train_data <- iris[train_num,]
test_data <- iris[-train_num,]
```

설명 : `-train_num`은 `train_num`에 있는 숫자 120개 말고 다른 숫자들이라는 뜻이다.

```
nrow(train_data) #120
nrow(test_data) #30
```

```
table(iris$Species)
```

```
table(train_data$Species)
table(test_data$Species)
```

문제189. (점심시간 문제) 위에서 아이리스 데이터를 `createDataPartition` 함수로 훈련데이터와 테스트 데이터로 잘 나누었으므로 바로 오스트리아 빈 대학교에서 만든 `knn` 함수에 입력하여 테스트 데이터의 라벨을 예측하고 이원 교차표 출력을 하시오~ (k 값을 직접 지정하세요~)

iris 데이터 파악

```
head(iris)
nrow(iris)
unique(iris$Species)
```

정규화

```
normalize <- function(x) {
  return ((x-min(x)) / (max(x) - min(x)))
}
```

```
iris2 <- as.data.frame(lapply(iris[1:4], normalize))
summary(iris2)
```

셔플 (여기선 따로 안해줘도 되긴 함)

```
set.seed(62)
```

```
iris_shuffle <- iris[sample(nrow(iris2)),]  
iris_shuffle
```

테스트 와 트레인 셋 구분 caret 이용

```
library(caret)
```

```
train_num <- createDataPartition(iris_shuffle$Species, p=0.8, list=F) # 여기서 자동 셔플이 되긴 함  
str(train_num)
```

```
train_data <- iris2[train_num,]  
test_data <- iris2[-train_num,]
```

```
nrow(train_data)  
nrow(test_data)
```

라벨 지정 (정규화 되기전의 것을 이용한다.)

```
iris_train_label <- iris[train_num,5]  
iris_test_label <- iris[-train_num,5]
```

```
iris_test_label
```

class 를 이용한 knn

```
library(class)  
result1 <- knn(train=train_data, test=test_data, cl=iris_train_label, k=7)
```

※ 주의사항: knn 함수 실행시 train=, test=에 데이터 프레임 넣을 때 그 데이터 프레임에 전부 숫자만 있어야 함

결과 확인

```
x <- data.frame('실제'=iris_test_label, '예측'=result1)  
table(x)
```

#이원분석표

```
library(gmodels)  
g2 <- CrossTable(x=iris_test_label, y=result1 )  
g2$prop.tbl  
print( g2$prop.tb[1] + g2$prop.tb[5] + g2$prop.tb[9] ) # 정확도
```

문제190. wine 데이터를 R로 불러와서 전체 건수가 몇건인지 확인하시오!

```
> nrow(wine)  
[1] 178
```

```
> summary(wine)
```

라벨(정답)이 되는 컬럼은 Type이고 나머지는 다 숫자형 데이터입니다.

문제191. 세원씨가 knn 머신러닝 모델을 만든 순서대로 똑같이 따라서 와인 데이터 분류 모델을 생성해 봅니다.

#1. iris 테이블 파악

```
head(wine)
nrow(wine)
unique(wine$Type)
```

데이터 설명: <https://codedragon.tistory.com/9480> 참조

#2. 정규화

```
normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x)) ) }
```

```
wine2 <- as.data.frame(lapply(wine[2:14],normalize))
summary(wine2)
```

#3. 셔플 (여기선 따로 안해줘도 되긴 함)

```
set.seed(62)
wine_shuffle <- wine[sample(nrow(wine2)),]
wine_shuffle
```

#4. 테스트 와 트레이닝 셋 구분 caret 이용 (8대 2로 훈련과 테스트를 나눈다)

```
library(caret)
set.seed(62)
train_num <- createDataPartition( wine_shuffle$type, p=0.8, list=F) # 여기서 자동 셔플이 되긴 함
str(train_num)

train_data <- wine2[train_num,]
test_data <- wine2[-train_num,]

nrow(train_data) #144
nrow(test_data) #34
```

5. 라벨 지정 (정규화 되기전의 것을 이용한다.)

```
wine_train_label <- wine[train_num,1]
wine_test_label <- wine[-train_num,1]

wine_test_label
```

#6. class 를 이용한 knn

```
library(class)
result1 <- knn(train=train_data, test=test_data, cl=wine_train_label, k=7) #유클리드 거리공식
```

#7. 결과 확인

```
x <- data.frame('실제'=iris_test_label, '예측'=result1)
table(x)
```

예측

실제 t1 t2 t3

t1	12	0	0
t2	0	10	0
t3	0	0	12

#8. 이원분석표

```
library(gmodels)
g2 <- CrossTable(x=wine_test_label, y=result1 )
g2$prop.tbl
print( g2$prop.tb[1] + g2$prop.tb[5] + g2$prop.tb[9] ) # 정확도
```

문제192. (오늘의 마지막문제 1/26) 라플라스 값 0.0004의 경우 0.9916297의 정확도를 보이는 모델입니다. 조금이라도 정확도를 올릴 수 있는 라플라스값을 알아내어 라플라스를 지정하고 정확도 확인 부분 코드와 함께 올리시오.

```
model3 <- naiveBayes(type~ . , data=mushrooms_train, laplace=0.00005)
result3 <- predict( model3, mushrooms_test[ , -1] )
g4 <- CrossTable( mushrooms_test[ ,1], result3)
g4$prop.tbl
print( g4$prop.tb[1] + g4$prop.tb[4] ) # 정확도
```



문제193. 나이가 20대이고 성별이 여자이며 직업이 IT이고 결혼을 안했으며 이성친구가 없는 사람이 선택할 가능성이 높은 영화 장르는?

```
test_data2 <- data.frame( age='20대', gender='여', job='IT', marry='NO', friend='NO' )
test_data2
```

```
result2 <- predict( model2, test_data2 ) # 로맨스
```

문제194. 직업이 학생이고 결혼을 안했으며 이성친구가 없는 20대 남자가 선호하는 영화 장르는?

```
test_data3 <- data.frame( age='20대', gender='남', job='학생', marry='NO', friend='NO' )
```

```
result3 <- predict( model2, test_data3 ) # 코미디
```

문제195. 독감 데이터를 가지고 독감환자인지 아닌지를 판별하는 머신러닝 모델을 생성하시오!

독감 데이터 로드

```
flu <- read.csv("flu.csv", header=T, stringsAsFactors=TRUE)
View(flu)
nrow(flu)
```

```
flu2 <- flu[ , -1]
flu2

#훈련데이터와 테스트 데이터 생성
train_data <- flu2[ 1:7, ]
test_data <- flu2[8, ]

# 나이브베이즈 모델 생성
library(e1071)

View( train_data[ , 1:4 ] )
model <- naiveBayes( train_data[ , 1:4 ], train_data$flue, laplace=0 )
```

#데이터를 가지고 예측
result <- predict(model, test_data[, 1:4])
result

#실제값 확인

```
test_data[, 5]
```

또는-----

#독감 데이터 로드
flu <- read.csv("flu.csv", header=T, stringsAsFactors=TRUE)
View(flu)
nrow(flu)

#훈련데이터와 테스트 데이터 생성

```
train_data <- flu[ 1:7, -1 ] #환자 id가 첫번째여서 -1로 환자 id를 제외함
test_data <- flu[8, -1 ]
```

나이브베이즈 모델 생성

```
library(e1071)
```

```
View( train_data[ , 1:4 ] )
model <- naiveBayes( train_data[ , 1:4 ], train_data$flue, laplace=0 )
# 설명 : train_data[ , 1:4 ]는 오한, 콧물, 두통, 열 컬럼만 가져오라는 뜻
# naiveBayes( 정답없는 훈련 데이터, 정답, 라플라스 값 )
```

#데이터를 가지고 예측

```
result <- predict( model, test_data[ , 1:4] )
# 설명 : test_data[ , 1:4] 는 오한, 콧물, 두통, 열 컬럼만 가져오라는 뜻
```

#예측값 확인

```
result
```

#실제값 확인

```
test_data[, 5 ]
```

문제196. 위의 독감환자 여부를 알아내는 모델인 **model**을 잘 활용할 수 있도록 R 함수를 생성하시오!

<보기>

naive_fun()

오한이 있습니까? Y

콧물이 있습니까? N

두통이 있습니까? MILD

열이 있습니까? N

독감환자가 아닙니다.

답 :

```
naive_func <- function() {  
    library(e1071)  
    flu <- read.csv("flu.csv", header=T, stringsAsFactors = TRUE)  
    train_data <- flu[ 1:8, -1 ]  
    model <- naiveBayes( train_data[ , 1:4 ], train_data$flue, laplace=0 )  
  
    #환자 증상을 물어본다.  
    v_chills <- readline('오한이 있나요? (Y/N) ')  
    v_runny <- readline('콧물이 있나요? (Y/N) ')  
    v_headaches <- readline('두통이 있나요? (STRONG/MILD/NO) ')  
    v_fever <- readline('열이 있나요? (Y/N) ')  
  
    test <- data.frame( chills=v_chills, runny_nose=v_runny, headache=v_headaches, fever=v_fever)  
  
    result <- predict(model, test)  
    print(result)  
  
    if (result=='Y'){print ('독감 환자 입니다')}  
    else {print('독감 환자가 아닙니다')}  
  
}
```

naive_func()

문제197. (점심시간 문제) 영화장르를 예측하는 나이브 베이즈 모델을 이용해서 아래와 같이 질문을 하고 결과를 출력하는 **movie_fun** 함수를 생성하시오!

moive_fun()

성별을 입력하세요(여/ 남) :

나이대를 입력하세요(20대/ 30대/ 40대) :

직업을 입력하세요 (IT/ 디자이너/ 무직/ 언론/ 영업/ 자영업/ 학생/ 홍보마케팅) :

결혼 여부를 입력하세요 (YES/ NO) :

이성친구 여부를 입력하세요 (YES/ NO) :

답:

```
movie_fun <- function() {  
    library(e1071)  
    movie <- read.csv( "movie.csv", header=T, stringsAsFactors=TRUE )
```

```

colnames(movie) <- c( "age", "gender", "job", "marry", "friend", "m_type" )
train_data <- movie[ 1:39, ]
model2 <- naiveBayes( train_data[ , 1:5 ], train_data$m_type, laplace=0 )

#고객 정보에 대한 질문
gender_q <- readline('성별을 입력하세요( 여/ 남 ) : ')
age_q <- readline('나이대를 입력하세요( 20대/ 30대/ 40대 ) : ')
job_q <- readline('직업을 입력하세요 ( IT/ 디자이너/ 무직/ 언론/ 영업/ 자영업/ 학생/ 홍보마케팅) : ')
marriage_q <- readline('결혼 여부를 입력하세요 ( YES/ NO ) : ')
friend_q <- readline('이성친구 여부를 입력하세요 ( YES/ NO ) : ')

test <- data.frame( age=age_q, gender=gender_q, job=job_q, marry=marriage_q, friend=friend_q)
result1 <- predict(model2, test)
print(paste('좋아하는 장르는', result1, '입니다'))

}

movie_fun()

```

문제198. 지방간을 일으키는 원인중에 가장 큰 영향력을 보이는 요인은 무엇인지 정보획득량을 구해서 알아내시오!

```

fat <- read.csv("fatliver2.csv", header=T, stringsAsFactors=T )
View(fat)
wg <- information.gain(FATLIVER ~. , fat, unit='log2')
wg

```

결과:

	attr_importance
AGE	0.032256902
GENDER	0.028650604
DRINK	0.012189492
SMOKING	0.009812076

나이가 가장 정보획득량이 큼

문제199. skin의 의사결정트리의 성능을 높이시오!

knn일 때의 하이퍼 파라미터 : k값, seed값
naivebayes일 때 하이퍼 파라미터 : laplace 값, seed 값
decision tree 일 때 하이퍼 파라미터 : trials 값, seed 값

trials 파라미터의 나무의 개수를 정한다. 여러개의 나무를 만들어서 그 나무들 중 가장 분류를 잘하는 나무를 투표를 통해서 선택한다.

```

library(C50)
skin_model <- C5.0(skin2_train[ , -6], skin2_train$cupon_react, trials=10 )

```

```
skin2_result <- predict( skin_model2 , skin2_test[ , -6])
```

```
library(gmodels)
CrossTable( skin2_test[ , 6], skin2_result )
```

데이터가 너무 작아서 아래와 같은 메세지가 나옴
Number of boosting iterations: 10 requested; 1 used due to early stopping

씨드값 20으로 변경-----

백화점 화장품 고객 데이터를 로드하고 shuffle 한다.

```
skin <- read.csv("skin.csv", header=T ,stringsAsFactors = TRUE)
nrow(skin)
```

skin_real_test_cust <- skin[30,] #테스트용으로 따로 분리한다.

```
skin2 <- skin[ 1:29, ] #29개의 데이터로 학습시켜서 의사결정나무를 생성
nrow(skin2)
```

skin_real_test_cust

결과 :

cust_no gender age job marry car cupon_react

30 30 female 40 YES YES NO YES

skin2 <- skin2[, -1] # 고객번호를 제외시킨다.

```
set.seed(20)
skin2_shuffle <- skin2[sample(nrow(skin2)), ] # shuffle 시킴
```

화장품 고객 데이터를 7대 3로 train 과 test 로 나눈다.

```
train_num <- round(0.7 * nrow(skin2_shuffle), 0)
skin2_train <- skin2_shuffle[1:train_num, ]
skin2_test <- skin2_shuffle[(train_num+1) : nrow(skin2_shuffle), ]
```

```
nrow(skin2_train) # 20
nrow(skin2_test) # 9
```

C50 패키지를 이용해서 분류 모델을 생성한다.

```
library(C50)
skin_model2 <- C5.0(skin2_train[ , -6], skin2_train$cupon_react, trials=10 )
```

```
skin2_result <- predict( skin_model2 , skin2_test[ , -6])
```

```
>skin2_result
```

```
NO YES NO YES NO NO NO NO NO  
Levels: NO YES
```

위에서 만든 skin_model 를 이용해서 테스테 데이터의 라벨을 예측

```
skin2_result <- predict( skin_model , skin2_test[ , -6])
```

이원 교차표로 결과를 확인

```
library(gmodels)  
CrossTable( skin2_test[ , 6], skin2_result )
```

결과:

		skin2_result		Row Total
skin2_test[, 6]		NO	YES	
NO	NO	5	0	5
	0.317	1.111		
	1.000	0.000	0.556	
	0.714	0.000		
	0.556	0.000		
	YES	2	2	4
YES	0.397	1.389		
	0.500	0.500	0.444	
	0.286	1.000		
	0.222	0.222		
	Column Total	7	2	9
		0.778	0.222	

문제200. (오늘의 마지막 문제 1/27) 어제 사용했던 와인 데이터를 분류하는 의사결정트리 모델을 생성하는 실험을 하시오~

데이터 : wine.csv

의사결정트리 패키지 : party 패키지의 ctree함수를 이용하세요! (코드, 이원교차표 올리기)

#party 불러오기

```
library(party)
```

#데이터 불러오기

```
wine <- read.csv("wine.csv", header=T, stringsAsFactors = T)  
View(wine)
```

#전체 행 개수 확인

```
nrow(wine) #178
```

#통계정보 확인

```
summary(wine)
```

#라벨컬럼 확인

```
table(wine$Type)
```

결과:



#훈련 데이터와 테스트 데이터를 만들

```
set.seed(60)
```

```
ind <- sample(2,nrow(wine), replace=T, prob=c(0.8,0.2))
```

```
traindata <- wine[ind==1,]
```

```
testdata <- wine[ind==2,]
```

```
nrow(traindata) #140
```

```
nrow(testdata) #38
```

의사결정트리 모델(나무) 생성

```
wine_ctree <- ctree(Type ~.,data=traindata)
```

#예측

```
predict(wine_ctree)
```

예측결과와 실제 테스트 데이터의 정답과 비교해 봄

```
table(predict(wine_ctree),testdata$Type)
```

결과 :



테스트 데이터 예측하고 잘 맞췄는지 확인

```
testpred <- predict(wine_ctree, newdata=testdata)
```

```
table(testpred,testdata$Type)
```

결과:



```
library(gmodels)
```

```
CrossTable(testpred,testdata$Type)
```

```
testpred
```

이원 교차표

```
library(gmodels)
```

```
CrossTable(testdata$Type, testpred)
```



×

문제201. 숫자 1번부터 10번까지의 숫자들을 주머니속에 넣고 랜덤추출하는데 9번 랜덤추출해서 결과를 출력하시오! (복원 추출)

```
set.seed(1)
sample(c(1:10), size=9, replace=TRUE )
```

문제202. 1번부터 10번까지의 숫자들을 주머니속에 넣고 9번 랜덤 추출하는데 비복원 추출하시오!

```
set.seed(1)
sample(c(1:10), size=9, replace=FALSE )
```

문제203. 1번부터 10번까지의 숫자들을 주머니속에 넣고 10번 랜덤추출하는데 비복원 추출하시오!

```
set.seed(1)
sample( c(1:10), size=10, replace=FALSE )
```

설명 : 비복원 추출이란 꺼냈던 숫자를 다시 주머니 속에 넣지 않고 계속 추출하는 것이다.
과연 아래의 코드는 실행이 될까?

```
set.seed(1)
```

```
sample( c(1:10), size=11, replace=FALSE )
```

---> 안된다.

Error in sample.int(length(x), size, replace, prob) :
'replace = FALSE' 일때는 모집단보다 큰 샘플을 가질 수 없습니다

근데 같은 코드로 복원 추출은 가능할까?

```
set.seed(1)  
sample( c(1:10), size=11, replace=TRUE )
```

---> 가능하다.

결과 : [1] 3 1 5 5 10 6 10 7 9 5 5

문제204. 아래의 코드를 해석하시오!

```
set.seed(11)  
ind <- sample( 2 , nrow(iris) ,replace=T, prob=c(0.7,0.3) )
```

답 : 위의 코드를 다시쓰면 아래와 같다.

```
set.seed(11)  
sample( c(1,2) , 150,replace=T )
```

설명 : 주머니 속 숫자 1과 2를 150번 복원 추출 한다.

이번에는 prob옵션을 주고 다시 수행한다.

```
ind <- sample( c(1,2) , 150 ,replace=T, prob=c(0.7,0.3) )
```

설명 : 주머니 속에 있는 숫자 1과 2를 150번 복원 추출하는데 숫자 1은 70%의 확률로 복원추출하고 숫자 2는 30%의 확률로 복원추출해라 라는 뜻

문제205. 아래의 코드로 수행해서 출력 된 150개의 숫자 1과 2가 각각 몇개씩 있는지 카운트 하시오!

```
sample( c(1,2) , 150 ,replace=T, prob=c(0.7,0.3) )
```

답:

```
ind <- sample( c(1,2) , 150 ,replace=T, prob=c(0.7,0.3) )
```

```
table(ind)
```

결과:

1	2
120	30

문제206. 위의 1과 2의 비율을 출력하시오!

```
prop.table(table(ind))
```

결과:

1 2
0.6933333 0.3066667

※ 만약 확률이 아니라 비율로 정확하게 7대 3으로 나눠지게 하고 싶다면?

문제207. 150개의 아이리스 데이터에서 2번째 행과 5번째 행과 73번째 행과 121번째 행을 출력하시오!

iris[c(2, 5, 73, 121),]

문제208. 아래의 변수에 담은 ind 변수에서 숫자 1이 몇 개 있는지 확인하시오!

ind <- sample(c(1,2), 150, replace=T, prob=c(0.7,0.3))

답 : ind==1

결과:

```
[1] TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE FALSE TRUE FALSE  
TRUE TRUE TRUE TRUE FALSE TRUE  
[2] TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE FALSE  
TRUE TRUE TRUE FALSE TRUE FALSE  
[43] TRUE TRUE FALSE FALSE TRUE TRUE TRUE FALSE FALSE TRUE FALSE FALSE TRUE FALSE  
FALSE FALSE TRUE TRUE TRUE TRUE  
[64] TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE  
TRUE TRUE TRUE TRUE FALSE  
[85] FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE FALSE FALSE  
TRUE FALSE FALSE FALSE TRUE TRUE  
[106] TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE  
FALSE FALSE FALSE TRUE TRUE FALSE  
[127] TRUE TRUE FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE  
TRUE TRUE TRUE TRUE TRUE TRUE  
[148] TRUE TRUE TRUE
```

결과를 보면 1인것은 TRUE이고 아닌것은 FALSE로 출력되고 있음.

그래서 TRUE인 것만 SUM 하면 된다.

sum(ind==1) #104개

문제209. 아이리스 데이터에서 위의 150개의 TRUE, FALSE 중에 TRUE인 것의 행만 출력되게 하시오!

```
iris[ind==1, ]  
nrow(iris[ind==1, ]) #104개
```

문제210. 아이리스 데이터에서 ind가 2인 자리의 행들만 출력하시오!

```
iris[ind==2, ]  
nrow(iris[ind==2, ])
```

문제211. 아이리스 데이터를 검색하는데 Sepal.Length (꽃받침의 너비)가 5.0이상인 꽃들만 출력하시오!

```
iris[iris$Sepal.Length >= 5.0, ]
```

문제212. 은행 채무 불이행자를 예측하는 머신러닝 모델을 의사결정트리 알고리즘으로 구현했는데 엔트로피 지수로 순수도를 계산하고 정보획득량을 계산하고 C50패키지로 모델을 생성하였다. 이번에는 party 패키지의 ctree함수를 이용해서 의사결정 분류모델을 만들어보는 실험을 하시오!

답 :

■ party 패키지를 이용하여 채무불이행자 예측 알고리즘 구현

1. party 패키지를 불러옴

```
library(party)  
library(gmodels)
```

2. 데이터를 불러옴

```
credit <-read.csv("credit.csv", stringsAsFactors=TRUE)  
str(credit)
```

#설명 : 문자형이 factor로 변환이 된것을 확인

3. 데이터에 대한 정보를 확인

```
nrow(credit) # 전체행이 1000개  
ncol(credit) # 컬럼은 모두 17개  
summary( credit )
```

※ 중요 : 데이터에 결측치가 있는지 반드시 확인할 것

```
colSums( is.na(credit) )
```

#결과:

```
#checking_balance months_loan_duration credit_history purpose amount savings_balance  
# 0 0 0 0 0 0  
# employment_duration percent_of_income years_at_residence age other_credit housing  
# 0 0 0 0 0 0  
# existing_loans_count job dependents phone default  
# 0 0 0 0 0 0
```

#설명 : 결측치가 없는 예쁜 데이터이다.

결측치가 있다면 결측치를 다른 데이터 치환하거나 삭제 해야한다.

4. 은행 데이터의 종속변수 컬럼의 종류와 건수를 확인

```
table(credit$default)
```

결과:

```
no yes  
700 300
```

#5. 데이터를 shuffle 시킵니다.

```
set.seed(659)  
credit_shuffle <- credit[ sample(1000), ]  
nrow(credit_shuffle)
```

#6. 데이터를 9대 1로 나눔. (훈련 데이터: 9, 테스트 데이터 : 1)

```
train_num <- 0.9 * 1000  
credit_train <- credit_shuffle[ 1:train_num, ] #1번부터 900번째행까지  
credit_test <- credit_shuffle[ (train_num+1) : nrow(credit_shuffle), ]  
# 901번부터 1000번까지는 테스트 데이터로 구성  
  
nrow(credit_train) #900  
nrow(credit_test) #100
```

7. 의사결정트리 모델을 생성 (party 패키지의 ctree 함수 이용)

```
myformula <- default ~ .  
credit_ctree <- ctree(myformula, data=credit_train)
```

8. 훈련데이터에 대해서 모델이 예측한 결과를 확인

```
predict(credit_ctree)  
length(predict(credit_ctree)) #900
```

9. 예측 결과와 실제 훈련 데이터의 정답과 비교

```
table(predict(credit_ctree), credit_train$default)
```

no	yes
no	524 136
yes	100 140

#설명: 훈련데이터(90%) 를 가지고 잘 맞췄는지 확인, 900개중에 236개 틀렸습니다.

10. 어떠한 질문을 기준으로 나뉘는지 확인 (스무고개 질문들 확인)

```
print(credit_ctree)  
print(credit_ctree, type='simple')
```

11. 테스트 데이터를 예측하고 잘 맞췄는지 확인

```
test_pred <- predict(credit_ctree, newdata=credit_test)
```

```
table(test_pred, credit_test$default)
```

결과:

no	66	9
yes	10	15

#설명: 테스트 데이터도 100개중에 19개나 틀렸습니다.

12. 교차검정표와 정확도 계산

```
g2 <- CrossTable(credit_test$default, test_pred)
```

#설명: CrossTable(테스트 데이터의 진짜 정답, 테스트 데이터를 기계가 예측한 결과)

```
sum(g2$prop.tbl*diag(2)) #0.81
```

13. 시각화를 합니다.

```
plot(credit_ctree)  
plot(credit_ctree, type='simple')
```

설명 : 의사결정트리 패키지 2가지

1. C50 : 엔트로피 지수로 정보획득량을 구해서 의사결정나무 구성
성능 개선 하이퍼 파라미터 : trials와 seed 값
2. party : 카이제곱검정으로 정보획득량을 구해서 의사결정나무 구성
성능 개선 하이퍼 파라미터 : seed값,

C50은 trials를 이용해서 바로 성능개선을 할 수 있는게 장점이고
party는 의사결정트리를 시각화 할 수 있는 장점이 있다.

문제213. (오늘의 마지막문제 1/28) party 패키지를 이용하여 채무불이행자 예측 알고리즘을 구현한 코드의 seed 값을 건준이가 알려준 코드를 이용해서 정확도 0.86일때 seed 값이 무엇인지 출력하시오!)

party 사용

```
library(party)
```

credit.csv 로드

```
credit <-read.csv("credit.csv", stringsAsFactors=TRUE)
```

#팩터로 바뀌었는지 확인

```
str(credit)
```

#repeat문 생성

```
i <- 1  
repeat {  
  set.seed(i)  
  i <- i+1  
  credit_shuffle <- credit[ sample(1000), ]  
  nrow(credit_shuffle)}
```

```
train_num <- 0.9 * 1000
```

```
credit_train <- credit_shuffle[ 1:train_num, ] #1번부터 900번째행까지 train 데이터셋에 담기
```

```
credit_test <- credit_shuffle[ (train_num+1) : nrow(credit_shuffle), ] #나머지를 test 데이터 셋에 담기
```

#의사결정트리 생성

```
credit_ctree <- ctree(default ~., data=credit_train)
```

예측 결과 확인

```
predict(credit_ctree)
```

```
#검증용 데이터를 예측하고 잘 맞췄는지 확인
```

```
test_pred <- predict(credit_ctree, newdata=credit_test)
```

```
#이원교차표의 형태 확인하고 g3에 넣기
```

```
library(gmodels)
```

```
CrossTable(credit_test$default, test_pred)
```

```
g3 <- CrossTable(credit_test$default, test_pred)
```

```
# 이원교차표 형태를 보고 대각선의 값을 x에 넣어주기
```

```
x<- (g3$prop.tb[1] + g3$prop.tb[4])
```

```
#조건에 만족하면 return문이 끝나게 설정
```

```
if(x>=0.87) break
```

```
print(i) }
```

```
#seed값 찾기
```

```
print(i) #4369
```

```
#정확도 계산
```

```
sum(g3$prop.tbl*diag(2)) #0.87
```

```
#plot 그리기
```

```
plot(credit_ctree)
```

```
plot(credit_ctree, type='simple')
```

```
# seed가 4369일때 정확도 0.87
```

문제214. OneR로 구한 독버섯모델의 정확도를 출력하세요.

```
답 : g1 <- CrossTable( mushroom_test[, 1], result1 )
```

```
print( sum(g1$prop.tbl * diag( 2 )) ) #0.9862137
```

문제215. w1을 다시 출력하는데 정보획득량이 높은것부터 출력되게하시오!

```
답 : library( doBy )
```

```
orderBy( ~ -attr_importance, w1 )
```

문제216. 와인 데이터를 Riper 알고리즘으로 분류하는 머신러닝 모델을 생성하고 정확도를 확인하세요!

```

wine <- read.csv( "wine.csv", header=T, stringsAsFactors = T)
str(wine)
View(wine)

i <- 1
repeat{
  set.seed(i)
  i <- i+1

  dim(wine) # 178 14
  train_cnt <- round( 0.75 * dim(wine)[1])
  train_index <- sample(1:dim(wine)[1], train_cnt, replace=F)
  wine_train <- wine[train_index, ] # 134
  wine_test <- wine[-train_index, ] #44

  nrow(wine_train) #134
  nrow(wine_test) #44

model3 <- JRip(Type~ ., data=wine_train)
model3

summary(model3)
result3 <- predict( model3, wine_test[ , -1] )
library(gmodels)
g2 <- CrossTable( wine_test[, 1], result3)

x <- sum(g2$prop.tbl*diag(3))

```

```

if(x==1)break
print(i)}

```

```

sum(g2$prop.tbl*diag(3))

```

신성이의 코드 :

```

##0. repeat
accuracy <- c() #정확도를 담기 위한 벡터 생성
seed <- c() # seed 값을 담기 위한 벡터 생성
i <- 1      # seed값을 1번부터 시작
repeat {    # repeat로 중괄호 안의 코드를 반복시킴
  set.seed(i) # seed값을 제일 처음에는 1번으로 지정
  seed <- append(seed, i) # seed값을 벡터에 append 시킴

```

```

##1. Load data
wine <- read.csv("wine.csv", header=T, stringsAsFactor=T)

##2. Check data

```

```

str(wine)  #'data.frame': 178 obs. of 14 variables:
#label column : $ Type      : Factor w/ 3 levels "t1","t2","t3": 1 1 ...

##3. set train_data : test_data = 0.75 : 0.25
library(caret)
#set.seed(49)  #hyper parameter
k <- createDataPartition(wine$type, p=0.75, list=F) #75%로 비복원추출하면서 분리
train_data <- wine[k, ] # 훈련 데이터 75%
test_data <- wine[-k, ] # 테스트 데이터 25%

```

##4. Make model / pred

```

library(RWeka)
model <- JRip(Type~., data=train_data)
summary(model)

```

##5. Pred with test_data : CrossTable

```
res <- predict(model, test_data[, -1]) # 테스트 데이터 예측
```

```

library(gmodels)
g <- CrossTable(test_data[, 1], res)
x <- sum(g$prop.tbl * diag(3))  # 정확도 확인 코드
x  #[1] 1
#-----
## repeat
accuracy <- append(accuracy, x)
temp <- data.frame(seed, accuracy) # 시드값과 정확도를 데이터 프레임으로 구성
print(i)
print(x)
if (x==1 | i==1000) break # 정확도가 1이거나 i 값이 1000이면 멈춰라
i <- i+1
}
print(i)  #value : satisfying condition
#=====

## Find hyper parameter
library(dplyr)
temp$rnk <- dense_rank(-temp$accuracy)
temp[temp$rnk==1, ]  # 순위가 1위인 행을 확인한다.
View(temp)

```

문제217. seed 값을 49로 하고 createDataPartition 함수를 사용해서 와인 데이터를 분류하시오 !

```

##1. Load data
wine <- read.csv("wine.csv", header=T, stringsAsFactor=T)

##2. Check data
str(wine)  #'data.frame': 178 obs. of 14 variables:
#label column : $ Type      : Factor w/ 3 levels "t1","t2","t3": 1 1 ...

```

```
##3. set train_data : test_data = 0.75 : 0.25
library(caret)
set.seed(49) #hyper parameter
k <- createDataPartition(wine$type, p=0.75, list=F)
train_data <- wine[k, ]
test_data <- wine[-k, ]
```

```
##4. Make model / pred
library(RWeka)
model <- JRip(Type~., data=train_data)
summary(model)
```

```
##5. Pred with test_data : CrossTable
res <- predict( model, test_data[ , -1])

library(gmodels)
g <- CrossTable( test_data[ , 1], res)
x <- sum(g$prop.tbl *diag(3))
x #[1] 1
```

문제218. (점심시간 문제) iris 데이터를 규칙 기반 알고리즘으로 분류하시오! 가장 적절한 seed값을 신성이의 코드로 알아내시오~

```
> nrow(iris) #150
```

#데이터 로드

```
data(iris)
```

```
str(iris)
View(iris)
```

```
library(caret)
```

#repeat문 돌리기

```
i<-1
repeat{
  set.seed(i)
  i <- i+1

  k <- createDataPartition(iris$Species, p=0.75, list=F)

  train_data <- iris[k, ]
  test_data <- iris[-k, ]

  library(RWeka)
  model <- JRip(Species~., data=train_data)
  summary(model)

  res <- predict( model, test_data[,-5])

  library(gmodels)
  g <- CrossTable( test_data[ , 5], res)
```

```
x <- sum(g$prop.tbl *diag(3))
```

```
if(x==1)break  
print(i)}
```

```
sum(g2$prop.tbl*diag(3))
```

seed가 13일 때 정확도 100%

문제219. 어느 실험실에서 10시간, 20시간, 30시간, 40시간마다 물질의 방사능 수치를 측정한 자료가 있을 때 35시간의 물질의 방사능 수치는 무엇인가? (x축 : 시간, y축 : 방사능 수치)

```
x = c( 10, 20, 30, 40 )
```

```
y = c( 300, 250, 200, 150 )
```

공식 :

$$\sum (y - \hat{y})^2 = \sum e^2$$

$$b = \bar{y} - b\bar{x}$$

```
a = cov( x,y ) / var(x)
```

```
a # -5
```

```
b = y_mean - ( - 5 * x_mean )
```

```
b # 350
```

함수로 구하기

답 :

```
func_nuclear <- function(x_num) {  
  x = c( 10, 20, 30, 40 )  
  y = c( 300, 250, 200, 150 )  
  a = cov(x,y) / var(x) # 기울기  
  x_mean = mean(x)  
  y_mean = mean(y)  
  b = y_mean - a * x_mean  
  y_hat = b + a * x_num # y = b + ax  
  print( y_hat )  
}
```

```
func_nuclear(35) #175
```

문제220. 탄닌 함유량과 애벌레의 성장간의 실험표를 이용해서 탄닌 함유량이 9일때 성장률이 어떻게 되는지 알아내는 함수를 생성하시오! (데이터는 regression.txt 이용)

```
reg_func <- function(x_num) {  
  x = reg$growth
```

```

y = reg$tannin
a = cov(x,y) / var(x) # 기울기
x_mean = mean(x)
y_mean = mean(y)
b = y_mean - a * x_mean
y_hat = b + a * x_num  # y = b + ax
print( y_hat )
}
reg_func(9)

```

문제221. 광고비가 매출에 미치는 영향을 조사하기 위한 회귀분석을 하시오!

고객에게 보고하기 위한 회귀식과 그래프를 시각화 하시오!

(면접질문 : 데이터를 받으면 R로 회귀분석을 할 수 있는가?)

데이터 : simple_hg.csv

cost : 광고비

input : 매출액

```

ad <- read.csv("simple_hg.csv", header = T )
ad

attach(ad)
plot( ad$input ~ad$cost, data=reg, pch=21, col='blue', bg='blue' )

model1 <- lm( input ~ cost, data=reg )
model1

abline( model1, col='red' )

model1$coefficients[2] #기울기
model1$coefficients[1] #절편

title( paste( '매출액 = ', model1$coefficients[2], 'x 광고비 + ', model1$coefficients[1] ) )

```



✗

#잔차그리는 코드

```
yhat <- predict( model1, cost=cost )
join <- function(i)
  lines( c(cost[i], cost[i]), c(input[i], yhat[i]), col="green")
  sapply(1:19, join)
```

설명: 회귀분석에서의 용어 2가지?

1. 오차 : 모집단에서의 실제값이 회귀선과 비교해 볼때 나타나는 차이(정확치와 관측치의 차이)
2. 잔차 : 표본에서 나온 관측값이 회귀선과 비교해 볼 때 나타나는 차이



×

문제222. 우주 왕복선 챌린저호의 폭파원인을 분석하시오.

(예 : 4기 선배가 삼성전자에 파견나가서 했던 일 중 삼성 갤럭시7의 폭파원인을 알아내는 분석이 있음)
우주 왕복선 데이터의 o형링 파손수를 y축으로 두고 온도를 x축으로 두어서 단순회귀분석을 하시오!
(데이터 : challenger.csv)

데이터 소개 : distress_ct : o형링 파손 수

temperature : 온도

field_check_pressure : 압력

flight_num : 비행기 번호

답 :

```
spaceship = read.csv("challenger.csv", header = T)
attach(spaceship)

plot(distress_ct ~ temperature, data=spaceship, pch=21, col='red', bg='red')
m <- lm(distress_ct ~ temperature, spaceship)
abline(m, col='red')

title( paste( 'distress_ct = ', round(m$coefficients[2],4), 'x temperature + ', round(model1$coefficients[1],4) ) )

yhat <- predict(m, cost=temperature)
join <- function(i)
  lines(c(temperature[i],temperature[i]),c(distress_ct[i],yhat[i]), col="green")
sapply(1:19,join)
```



✗

문제223. [빅데이터 기사시험 대비] 나이브베이즈 공식을 이용해서 아래의 문제를 해결하시오!



✗



✗



✗

(오늘의 마지막 문제(2/1)) 나이브 베이즈 공식 문제2



✗

앞 뒤 모두 검은색 : A

앞뒤 각각 검은색, 흰색 : B

앞 뒤 모두 흰색 : C

본 면이 검은색 ---- A -- 다른면 검은색 : 1

검은색 x : 0

---- B -- 다른면 검은색 : 0

검은색 x : 1

---- C -- 다른면 검은색 : 0

다른면 검은색 x : 0

$$\frac{P(\text{검} | A) * P(A)}{P(\text{검} | A) * P(A) + P(\text{검} | A^c) * P(A^c)} = \frac{1 * \frac{1}{3}}{1 * \frac{1}{3} + \frac{1}{4} * \frac{2}{3}}$$

문제224. 챌린저호의 온도와 O형링 파손수간의 상관계수를 구하시오!

```
launch <- read.csv("challenger.csv", header = T)
launch
```

```
cor(launch$temperature, launch$distress_ct)
[1] -0.5111264
```

설명 : 피어슨 상관계수가 -0.51로 음의 상관관계를 보이고 있다. 온도와 O형링 손상간의 상대적인 강도가 -0.51로 최대값인 -1의 절반정도이기 때문에 적당히 강한 음의 선형관계가 있음을 의미한다.

상관계수 값 : -1 ~ 1로 구성, 0은 상관관계가 없음을 의미한다.

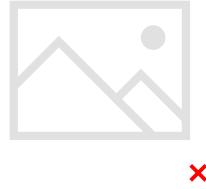
문제225. 챌린저호의 O형링 파손과 상관관계 높은 컬럼은 아래의 3개중 어떤것인지 확인하시오!

1. 온도
2. 압력
3. 비행기 번호(비행기의 노후화와 연관이 있는 번호)

답: cor(launch) #한번에 보는 코드, 정답은 1번 (온도)

문제226. 위의 상관관계를 R로 시각화해서 출력하시오! p. 274

```
install.packages("psych")
library(psych)
pairs.panels(launch)
```



문제227. 삼성전자와 현대자동차 둘중에 코스피 등락율과 더 상관관계가 높은 주식이 어떤건지 알아내시오!

데이터 3가지 : K_index.csv, S_stock.csv, H_stock.csv
(코스피 등락율) (삼성전자) (현대자동차)

```
kospi <- read.csv("K_index.csv", header = T, stringsAsFactors = T)
samsung <- read.csv("S_stock.csv", header = T, stringsAsFactors = T)
hyundai <- read.csv("H_stock.csv", header = T, stringsAsFactors = T)
```

```
> cor(na.omit(kospi$k_rate), na.omit(samsung$s_rate)) # 삼성전자
[1] 0.5142455
> cor(na.omit(kospi$k_rate), na.omit(hyundai$h_rate)) # 현대자동차
[1] 0.3262777
```

문제228. 코스피 등락율과 삼성 수익률 등락율을 plot그래프로 그리고 그 그래프에 회귀직선을 그으시오! (x축 : 코스피 등락율, y축 : 삼성수익 등락율)

```
all_data <- merge(kospi, samsung )
# head(all_data)
attach(all_data)
plot(k_rate, s_rate, col='blue')
model_s <- lm(s_rate ~ k_rate, data=all_data)
abline(model_s, col='red')
```

설명 : 코스피 등락율에 따른 삼성 수익률의 등락율을 예측할 수 있는 회귀모델을 생성함

문제229. 위의 그래프에서 제목에 회귀계수를 사용한 직선의 방정식이 들어가게 하시오!

title(paste('삼성등락률 = ', model_s\$coefficients[2], '코스피등락률 + ', model_s\$coefficients[1])) 추가



×

문제230. 아래의 행렬을 R로 구현하고 아래의 행렬의 전치 행렬을 출력하시오!

```
a <- matrix(c(1,2,3,4,5,6), nrow=3, ncol=2, byrow=T)  
a
```

결과 :

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

전치 : t(a)

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

문제231. 아래의 단위행렬을 만드시오!

```
1 0 0  
0 1 0  
0 0 1
```

```
b <- diag(3)
b
```

문제232. 아래의 행렬의 곱을 R로 구현하시오! (R에서의 행렬곱 --> %*%)

$$\begin{matrix} 1 & 2 & 3 & 1 & 0 & 0 \\ 4 & 5 & 6 & x & 0 & 1 & 0 \\ 7 & 8 & 9 & 0 & 0 & 1 \end{matrix} = ?$$

```
2x3      3x3
a <- matrix(c(1,2,3,4,5,6,7,8,9), nrow=3, ncol=3, byrow=T)
b <- diag(3)
a%*%b
```

설명 : a와 단위행렬을 곱했더니 자기자신(a)이 출력되었습니다.

문제233. 아래의 행렬의 곱의 결과가 무엇인지 실험하시오!

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

```
[,1] [,2]
[1,] 1 0
[2,] 0 1
```

```
a <- matrix( c(1,2,3,4), nrow=2, ncol=2, byrow=T )
b <- diag(2)
```

```
a%*%b
```

설명 : 단위행렬과 곱하면(내적하면) 자기자신이 나온다.

문제234. 아래의 a 행렬의 역행렬을 R로 구하시오!

```
a <- matrix( c(1,2,3,4), nrow=2, ncol=2, byrow=T )
a
```

```
solve(a) # 역행렬구하기
```

결과 :

```
[,1] [,2]
[1,] -2.0 1.0
[2,] 1.5 -0.5
```

설명 : a와 a의 역행렬을 행렬곱하면 "단위행렬"이 출력된다.

round(a %*% solve(a)) 하면 단위행렬이 나온다.

문제235. 아래 행렬a의 역행렬을 구하시오.

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

```
a <- matrix(c(1,2,3,4), nrow=2, ncol=2, byrow=T)
```

```
solve(a)
```

문제236. 아래의 a 행렬을 만들고 아래의 a행렬의 전치행렬을 곱하면 정방행렬이 나오는지 확인하시오!



```
a <- matrix(c(1:12), nrow=3, ncol=4, byrow=T)  
a %*% t(a)
```

결과 :

```
[,1] [,2]  
[1,] 5 11  
[2,] 11 25
```

문제237. 책 264페이지의 가운데에 나오는 식을 R로 구현하시오!



```

reg <- function(y, x){
  x <- as.matrix(x) #행렬로 변경하는 코드
  x <- cbind( Intercept=1, x ) #절편을 추가하는 코드
  b <- solve( t(x) %*% x ) %*% t(x) %*% y # 다중회귀의 회귀 계수 구하는 수학식
  colnames(b) <- "estimate" # 컬럼명을 지정
  print(b)
}

```

문제238. 우주 왕복선 챌린저호의 O형링 파손원인중 가장 영향이 큰 요소가 온도, 압력, 비행기 노후화를 나타내는 비행기 번호를 나타내시오!

```

launch <- read.csv( "challenger.csv", header=T )
launch
reg( y = launch$distress_ct, x=launch[ , 2:4] )

```

결과:

	estimate
Intercept	3.527093383
temperature	-0.051385940
field_check_pressure	0.001757009
flight_num	0.014292843

회귀식: $y = 3.527093383 - 0.051385940 \times X_1 + 0.001757009 \times X_2 + 0.014292843 \times X_3$

설명 : O형링 파손에 영향을 주는 가장 큰 독립변수는 온도이다. 그 다음이 비행기의 노후화를 나타내는 비행기 번호이다.

문제239. 스마트폰 만족도(종속변수)에 영향을 미치는 요소중 가장 영향력이 있는 독립변수는 무엇인가?
(데이터 : multi_hg.csv)

```
reg(y=multi_hg$만족감, x=multi_hg[1:3])
```

	estimate
Intercept	3.5136006
외관	0.2694261
편의성	0.2105249
유용성	0.1623154

```

attach( m )
lm( 만족감 ~외관 + 편의성 + 유용성, data=m )

```

결과 :

Coefficients:

(Intercept)	외관	편의성	유용성
3.5136	0.2694	0.2105	0.1623

문제240. 미국 대학교 입학에 가장 크게 영향을 미치는 요소가 무엇인지 출력하세요!
(정규화 X)

데이터 : sport.csv

데이터 설명 : academic : 학점수

sports : 체육점수

music : 음악점수

acceptance : 입학기준점수 (종속변수)

답 :

```
sport <- read.csv("sports.csv", header = T)
summary(sport) #키와 몸무게처럼 단위가 다름
reg(y=sport$acceptance, x=sport[2:4])
```

```
estimate
Intercept 11.4902799
academic 0.1557737
sports 0.5726859
music 0.1046008
```

설명 : 정규화를 안하고 영향도를 보았더니 sports가 가장 높게 나옴.

문제241. 이번에는 min/max 정규화를 하고 영향도를 확인하세요!

```
sports <- read.csv("sports.csv", header = T)
sports <- sports[ , c(2:5) ] # 학생번호 빼고 2번째 컬럼부터 5번째 컬럼까지

normalize <- function(x) {
  return ( ( x-min(x) ) / ( max(x) - min(x) ) )

}
sports_n <- as.data.frame(lapply(sports, normalize))
summary(sports_n)
reg(y= sports_n$acceptance, x=sports_n[, c(1:3)])
```

결과 :

```
estimate
Intercept 0.06121748
academic 0.48963854 #가장크다
sports 0.30194528
music 0.11432339
```

문제242. 나이가 많을수록 의료비가 많이 든다는것을 앞에서 상관관계로 확인했는데 나이 데이터를 더 크게 만 들어서 파생변수로 생성하면 결정계수가 더 올라가는지 확인해봅니다.

답 : insurance\$age2 <- insurance\$age^2

```
head(insurance)
```

기준 결정계수 : Multiple R-squared: 0.7509

```
model<- lm(expenses ~ ., data=insurance )  
summary( model3 )
```

조금 올라감

Multiple R-squared: 0.7537

문제243. 비만인 사람(bmi가 30이상)이 의료비가 더 많이 들거라 예상하고 insurance테이블에 비만인 사람과 비만이 아닌 사람에 대한 컬럼을 추가해서 결정계수가 더 올라가는지 확인해봅니다.

```
insurance$bmi30 <- ifelse( insurance$bmi >= 30, 1, 0)  
head(insurance)
```

```
model4 <- lm( expenses ~ ., data=insurance)  
summary(model4)
```

조금 더 올라감

Multiple R-squared: 0.7582

문제244. 비만인 사람이 담배까지 피게되면 의료비가 더 증가할 것으로 예측이 되므로 비만이면서 흡연자인 사람은 1로 하고 아니면 0으로 하는 파생변수를 생성하고 결정계수가 더 올라가는지 확인해 봅니다.

```
model5 <- lm( expenses ~ age + age2 + children + bmi + sex + bmi30*smoker + region, data=insurance )  
summary( model5 )
```

확 올라감

Multiple R-squared: 0.8664

```
bmi30:smokeryes 19810.1534 604.6769 32.762 < 2e-16 ***
```

위를 보면 흡연만 할 때보다 비만인 사람이 흡연할 때 연간 의료비가 더 드는데 원래 흡연만 할 때는 연간 의료비가 평균 13404 달러인데 비만인 사람이 흡연가지 하게되면 연간의료비가 19810달러가 지출됨을 확인할 수 있다.

문제245. (오늘의 마지막 문제 2/2) 현재 결정계수가 0.8664인데 이 결정계수를 더 올릴 수 있는 파생변수를 생각해서 모델을 생성하시오!

```
insurance$age29 <- ifelse(insurance$age >= 29, 1, 0) #나이 29세 이상  
insurance$children2 <- ifelse(insurance$children >= 2, 1, 0) #아이가 2명이상  
#head(insurance)
```

```
model <- lm( expenses ~ age + age2 + children + bmi + bmi30*smoker*sex*age29*children2*region,  
data=insurance )  
summary(model)
```

결과 :

```
Residual standard error: 4454 on 1209 degrees of freedom  
Multiple R-squared:  0.8777, Adjusted R-squared:  0.8647  
F-statistic: 67.78 on 128 and 1209 DF,  p-value: < 2.2e-16
```

문제246. test_vif2.csv 를 로드하면 등급평균이 추가되어있는데 이 데이터를 로드해서 다중회귀분석을 하고 다중공선성을 보이는 독립변수들이 있는지 확인하시오!

※ 중요하게 확인해야 할 내용 :

다중 공선성을 보이는 변수들의 p-value 값이 어떻게 나타나고 있는지 확인해야 한다!!!!!!
그 독립변수의 p-value가 0.05 이내를 나타내는지 확인해야 한다. 그래야 유의한 변수이기 때문

1. 데이터 로드

```
test2 <- read.csv( "test_vif2.csv" )  
View( test2 )
```

2. 상관관계 확인

```
test2 <- test[, -1 ]  
cor( test2 )  
cor(test2[ , c("아이큐", "공부시간", "등급평균")])
```

결과 :

```
아이큐  공부시간  등급평균  
아이큐  1.0000000 0.7710712 0.9736894  
공부시간 0.7710712 1.0000000 0.7300546  
등급평균 0.9736894 0.7300546 1.0000000
```

3. 회귀분석을 한다.

```
m2 <- lm(test2$시험점수 ~., data=test2)
```

Residuals:

Min	1Q	Median	3Q	Max
-6.3146	-1.2184	-0.4266	1.5516	5.6358

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	50.30669	35.70317	1.409	0.2085
아이큐	0.05875	0.55872	0.105	0.9197
공부시간	0.48876	0.17719	2.758	0.0329 *
등급평균	7.37578	8.63161	0.855	0.4256

Signif. codes:	0	'***'	0.001	'**'
			0.01	'*'
			0.05	'. '
			0.1	' '
			1	

설명 : 아이큐와 등급평균의 p-value가 크게 나왔다. 즉 각각의 독립변수들은 종속변수에 유의한 영향을 미치지 않고 있다. 그렇게 때문에 아이큐와 등급평균 둘다 시험점수에 미치는 의미가 없거나 두 변수들의 다중 공선성

을 의심해봐야한다.

Residual standard error: 3.952 on 6 degrees of freedom
Multiple R-squared: 0.9155, Adjusted R-squared: 0.8733
F-statistic: 21.68 on 3 and 6 DF, p-value: 0.001275

4. 다중공선성 여부를 확인한다.

```
vif(m2) > 5  
아이큐 공부시간 등급평균  
TRUE FALSE TRUE
```

```
vif(m2)  
아이큐 공부시간 등급평균  
22.643553 2.517786 19.658264
```

5. 이럴때는 아이큐와 등급평균이 둘 다 강한 상관관계를 보여서 회귀분석 결과에 좋지 않은 결과를 보였으므로 둘중에 하나를 제외하고 회귀 분석을 해야함.

둘 중에 어떤것을 제외시켜야할지는 각각 테스트 해보고 결정계수가 높은것을 선택하면 됨

a. 아이큐, 공부시간

b. 등급평균, 공부시간

```
model3 <- lm( test2$시험점수 ~., data=test2[, c(1,2,3)] )  
summary( model3 ) #결정계수 0.90
```

```
model4 <- lm( test2$시험점수 ~., data=test2[, c(1,3,4)] )  
summary( model4 ) #결정계수 0.91
```

결론 :

중요하다고 생각되는 독립변수의 회귀계수에 대한 검정결과가 유의하지 않게 나왔다면 (p-value가 크게 나왔다면) 다중 공선성을 의심해봐야 한다.

문제247. 미국 국민의 의료비 데이터로 의료비를 예측하는 회귀모델을 생성했는데 이 모델을 이용해서 의료비를 쉽게 예측할 수 있는 IBM의 웃슨과 같은 질문을 던지는 코드를 구현하시오!

나이가 어떻게 되십니까?

부양가족수가 몇명입니까?

흡연을 하십니까? (yes/no)

비만지수가 어떻게 되십니까? (16~53)

사는 지역이 어디십니까? (southwest/southeast/northwest/northeast)

결과 : AI 가 예측한 연간 의료비는 13270 달러 입니다.

```
reg_func <- function(){
```

```

x1=as.integer(readline('나이가 어떻게 되십니까? '))
x2=readline('성별이 어떻게 되십니까? 여자면 1, 남자면 0 ')
x3= readline('비만지수가 어떻게 되십니까? (16~53) ')
x4=readline('부양가족수가 몇명입니까?(0~5)')
smoke=readline('흡연을 하십니까?(yes/no)')
region=readline('사는 지역이 어디십니까? (southwest/southeast/northwest/northeast) ')

if ( smoke=='yes' ) { smoke_x <- 23847.5 }
else if ( smoke=='no' ) { smoke_x <- 1      }

region_x <- 0

if ( region =='northeast' ) { region_x<- 1  }
else if ( region =='northwest' ) { region_x <- -352.8 }
else if ( region =='southeast' ) { region_x <- -1035.6 }
else if ( region =='southwest ' ) { region_x <- -959.3 }

y <- 256.8*x1 - 131.4*x2 + 339.3*x3 + 475.7*x4 + smoke_x + region_x

print ( ' ai 가 예측한 연간 의료비는 ' , y , ' 입니다 ' )

}

reg_func()

```

문제248. 책 248페이지의 그림인 원본 데이터를 A속성으로 나누는게 더 나은지 B 속성으로 나누는게 더 나은지 SDR을 구해서 알아내시오. 어떤게 더 균일하게 잘 나눈건지 확인하시오!



1. 원본 데이터를 만든다.

```
tee <- c( 1, 1, 1, 2, 2, 3, 4, 5, 5, 6, 6, 7, 7, 7, 7 )
```

2. 원본 데이터를 A 속성으로 나누었을 때의 데이터

```
at1 <- c( 1, 1, 1, 2, 2, 3, 4, 5, 5 )
at2 <- c( 6, 6, 7, 7, 7, 7 )
```

3. 원본 데이터를 B 속성으로 나누었을 때의 데이터

```
bt1 <- c( 1, 1, 1, 2, 2, 3, 4 )
bt2 <- c( 5, 5, 6, 6, 7, 7, 7, 7 )
```

4. A 속성으로 나누었을 때의 SDR을 구한다.

```
sdr_a <- sd(tee) - ( length(at1) / length(tee) * sd(at1) + length(at2) / length(tee) * sd(at2) )
sdr_a # 1.202815
```

5. (점심시간 문제) B 속성으로 나누었을 때의 SDR을 구한다.

```
sdr_b <- sd(tee) - ( length(bt1) / length(tee) * sd(bt1) + length(bt2) / length(tee) * sd(bt2) )
sdr_b #1.392751
```

6. 둘 중에 SDR이 높은 것으로 분류한다.

```
b !!
```

7. b 속성으로 분류한 원본 데이터의 두 영역의 평균값을 각각 구해서 등급을 예측하자

```
bt1 <- c( 1, 1, 1, 2, 2, 3, 4 )
bt2 <- c( 5, 5, 6, 6, 7, 7, 7 )
```

mean(bt1) #2로 예측한다

mean(bt2) #6.25로 예측한다

문제249. 미국 보스톤 지역의 집값을 예측하는 머신러닝 모델을 생성하는데 회귀트리를 먼저 생성하고 상관관계와 오차를 확인하고 모델트리를 생성해서 상관관계와 오차가 더 향상이 되었는지 확인하는 테스트를 수행하시오!

(오늘의 마지막 문제 2/3) SQL로 회귀분석을 성공했으면 구현한 스크립트와 csv파일을 답글로 올려서 검사맡으시오.

--1. 보스톤 하우징 데이터를 테이블로 구성한다.

```
drop table BOSTON_HOUSING;
```

```
CREATE TABLE BOSTON_HOUSING
```

```
(
```

```
    ID      NUMBER,
    CRIM    NUMBER,
    ZN      NUMBER,
    INDUS   NUMBER,
    CHAS    NUMBER,
    NOX    NUMBER,
    RM     NUMBER,
    AGE    NUMBER,
    DIS     NUMBER,
    RAD    NUMBER,
    TAX    NUMBER,
    PTRATIO NUMBER,
    BLACK   NUMBER,
```

```
LSTAT    NUMBER,  
MEDV     NUMBER  
);
```

--2. 머신러닝 구현

```
DROP TABLE SETTINGS_GLM;
```

```
CREATE TABLE SETTINGS_GLM  
AS  
SELECT *  
FROM TABLE (DBMS_DATA_MINING.GET_DEFAULT_SETTINGS)  
WHERE SETTING_NAME LIKE '%GLM%';
```

```
BEGIN
```

```
INSERT INTO SETTINGS_GLM  
VALUES (DBMS_DATA_MINING.ALGO_NAME, 'ALGO_GENERALIZED_LINEAR_MODEL');  
-- 회귀분석 코드
```

```
INSERT INTO SETTINGS_GLM  
VALUES (DBMS_DATA_MINING.PREP_AUTO, 'ON');
```

```
INSERT INTO SETTINGS_GLM  
VALUES (  
DBMS_DATA_MINING.GLMS RIDGE_REGRESSION,  
'GLMS RIDGE_REG_DISABLE');
```

```
INSERT INTO SETTINGS_GLM  
VALUES (  
DBMS_DATA_MINING.ODMS MISSING_VALUE_TREATMENT,  
'ODMS MISSING_VALUE_MEAN_MODE');
```

```
--결측치를 그 데이터의 평균값으로 치환하라는 코드
```

```
COMMIT;
```

```
END;  
/
```

--3. 모델 생성

```
CREATE OR REPLACE VIEW VW_BOSTON_HOUSING
```

```

AS SELECT * FROM BOSTON_HOUSING;

BEGIN

DBMS_DATA_MINING.CREATE_MODEL(
    model_name      => 'MD_GLM_MODEL', --모델명
    mining_function => DBMS_DATA_MINING.REGRESSION, --회귀분석하겠다.
    data_table_name => 'VW_BOSTON_HOUSING', --회귀분석 할 테이블명
    case_id_column_name => 'ID', --보스톤 하우징의 집의 id 번호
    target_column_name => 'MEDV', --라벨컬럼
    settings_table_name => 'SETTINGS_GLM'); --환경셋팅 테이블 이름

END;
/

```

--4. 모델 확인

```

SELECT MODEL_NAME,
       ALGORITHM,
       COMMENTS,
       CREATION_DATE,
       MINING_FUNCTION,
       MODEL_SIZE
  FROM ALL_MINING_MODELS
 WHERE MODEL_NAME = 'MD_GLM_MODEL';

```

```

-- 만든 머신러닝 환경정보 확인
SELECT SETTING_NAME, SETTING_VALUE
  FROM ALL_MINING_MODEL_SETTINGS
 WHERE MODEL_NAME = 'MD_GLM_MODEL';

```

--R에서 회귀분석 결과를 본 내용이 아래의 SQL이다.

```

SELECT * FROM
TABLE (DBMS_DATA_MINING.GET_MODEL_DETAILS_GLM ('MD_GLM_MODEL'));

```

-- 5. 테스트 데이터를 담을 테이블을 생성하고 test.csv를 로드한다.

```
CREATE TABLE BOSTON_HOUSING_TEST
```

```
(  
    ID      NUMBER,  
    CRIM    NUMBER,  
    ZN      NUMBER,  
    INDUS   NUMBER,  
    CHAS    NUMBER,  
    NOX     NUMBER,  
    RM      NUMBER,  
    AGE     NUMBER,  
    DIS     NUMBER,  
    RAD     NUMBER,  
    TAX     NUMBER,  
    PTRATIO  NUMBER,  
    BLACK   NUMBER,  
    LSTAT   NUMBER,  
    MEDV   NUMBER  
);
```

```
SELECT * FROM BOSTON_HOUSING_TEST;
```

-- 6. 테스트 데이터의 집값을 위해서 만든 회귀 모델로 예측한다.

```
SELECT id,  
       PREDICTION (MD_GLM_MODEL USING *) MODEL_PREDICT_RESPONSE  
FROM BOSTON_HOUSING_TEST T;
```

문제250. (점심시간문제) AND 패셉트론의 최종 가중치가 어떻게되는지 손으로 풀어서 알아내시오!

신성이의 답 :

##가중치 w_i 와 변수설명 : 변수 x_1, x_2 가정

가중치 $w_i = w_i + r \cdot x_i \cdot (t - f(k))$

r : running rate(학습률) --> hyper parameter --> 이번실습에서 $r=0.01$ 로 가정

k : $\sum(w_i \cdot x_i)$ # $k = x_0 \cdot w_0 + x_1 \cdot w_1 + x_2 \cdot w_2$

t : label column

function $f()$: 활성함수 --> if $k > 0$, 1 ... else 0

$f(k)$: predict val.

#변수설정(x_0, w_0, w_1, w_2)

$x_0 = -1$ (입력신호 1~4 모두 동일, $x_0=-1$ 은 회귀분석의 절편값으로 이해할 수 있음)

w0=0.3, w1=0.4, w2=0.1 (w0, w1, w2 는 임의지정)

loop1	x1	x2	k	f(k)	t=label	err = t-f(k)	wi = wi + r · xi · (t-f(k))	i=0	i=1	i=2
입력신호1	0	0	-0.3	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호2	0	1	-0.2	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호3	1	0	0.1	1	0	-1	wi = wi + 0.01 · xi · -1	w0=0.31	w1=0.39	w2=w2
입력신호4	1	1	0.18	1	1	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2

#7번째 컬럼은 가중치 wi를 구하는 식

#8~10번째 컬럼은 i=0,1,2 일때의 가중치값이다.

#w0=w0 는 값이 변하지 않은경우이며 w0=0.31은 기존의 w0=0.3에서 값이 갱신된 경우이다.

#wi 값이갱신되면갱신된값을넣어서계산해주어야한다. 위표에서입력신호 3에서 w0, w1 값이변경되었으므로
변경된 w0=0.31, w1=0.39, w2=w2=0.1 을 대입하여 입력신호 4를 처리해주어야 한다.

#입력신호 4가 끝났으면 loop1을 끝내고 loop2에서 다시 입력신호1부터 위 과정을 반복한다.

k : $\sum(wi \cdot xi)$ #k = x0·w0 + x1·w1 + x2·w2

loop1에서 설정된 변수값 : x0=-1 // w0=0.31, w1=0.39, w2=w2=0.1 //

loop2	x1	x2	k	f(k)	t=label	err = t-f(k)	wi = wi + r · xi · (t-f(k))	i=0	i=1	i=2
입력신호1	0	0	-0.31	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호2	0	1	-0.21	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호3	1	0	0.08	1	0	-1	wi = wi + 0.01 · xi · -1	w0=0.32	w1=0.38	w2=w2
입력신호4	1	1	0.16	1	1	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2

k : $\sum(wi \cdot xi)$ #k = x0·w0 + x1·w1 + x2·w2

loop2에서 설정된 변수값 : x0=-1 // w0=0.32, w1=0.38, w2=w2=0.1 //

loop3	x1	x2	k	f(k)	t=label	err = t-f(k)	wi = wi + r · xi · (t-f(k))	i=0	i=1	i=2
입력신호1	0	0	-0.32	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호2	0	1	-0.22	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호3	1	0	0.06	1	0	-1	wi = wi + 0.01 · xi · -1	w0=0.33	w1=0.37	w2=w2
입력신호4	1	1	0.14	1	1	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2

이쯤에서 위 loop1, loop2, loop3 에서 k값이 일정한 크기로 작아지고 있는것을 관찰할 수 있다.

규칙대로라면 loop 6에서 입력신호 3번의 k값이 0.00 이 되고 f(k)=0 으로 err 값이 0이 되어 wi 값이 고정될 것으로 기대할 수 있다. 실제로 그럴지 loop6 까지 수작업으로 진행해 보자.

loop3에서 설정된 변수값 : x0=-1 // w0=0.33, w1=0.37, w2=0.1 //

loop4	x1	x2	k	f(k)	t=label	err = t-f(k)	wi = wi + r · xi · (t-f(k))	i=0	i=1	i=2
입력신호1	0	0	-0.33	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호2	0	1	-0.23	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호3	1	0	0.04	1	0	-1	wi = wi + 0.01 · xi · -1	w0=0.34	w1=0.36	w2=w2
입력신호4	1	1	0.12	1	1	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2

loop4에서 설정된 변수값 : $x_0=-1$ // $w_0=0.34$, $w_1=0.36$, $w_2=0.1$ //

loop5	x1	x2	k	f(k)	t=label	err = t-f(k)	wi = wi + r · xi · (t-f(k))	i=0	i=1	i=2
입력신호1	0	0	-0.34	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호2	0	1	-0.24	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호3	1	0	0.02	1	0	-1	wi = wi + 0.01 · xi · -1	w0=0.35	w1=0.35	w2=w2
입력신호4	1	1	0.1	1	1	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2

loop5에서 설정된 변수값 : $x_0=-1$ // $w_0=0.35$, $w_1=0.35$, $w_2=0.1$ //

loop6	x1	x2	k	f(k)	t=label	err = t-f(k)	wi = wi + r · xi · (t-f(k))	i=0	i=1	i=2
입력신호1	0	0	-0.35	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호2	0	1	-0.25	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호3	1	0	0.00	0	0	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2
입력신호4	1	1	0.1	1	1	0	wi = wi + 0.01 · xi · 0	w0=w0	w1=w1	w2=w2

##결론

중간에 예상한대로 loop6에서 모든 입력신호(1~4)에 대해 w_i 값이 (0~2)일때 변화가 없었다.

이 경우 가중치 w_i 를 모델에 잘 학습된 가중치라고 생각할 수 있다.

위 loop에 따라 산출된 최종 가중치는 아래와 같다.

$w_0=0.35$

$w_1=0.35$

$w_2=0.1$

$r = 0.01$

\

문제251. 아래의 행렬을 만들고 inputs1이라는 변수에 아래의 행렬을 입력하세요!

```
0  0
1  0
0  1
1  1
```

답 :

```
inputs1 <- matrix( c( 0, 0, 1, 0, 0, 1, 1, 1), nrow=4, ncol=2, byrow=T )
inputs1
```

문제252. targets1이라는 변수에 아래의 행렬을 입력하시오!

```
0
0
0
1
```

답 :

```
targets1 <- matrix( c(0, 0, 0, 1), nrow=4)
```

문제253. w0, w1, w2에 해당하는 가중치 행렬을 3행 1열로 만들어서 w라는 변수에 저장하시오!

```
w <- matrix( c( 0.3, 0.4, 0.1), nrow=3)
```

문제254. -1의 값을 4행 1열로 하는 행렬을 만드시오. 만들고 x0이라는 변수에 넣으시오!

```
x0 <- matrix( c (-1, -1, -1, -1), nrow=4 )
```

```
-1  
-1  
-1  
-1
```

문제255. x0행렬과 inputs1 행렬을 cbind로 묶어서 아래의 행렬을 new_input에 넣으시오.

```
-1 0 0  
-1 1 0  
-1 0 1  
-1 1 1
```

답:

```
new_input <- cbind(x0, inputs1)  
new_input
```

문제256. new_input행렬과 w 행렬과의 곱을 구하시오!

	w
-1 0 0	0.3
-1 1 0	0.4
-1 0 1	0.1
-1 1 1	

```
k <- new_input%*% w  
k
```

결과 :

```
[,1]  
[1,] -0.3  
[2,] 0.1  
[3,] -0.2  
[4,] 0.2
```

문제257. 아래의 계단함수(활성화 함수)를 생성하시오! 함수이름 : step

```
step <- function(x){ ifelse( x>0, 1, 0 ) }  
step(0.3) #1  
step(-0.2) #0
```

문제258. 위에서 생성한 k 값을 step함수에 넣고 targets1과의 차이를 구하시오!

```
tkf <- targets1 - step(k)
tkf
```

문제259. for loop문을 이용해서 1부터 4까지 숫자가 출력되게하시오!

```
답 : for (i in 1:4){
    cat("₩n", i )
}
```

문제260. 위에서 만든 오차(tkf변수)를 이용해서 가중치(w변수)를 갱신하는 아래의 식을 구현하시오!

```
wi <- wi + 0.01*x1*tkf
```

```
for (i in 1:4) {
    for (j in 1:3) {
        w[j] <- w[j] + 0.01* new_input[i, j] * tkf[i]
    }
    cat( '₩n row:', i, w )
}
```

문제261. 활성화 함수인 계단함수를 R로 생성하고 계단 그래프를 그리시오.

```
step <- function( x ) { ifelse( x>0, 1, 0 ) }
step( -0.1 )
step( 1.4 )

x <- seq( -5, 5, 0.01 )
x
plot( x, step(x), col='blue', type='o')
```

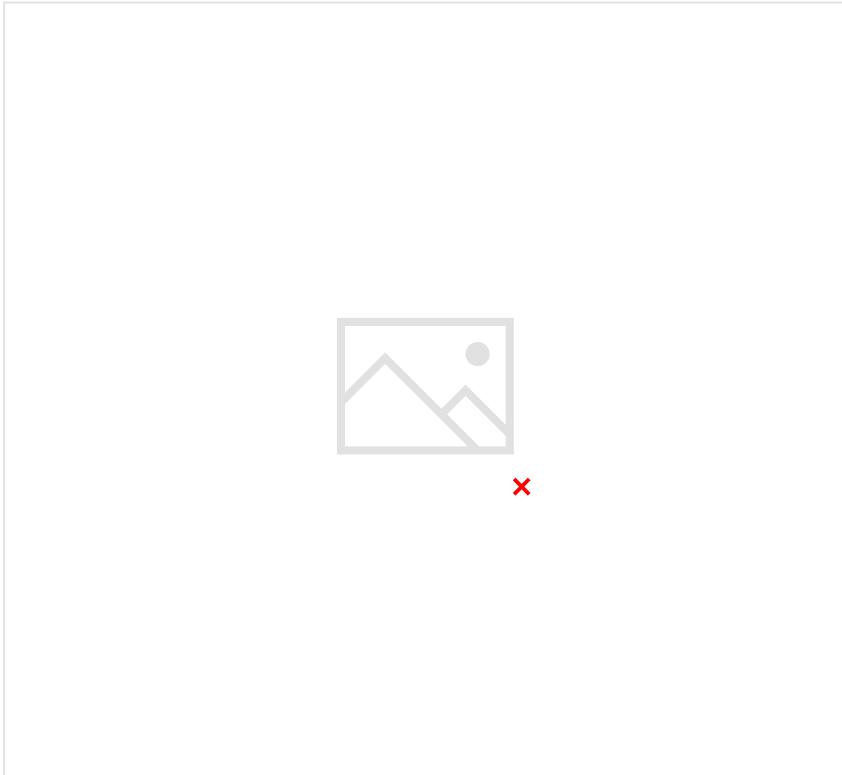


문제262. R로 시그모이드 함수를 생성하시오!

```
sigmoid <- function(x) { 1 / ( 1 + exp( -x ) ) }  
sigmoid(3)
```

문제263. 계단함수처럼 시그모이드 함수도 그래프로 그리시오!

```
sigmoid <- function(x) { 1 / ( 1 + exp( -x ) ) }  
x <- seq( -5, 5, 0.01 )  
plot(x, sigmoid(x), col='blue')
```



문제264. 렐루 함수를 생성하시오!

렐루함수 : 입력되는 값이 0이거나 음수이면 0을 출력하고 0보다 크면 입력되는 값을 그대로 출력하는 함수

```
relu <- function(x){ ifelse(x<=0,0,x)}  
relu(4)  
relu(-3)
```

```
relu(4) #4  
relu(-3) #0
```

문제265. 렐루 함수의 그래프를 그리시오!

```
plot(x, relu(x), col='blue')
```



문제266. 지금보다 더 성능이 좋은 콘크리트 강도를 예측하는 신경망을 생성하시오!

```
concrete_model3 <- neuralnet(formula=strength ~ cement + slag + ash +  
+ water +superplastic + coarseagg + fineagg + age, data =concrete_train,  
hidden=c(10,6,3) )  
model_results2 <- compute(concrete_model3, concrete_test[1:8])  
predicted_strength3 <- model_results2$net.result  
cor(predicted_strength3, concrete_test$strength)
```

문제267. (오늘의 마지막문제 2/4) 보스톤 지역 집값을 예측하는 신경망을 만들고 케글에 올려서 자신의 스코어 (오차)를 캡쳐해서 결과를 올리시오!

```
getwd()
```

1. 데이터를 수집한다.

```
boston_train<- read.csv('train.csv', head=T)  
boston_test<-read.csv("test.csv", head=T)
```

```
str(boston_train)
```

데이터 설명 :

- [01] CRIM 자치시(town) 별 1인당 범죄율
- [02] ZN 25,000 평방피트를 초과하는 거주지역의 비율
- [03] INDUS 비소매상업지역이 점유하고 있는 토지의 비율
- [04] CHAS 찰스강에 대한 더미변수(강의 경계에 위치한 경우는 1, 아니면 0)
- [05] NOX 10ppm 당 농축 일산화질소

- [06] RM 주택 1가구당 평균 방의 개수
- [07] AGE 1940년 이전에 건축된 소유주택의 비율
- [08] DIS 5개보스턴 직업센터까지의 접근성 지수
- [09] RAD 방사형 도로까지의 접근성 지수
- [10] TAX 10,000 달러 당 재산세율
- [11] PTRATIO 자치시(town)별 학생/교사 비율
- [12] B $1000(Bk-0.63)^2$, 여기서 Bk는 자치시별 흑인의 비율을 말함.
- [13] LSTAT 모집단의 하위계층의 비율(%)
- [14] MEDV 본인 소유의 주택가격(중앙값) (단위: \$1,000)

2. 정규화 작업을 진행한다.

정규화를 왜 하는가? 컬럼(변수)들의 단위가 다 다르므로 맞춰주기 위해 정규화 함 (예: 키와 몸무게)

1. min/max 함수 : 0 ~ 1 사이의 데이터로 재구성하는 함수
2. scale 함수 : 평균이 0이고 표준편차가 1인 데이터로 재구성하는 함수

```
normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x) ) )
}
```

```
head(boston_train)
head(boston_test)

boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

summary(boston_train_norm)
summary(boston_test_norm)
```

설명 : id 컬럼은 제외하고 나머지 컬럼들만 정규화 함

3. 신경망 모델을 생성한다.

```
#install.packages("neuralnet")
library(neuralnet)

boston_model <- neuralnet(formula=medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad + tax
+ ptratio + black + lstat ,
  data=boston_train_norm ,hidden = c(2,5) , learningrate =0.02)
```

입력층 ---> 은닉1층 ---> 은닉2층 ---> 출력층(3층) : 3층신경망

입력층 : 독립변수의 개수들만큼 뉴런의 개수

은닉1층 : 뉴런의 개수 2개

은닉2층 : 뉴런의 개수 5개

출력층 : 뉴런의 개수 1개 (종속변수)

※ learning rate (학습률)

우리가 신경망을 통해서 알아내야하는 파라미터는 ? 가중치 (weight)

4. 모델을 평가한다.

```
model_results <- compute(boston_model, boston_test_norm)
```

#설명 : 컴퓨터가 예측한 집값도 0~1사이의 정규화된 결과로 출력한다.

```
predicted_medv <- model_results$net.result  
predicted_medv
```

※ 중요 : 캐글에 테스트 데이터에 대한 예측한 집값을 올릴때는 정규화 된 것으로 올리면 안되고 실제 집값으로 올려줘야한다.

```
boston_test_norm2 <- data.frame(boston_test_norm, "MEDV"=predicted_medv)
```

```
minvec <- sapply(boston_test,min)  
maxvec <- sapply(boston_test,max)
```

5. 다시 데이터를 역정규화 시켜준다.

```
denormalize <- function(x) { return ( x*45+5 ) }  
denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data$medv)  
pred_medv_un <- denormalize(pred_medv)  
  
sample <- cbind(test_data_id, pred_medv_un)  
colnames(sample) <- c("id", "medv")  
View(sample)  
write.csv(sample, "Submission_sample.csv", row.names=FALSE)
```

#5.1 정규화된 테스트 데이터에 예측한 집값을 붙여서 데이터 프레임으로 구성함

```
boston_test_norm2 <- data.frame(boston_test_norm, "MEDV"=predicted_medv)  
head(boston_test_norm2)
```

#5.2 역정규화 작업을 수행합니다.

```
minvec <- sapply(boston_test,min)  
maxvec <- sapply(boston_test,max)
```

```
denormalize <- function(x,minval,maxval) {  
  x*(maxval-minval) + minval  
}
```

```
boston_test2 <- as.data.frame(Map(denormalize,boston_test_norm2, minvec,maxvec))
```

6. 캐글에 제출한 형식으로 csv 파일을 생성하기

```
result <- cbind(boston_test$ID, boston_test2$MEDV)
```

```
result  
  
colnames(result) <- c('ID','MEDV')  
result  
  
result2<-data.table(result)  
  
result2
```

```
write.csv(result2, file="boston_test9.csv",row.names = F)  
# rowname을 붙이면 안 올라가기 때문에 rowname을 없애야함
```

7. 캐글에 결과를 올려본다.

<https://www.kaggle.com/c/boston-housing/submissions>

Q : 하이퍼 파라미터를 조정해서 오차가 더 떨어지게 하시오!

- 하든레이어 변경 ex : hidden= c(6,4,4)
- 책 p 399 소프트 활성화 함수로 신경망의 활성화 함수를 변경



```
getwd()
```

```
normalize <- function(x) {  
  return ( (x-min(x)) / (max(x) - min(x) ) )  
}  
  
boston_train<- read.csv('train.csv', head=T)  
boston_test<-read.csv("test.csv", head=T)  
str(boston_test)
```

```

head(boston_train)
head(boston_test)

boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

summary(boston_train_norm)
summary(boston_test_norm)

#install.packages("neuralnet")
library(neuralnet)
softplus <- function(x){log(1+exp(x))}
boston_model <- neuralnet(formula=medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad + tax
+ ptratio + black + lstat ,
                           data=boston_train_norm ,hidden = c(6,4,2) , act.fct = softplus)

model_results <- compute(boston_model, boston_test_norm)

pred_medv <- model_results$net.result
pred_medv

denormalize <- function(x) { return ( x*(50-5)+5 ) } #max(train_data$medv) #min(train_data$medv)
pred_medv_un <- denormalize(pred_medv)

sample <- cbind(boston_test$ID, pred_medv_un)
head(sample)

colnames(sample) <- c("id", "medv")

write.csv(sample, "Submission_sample.csv", row.names=FALSE)

```

문제268. (점심시간 문제) 어제 마지막 문제를 오전에 다 설명해 줬으니 전체 코드와 스코아를 올리시오!



```

getwd()

normalize <- function(x) {
  return ( (x-min(x)) / (max(x) - min(x) ) )
}

boston_train<- read.csv('train.csv', head=T)
boston_test<-read.csv("test.csv", head=T)
str(boston_test)

head(boston_train)
head(boston_test)

boston_train_norm <- as.data.frame(lapply(boston_train[,-1], normalize))
boston_test_norm <- as.data.frame(lapply(boston_test[,-1],normalize) )

summary(boston_train_norm)
summary(boston_test_norm)

library(neuralnet)
softplus <- function(x){log(1+exp(x))}
boston_model <- neuralnet(formula=medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad + tax
+ ptratio + black + lstat ,
                           data=boston_train_norm ,hidden = c(6,4,2) , act.fct = softplus)

model_results <- compute(boston_model, boston_test_norm)

pred_medv <- model_results$net.result
pred_medv

denormalize <- function(x) { return ( x*(50-5)+5 ) } #max(train_data$medv) #min(train_data$medv)
pred_medv_un <- denormalize(pred_medv)

sample <- cbind(boston_test$ID, pred_medv_un)
head(sample)

colnames(sample) <- c("id", "medv")

write.csv(sample, "Submission_sample.csv", row.names=FALSE)

```

문제269. 보스턴 하우징 데이터를 다중회귀 분석하는데 moker*bmi30처럼 더미변수가 생성되게 하고 결정계수를 높여보시오!

문제270. (오늘의 마지막문제 2/5) 다중 회귀분석으로 보스턴 하우징 데이터의 집값 예측 모델을 만들어서 테스트 데이터를 예측하고 캐글에 점수를 확인하시오!

문제271. 다음 데이터의 지지도를 구하시오!

거래번호	거래 아이템	지지도(우유 → 시리얼) ?
1	우유, 버터, 시리얼	
2	우유, 시리얼	우유와 시리얼을 동시에 구매할 지지도?
3	우유, 빵	(결합확률)
4	버터, 맥주, 오징어	

$$\frac{n(\text{ 우유} \cap \text{ 시리얼})}{\text{전체건수}} = \frac{2}{4} = 50\%$$

문제272. 우유를 구매할 때 시리얼도 구매할 신뢰도를 구하시오! (조건부 확률)

거래번호	거래 아이템	신뢰도(우유 → 시리얼) ?
1	우유, 버터, 시리얼	
2	우유, 시리얼	우유를 샀을 때 시리얼도 구매할 확률?
3	우유, 빵	(조건부 확률)
4	버터, 맥주, 오징어	

$$\text{표기식} : \text{confidence}(X \rightarrow Y) = \frac{n(X \cap Y)}{n(X)} = \frac{2}{3} = \text{약 } 66\%$$

문제273. 우유를 구매할 때 시리얼도 구매할 향상도를 구하시오!

거래번호	거래 아이템	향상도(우유 → 시리얼) ?
1	우유, 버터, 시리얼	
2	우유, 시리얼	우유를 샀을 때 시리얼을 산다는게 우연인지 아닌지를 알아보는 확률
3	우유, 빵	
4	버터, 맥주, 오징어	

$$\text{향상식} : \text{lift}(A \rightarrow B) = \frac{P(B | A)}{P(B)} = \frac{\text{confidence}(A \rightarrow B)}{\text{support}(Y)} = \frac{(2/3)}{(2/4)} = \frac{4}{3} = 1.33$$

문제274. 우유와 시리얼을 샀을 때의 지지도와 신뢰도를 각각 구하시오!

	지지도	신뢰도
우유 ----> 시리얼 :	50%	66%
시리얼 ----> 우유 :	50%	100%

거래번호	거래 아이템

- 1 우유, 버터, 시리얼
- 2 우유, 시리얼
- 3 우유, 빵
- 4 버터, 맥주, 오징어

문제275. (빅데이터 기사시험문제 - 연관규칙) 다음은 남학생과 여학생이 선호하는 책에 대한 교차표이다. 전체에서 1명을 뽑았을 때 그 학생이 남학생인 경우 소설책을 좋아할 확률은 무엇인가?

	소설책	여행책	
남학생	50	30	80
여학생	10	20	30
	60	50	



답 : 5/8

문제276. (빅데이터 기사시험 - 연관성 분석) 다음은 쇼핑몰의 거래내역이다.

연관규칙 : 우유 -----> 빵에 대한 신뢰도는 얼마인가?

항목	거래수
우유	10
빵	20
우유, 빵	50
빵, 초콜릿	40
전체 거래건수	100



답 : 0.83

문제277. (빅데이터 기사 시험 - 연관규칙) 다음중 연관규칙 분석에서 품목A의 거리수가 5이고 품목 B의 거래수가 3, A와 B가 동시에 포함된 거래수가 4이다.

전체 거래수가 10일 때 품목 A,B의 지지도를 구하시오!



문제278. (빅데이터 기사시험 연관규칙) 아래는 A 쇼핑몰의 거래내역이다. 연관규칙 '과자 ---> 기저귀 '에 대한 지지도(support)는 얼마인가?

품목	판매수량
과자	10
기저귀	20
과자, 기저귀	40
기저귀, 맥주	20

전체 거래건수 100



×

문제279. (빅데이터 기사시험 응용문제) (점심시간 문제) 아래는 A 쇼핑몰의 거래내역이다.
연관규칙 '과자 ---> 기저귀'에 대한 신뢰도와 향상도 구하시오~

품목	판매수량
과자	10
기저귀	20
과자, 기저귀	40
기저귀, 맥주	20
전체 거래건수	100

빅데이터 기사 시험책 p.3- 55

답 :



문제280. (오늘의 마지막 문제 2/8) 다음은 쇼핑몰의 거래내역입니다. 연관규칙 '커피 -> 빵'에 대한 지지도는 얼마입니까?

항목	거래수
커피	10
빵	20
커피, 빵 동시 구매	50
전체 거래수	100

$$n(\text{ 커피 } \cap \text{ 빵 }) = 50$$

표기식 : $\text{support}(\text{ 커피 }, \text{ 빵 }) = \frac{n}{N} = \frac{50}{100} = 50\%$

문제281. (빅데이터 기사 시험 문제) 수제비 책 p. 3-134

다음 중 아래는 학생들의 키와 몸무게를 정규화 한 데이터이다. 유클리디안 거리를 사용해서 최단 연결법을 통해 학생들을 3개의 군집으로 나누고자 할 때 가장 적정한 것은?

- 사람 (키, 몸무게)
- A (1, 5)
 - B (2, 4)
 - C (4, 6)
 - D (4, 3)
 - E (5, 3)

1. (A, C), (B), (D, E)
2. (A, D), (B), (C, E)
3. (A, E), (C), (B, D)
4. (A, B), (C), (D, E)

가장 가까운 거리 찾기 (정답 : 4번)

문제282. (점심시간 문제) 아래의 데이터 세트 A와 B간의 맨하튼(Manhattan)거리로 계산하면 얼마인가 ?

김xx	서xx
키	165 180
몸무게	50 65



$$15 + 15 = 30$$

문제283. factoextra 패키지를 이용해서 동물 데이터를 k-means로 군집화하고 시각화 하시오!

```

zoo <- read.csv("zoo.csv")
ncol(zoo) #18, 18개의 컬럼
zoo_n <- zoo[ , 2:17] #동물 이름과 라벨을 제외한 컬럼으로만 구성

ncol(zoo_n)

zoo_model <- kmeans(zoo_n, 7) # k값을 7로 주고 군집화하는 모형을 생성

x <- cbind( zoo[ , 18], zoo_model$cluster)

x2 <- data.frame(x)

x2

library(dplyr)

orderBy(~X1, x2)

```

문제284. 유방암 데이터(wisc_bc_data.csv)의 악성종양과 양성종양을 k-means로 2개로 군집화해서 정답과 비교해 보시오~

```

wisc <- read.csv("wisc_bc_data.csv", header = T)
head(wisc)
ncol(wisc)

```

```

plot(wisc)

wisc2 <- wisc[, 3:32]

km <- kmeans(wisc2, 2)
km
cbind(wisc$diagnosis, km$cluster)

fviz_cluster( km, data = wisc2, stand=F)

```

준혁이가 만든 정답과 비교하는 코드:

```

wisc <- read.csv("wisc_bc_data.csv")
wisc$diagnosis <- factor(wisc$diagnosis,
                           level=c("B","M"),
                           labels = c(1,2))
#View(wisc)

wisc_km <- wisc[,c(-1,-2)]
#View(wisc_km)

km <- kmeans(wisc_km, 2)

cbind(wisc$diagnosis, km$cluster)
fviz_cluster(km, data = wisc_km)

library(gmodels)
CrossTable(wisc$diagnosis, km$cluster)

```

문제285. 국영수 데이터를 참고하여 수학은 아주 잘하는데 영어가 보통인 학생들에게 연락하기 위해서 그 학생들의 학생번호만 추출하시오!

첨부파일 academy.csv

1. 데이터를 로드한다.

```
academy <- read.csv("academy.csv")
```

수학점수와 영어점수만 선택

```
academy <- academy[ , c(3,4) ]
```

2. k 값을 4로 주고 비지도학습 시켜 모델을 생성한다.

```
km <- kmeans( academy, 4)
km
```

3. 시각화를 한다.

```
library(factoextra)
fviz_cluster(km , data=academy, stand=F)
```

4. 학생번호, 수학점수, 영어점수, 분류번호가 같이 출력되게하시오 !

```
academy <- read.csv("academy.csv")
cbind( academy[,c(1,3,4)], km$cluster)
```

```
academy_seg[km$cluster==1, 1]
```

```
academy_seg[km$cluster==1, "학생 번호"]
```

설명 : 마케팅을 위해서 k-means를 이용해서 세그멘테이션을 해서 관련 학생들에게 광고를 따로 진행

문제286. 부서번호, 부서번호별 평균월급을 출력하시오!

```
aggregate(sal~deptno, emp, mean)
```

결과 :

	deptno	sal
1	10	2916.667
2	20	2175.000
3	30	1566.667

문제287. 사원 테이블에서 자기가 속하는 부서번호의 평균월급이 출력되게 하시오!

```
ave(emp$sal)
```

결과 :

```
[1] 2073.214 2073.214 2073.214 2073.214 2073.214 2073.214 2073.214 2073.214
[10] 2073.214 2073.214 2073.214 2073.214 2073.214
```

```
ave( emp$sal, emp$deptno)
```

문제288. 이름, 월급, 부서번호, 자기가 속한 부서번호의 평균월급을 출력하시오!

```
SQL> select ename, sal, deptno,
      avg(sal) over ( partition by deptno ) as avgSal
     from emp;
```

```
R > emp$avgSal <- ave( emp$sal, emp$deptno, FUN=function(x)mean(x) )
emp [ , c("ename", "sal", "deptno", "avgSal")]
```

문제289. 사원 테이블에 결측치가 얼마나 있는지 확인하시오!

```
colSums( is.na( emp ) )
```

문제290. 사원 테이블의 커미션의 결측치를 자기가 속한 부서번호의 평균월급으로 치환하시오!

```
emp$comm[is.na( emp$comm )] <- ave( emp$sal, emp$deptno, FUN=function(x)mean(x))
emp
```

설명 : FUN=function(x)max(x) 로 하면 최대값으로 출력됩니다.

문제291. 남녀 성별의 비율이 어떻게 되는지 출력하시오!

```
prop.table(table(teens$gender))
```

결과:

F	M
0.8085496	0.1914504

설명 : 여성의 비율이 80%를 보이고 있다.

문제292. teens 데이터 프레임에 결측치가 얼마나 있는지 조회하시오!

```
답 : colSums( is.na( teens ) )
```

문제293. 졸업년도, 졸업년도(gradyear)별 평균나이를 출력하시오!

```
aggregate( age ~ gradyear, data=teens, mean, na.rm=TRUE )
```

문제294. teens 데이터 프레임에 결측치 있는지 확인하시오!

```
colSums(is.na(teens))
```

문제295. (오늘의 마지막 문제 2/9) 저자의 해석을 알아내기 위해 아래는 각 군집별로 female(여성)이 몇명 인지 출력한 결과이다. 아래의 결과를 출력하세요.

결과:

1	2	3	4	5
833	435	3605	1883	15298

힌트 : attach(emp)

```
tapply( sal, job, length )
```

답 :

```
tapply(female, cluster, sum)
```

문제296. [빅데이터 기사시험] [점심시간 문제] 다음 중 SOM에 대한 설명으로 가장 알맞지 않은 것은?

1. SOM은 경쟁학습으로 각각의 뉴런이 입력 벡터와 얼마나 가까운가를 계산하여 연결강도를 반복적으로 재조정하는 학습과정을 거치면서 연결 강도는 입력 패턴과 가장 유사한 경쟁층 뉴런이 승자가 된다.

2. SOM은 고차원 데이터를 저차원의 지도 형태로 형성하기 때문에 시각적으로 이해하기 쉬울 뿐 아니라 변수의 위치 관계를 그대로 보존하기 때문에 실제 데이터가 유사하면 지도상 가깝게 표현된다.

3. SOM은 입력변수의 위치 관계를 그대로 보전하여 입력변수의 정보와 그들의 관계가 지도상에 그대로 나타난다.

4. SOM을 이용한 군집분석은 역전파 알고리즘을 사용함으로써 군집의 성능이 우수하고 수행속도가 빠르다.

문제297. 책 446 페이지의 이원 교차표의 카파 통계량을 계산하시오!

(답 : 카페 게시글 794번 참조)

$$pr(a) = (1203 + 152) / 1390 = 0.9748$$

$$\begin{aligned} \text{actual ham} &: 1234 / 1390 = 0.8878 \\ &1207 / 1390 = 0.8683 \end{aligned}$$

$$\begin{aligned} \text{spam} &: 183 / 1390 = 0.1317 \\ &156 / 1390 = 0.1122 \end{aligned}$$

$$\begin{aligned} pr(e) &: 0.8878 \times 0.8683 = 0.7709 \\ &0.1317 \times 0.1122 = 0.0148 \\ &0.7709 + 0.0148 = 0.7857 \end{aligned}$$

$$\begin{aligned} 0.9748 - 0.7857 &= 0.1891 \\ 1 - 0.7857 &= 0.2143 \end{aligned}$$

$$0.1891 / 0.2143 = 0.8824$$

$$k = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)} = \frac{0.9748 - 0.7857}{1 - 0.7857} = \frac{0.1891}{0.2143} = 0.8824$$

문제298. 문제 297번의 카파통계량을 R로 구하시오!

```
sms_result <- read.csv( "sms_results.csv" )
head( sms_result )

install.packages( "vcd" ) #kappa를 계산하기 위한 패키지
library(vcd)

table(sms_result$actual_type, sms_result$predict_type)
```

결과 :

		ham	spam
ham	1203	4	
spam	31	152	

```
Kappa( table(sms_result$actual_type, sms_result$predict_type) )
```

	value	ASE	z	Pr(> z)
Unweighted	0.8825	0.01949	45.27	0
Weighted	0.8825	0.01949	45.27	0

설명 : 책 453페이지 맨 밑에 나온것처럼 kappa의 k를 소문자로 사용하지 않도록 조심해야 한다.

문제299. [빅데이터 기사 시험 응용] p4-36 아래의 혼동행렬에서 카파통계량은 얼마인가?

실제값 / 예측치	True	False	합계
True	30	70	100
False	60	40	100
합계	90	110	200

$$\Pr(a) = (30 + 40) / 200 = 0.35$$

$$\begin{aligned} \text{True} : 100/200 &= 0.5 \\ 90/200 &= 0.45 \end{aligned}$$

$$\begin{aligned} \text{False} : 100/200 &= 0.5 \\ 110/200 &= 0.55 \end{aligned}$$

$$\Pr(e) = (0.5 \times 0.45) + (0.5 \times 0.55) = 0.5$$

$$k = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)} = \frac{0.35 - 0.5}{1 - 0.5} = \frac{-0.15}{0.5} = -0.3$$

```
a <- as.table(matrix(c(30,70,60,40), byrow=T, nrow=2, ncol=2))
Kappa(a)
```

결과 :

	value	ASE	z	Pr(> z)
Unweighted	-0.3	0.06712	-4.47	7.826e-06
Weighted	-0.3	0.06712	-4.47	7.826e-06

문제300. 스팸분류기의 이원교차표를 보고 민감도와 특이도를 R로 구하시오!

TN(1203) FP(4)
FN(31) TP(152)

```
sms_result <- read.csv("sms_results.csv", stringsAsFactors = T)
```

```
install.packages("caret")
```

```
library(caret)
sensitivity( sms_result$predict_type, sms_result$actual_type, positive='spam')
```

결과 :
[1] 0.8306011

```
specificity( sms_result$predict_type, sms_result$actual_type, negative='ham' )
```

결과 :
[1] 0.996686

문제301. (빅데이터 기사시험) 아래의 혼동행렬에서 특이도는 얼마인가? (수제비 책 p4-36)

실제값 / 예측치	True	False	합계
True	30	70	100
False	60	40	100
합계	90	110	200

$$\text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} = \frac{40}{40 + 60} = 0.4$$

문제302. 아래의 스팸 분류기의 정밀도를 구하시오!

precision = TP / (TP + FP)

**TN(1203) FP(4)
FN(31) TP(152)**

답 :

```
sms_result <- read.csv("sms_results.csv", stringsAsFactors = T)
library(caret)
posPredValue( sms_result$predict_type, sms_result$actual_type, positive='spam' )
```

결과 :
[1] 0.974359

문제303. (오늘의 마지막문제 2/10 / 빅데이터 기사시험)

아래의 혼동행렬에서 민감도는 얼마인가? (수제비 책 p4-36)

실제값 / 예측치	True	False	합계
True	30	70	100
False	60	40	100
합계	90	110	200

민감도 : $\text{TP} / (\text{TP} + \text{FN}) = 30 / (70 + 30) = 0.3$

		예측			
		NO	YES		
실제	NO	TN	FP	특이도	$\frac{TN}{TN + FP}$
	YES	FN	TP	민감도	$\frac{TP}{FN + TP}$
		정밀도			
		$\frac{TP}{TP + FP}$			

문제304. 다시 위의 머신러닝 모델의 성능을 올리는 테스트를 진행하시오! 하이퍼 파라미터 seed값과 trials를 조정해서 정확도와 정밀도 외의 다른 성능척도가 어떻게 변하는지 테스트하고 엑셀에 기입하시오!

#1. 데이터를 로드한다.

```
credit <- read.csv("credit.csv", stringsAsFactor=TRUE)
str(credit)
```

#2. 데이터에 각 컬럼들을 이해한다.

```
#라벨 컬럼 : default ---> yes : 대출금 상환 안함
#no : 대출금 상환
```

```
prop.table( table(credit$default) )
summary( credit$amount)
```

#3. 데이터가 명목형 데이터인지 확인해본다.

```
str(credit)
```

#4. 데이터를 shuffle 시킨다.

```
set.seed(659)
credit_shuffle <- credit[ sample( nrow(credit) ), ]
```

#5. 데이터를 9 대 1로 나눈다.

```
train_num <- round( 0.9 * nrow(credit_shuffle), 0)

credit_train <- credit_shuffle[1:train_num , ]

credit_test <- credit_shuffle[(train_num+1) : nrow(credit_shuffle), ]

nrow(credit_train)
nrow(credit_test)
```

#6. C5.0 패키지와 훈련 데이터를 이용해서 모델을 생성한다.

```
library(C50)
```

```
credit_model <- C5.0( credit_train[ ,-17] , credit_train[ , 17], trials=100 )
```

#7. 위에서 만든 모델을 이용해서 테스트 데이터의 라벨을 예측한다.

```
credit_result <- predict( credit_model, credit_test[ , -17] )
```

#8. 이원 교차표로 결과를 확인한다.

```
library(gmodels)
```

```
CrossTable( credit_test[ , 17], credit_result )
```

#■ 실제값과 예측값 대입

```
actual_type <- credit_test[ , 17]
predict_type <- credit_result
positive_value <- 'yes'
negative_value <- 'no'
```

#■ 정확도

```
g <- CrossTable( actual_type, predict_type )
```

```
x <- sum(g$prop.tbl *diag(2)) # 정확도 확인하는 코드
x
```

#■ 카파통계량

```
#install.packages("vcd")
library(vcd)
```

```
table( actual_type, predict_type)
```

```
Kappa( table( actual_type, predict_type) )
```

#■ 민감도

```
head(credit_results)
```

```
#install.packages("caret")
library(caret)
sensitivity( predict_type, actual_type,
             positive=positive_value)
```

#■ 특이도

```
specificity( predict_type, actual_type,
              negative=negative_value)
```

#■ 정밀도

```
posPredValue( predict_type, actual_type,
```

```
positive=positive_value)
```

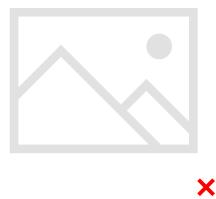
#■ 재현율

```
sensitivity( predict_type, actual_type,  
positive=positive_value)
```

문제305. 엑셀이 있으면 엑셀에 다음과 같이 정리하고 엑셀이 없으면 openoffice를 다운받아서 아래와 같이 표로 정리하시오! 어떤 알고리즘을 썼는지 기입하세요.

	trials = 1, seed=31	trials = 100, seed = 69	trials = 100, seed = 659
정확도	0.77	0.79	0.87
카파통계량	0.38	0.45	0.61
민감도	0.44	0.42	0.62
특이도	0.90	0.97	0.94
정밀도	0.65	0.87	0.78
재현율	0.44	0.42	0.62

문제306. (점심시간 문제) ROC 그래프 그리는 코드를 색깔만 변경하시오.



문제307. 위의 ROC 그래프를 다시 그리는데 seed값을 659로 주고, trials를 100으로 주고 다시 그리시오!
AUC가 넓어지는지 확인하시오~

AUC :

0.7998904 로 올라감

문제308. 위의 정확도와 cutoff를 출력하는 코드를 이용해서 독일은행 데이터의 정확도와 cutoff지점을 출력하시오!

점심시간 문제 전체 코드

+

맨 아래에 정확도와 cutoff 값을 출력하는 코드

#1. 데이터를 로드한다.

```
credit <- read.csv("credit.csv", stringsAsFactor=TRUE)
str(credit)
```

#2. 데이터에 각 컬럼들을 이해한다.

```
#라벨 컬럼 : default ---> yes : 대출금 상환 안함
#no : 대출금 상환
```

```
prop.table( table(credit$default) )
summary( credit$amount)
```

#3. 데이터가 명목형 데이터인지 확인해본다.

```
str(credit)
```

#4. 데이터를 shuffle 시킨다.

```
set.seed(31)
credit_shuffle <- credit[ sample( nrow(credit) ), ]
```

#5. 데이터를 9 대 1로 나눈다.

```
train_num <- round( 0.9 * nrow(credit_shuffle), 0)

credit_train <- credit_shuffle[1:train_num , ]
credit_test <- credit_shuffle[(train_num+1) : nrow(credit_shuffle), ]
```

#6. C5.0 패키지와 훈련 데이터를 이용해서 모델을 생성한다.

```
library(C50)
```

```
credit_model <- C5.0( credit_train[ ,-17] , credit_train[ , 17] )
```

#7. 위에서 만든 모델을 이용해서 테스트 데이터의 라벨을 예측한다.

```
credit_result <- predict( credit_model, credit_test[ , -17] )
```

#8. 이원 교차표로 결과를 확인한다.

```
library(gmodels)
```

```
CrossTable( credit_test[ , 17], credit_result )
```

#■ 실제값과 예측값 대입

```
credit_test_prob <- predict(credit_model, credit_test[ , -17], type = "prob")
```

```
credit_test_prob
```

```
# combine the results into a data frame
```

```
credit_results <- data.frame(actual_type = credit_test[ , 17],
                             predict_type = credit_result,
                             prob_yes = round(credit_test_prob[ , 2], 5),
                             prob_no = round(credit_test_prob[ , 1], 5))
```

```
credit_results
```

#3. 예측 데이터 프레임을 csv로 저장합니다.

```
# uncomment this line to output the sms_results to CSV
```

```
write.csv(credit_results, "final_results.csv", row.names = FALSE)
```

#■ 실제값과 예측값 대입

```
actual_type <- credit_test[ , 17]
predict_type <- credit_result
positive_value <- 'yes'
negative_value <- 'no'
```

#■ ROC 곡선 그리기

```
#install.packages("ROCR")
library(ROCR)
head(credit_results) # 3번째 컬럼과 4번째 컬럼의 확률을 확인한다.
pred <- prediction(predictions = credit_results$prob_yes,
                     labels = credit_results$actual_type)
pred
# ROC curves
```

```

perf <- performance(pred, measure = "tpr", x.measure = "fpr")

plot(perf, main = "ROC curve for SMS spam filter", col = "blue", lwd = 2)

# add a reference line to the graph
# 대각선 출력

abline(a = 0, b = 1, lwd = 2, lty = 2)

# calculate AUC
perf.auc <- performance(pred, measure = "auc")
str(perf.auc)
unlist(perf.auc@y.values)

# 정확도와 cutoff 출력하는 부분
# perf <- performance(pred, measure = "tpr", x.measure = "fpr")
eval <- performance(pred,"acc") # y 축을 정확도로 출력
eval # 392개의 데이터 포인트 추출
plot(eval)

```

#설명: h는 수평선, v 가 수직선의 지점

```

#Identifying the best cutoff and Accuracy
eval # x축이 cutoff이고 y축이 정확도를 그래프로 시각화하기 위한 392개의 데이터 포인트

slot(eval,"y.values") # 392개의 데이터 포인트를 출력
max <- which.max(slot(eval,"y.values")[[1]]) # 392개의 데이터 포인트중에 max 값에 해당하는
                                              # 인덱스 번호 출력

max # 61번째 인덱스 번호에 해당하는 데이터가 가장 큰 값

acc <- slot(eval,"y.values")[[1]][[max]] # y축에 정확도중에 61번째에 해당하는 값을 출력
cut <- slot(eval,"x.values")[[1]][[max]] # x축에 cutoff 값들중에서 61번째 해당하는 값을 출력
print(c(Accuracy=acc, Cutoff = cut))

```

문제309. 독일은행 데이터에서 하이퍼 파라미터 trials=1, seed=31과 trials=100, seed=659의 F1스코어가 어떻게 되는지 각각 출력하시오!

점심시간 문제 코드에 F1코드만 추가

#1. 데이터를 로드한다.

```

credit <- read.csv("credit.csv", stringsAsFactor=TRUE)
str(credit)

```

#2. 데이터에 각 컬럼들을 이해한다.

```

#라벨 컬럼 : default ---> yes : 대출금 상환 안함
#no : 대출금 상환

```

```
prop.table( table(credit$default) )
summary( credit$amount)
```

#3. 데이터가 명목형 데이터인지 확인해본다.

```
str(credit)
```

#4. 데이터를 shuffle 시킨다.

```
set.seed(659)
credit_shuffle <- credit[ sample( nrow(credit) ), ]
```

#5. 데이터를 9 대 1로 나눈다.

```
train_num <- round( 0.9 * nrow(credit_shuffle), 0)

credit_train <- credit_shuffle[1:train_num , ]

credit_test <- credit_shuffle[(train_num+1) : nrow(credit_shuffle), ]
```

#6. C5.0 패키지와 훈련 데이터를 이용해서 모델을 생성한다.

```
library(C50)
```

```
credit_model <- C5.0( credit_train[ ,-17] , credit_train[ , 17], trials = 100)
```

#7. 위에서 만든 모델을 이용해서 테스트 데이터의 라벨을 예측한다.

```
credit_result <- predict( credit_model, credit_test[ , -17] )
```

#8. 이원 교차표로 결과를 확인한다.

```
library(gmodels)
```

```
CrossTable( credit_test[ , 17], credit_result )
```

#F1 스코어

```
actual_type <- credit_test[ , 17]
```

```
predict_type <- credit_result
```

```
positive_value <- 'yes'
```

```
negative_value <- 'no'
```

```
F1_Score(actual_type, predict_type, positive_value)
```

문제310. (빅데이터 기사 시험문제) 다음 혼동행렬에서 F1스코어는 얼마인가 ?



$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

정밀도 : 0.75

재현율 : 0.6

문제311. (오늘의 마지막문제 2/15) 유방암 데이터를 악성과 양성으로 분류하는 머신러닝 모델의 정확도와 다른 성능 척도를 출력하시오!

k = 21과 k=11로 실험했을때의 차이 결과를 캡쳐해서 올리시오!

precision + recall

```
wbcd <- read.csv("wisc_bc_data.csv", header=T, stringsAsFactors=FALSE)
wbcd$diagnosis <- factor(wbcd$diagnosis,
                           levels= c("B", "M"),
                           labels=c("Benign", "Malignant") )
str(wbcd)
sample(10) # 1부터 10까지의 숫자를 랜덤으로 섞어서 출력하는 코드
wbcd_shuffle <- wbcd[ sample(569), ] # 설명: wbcd[ 행, 열 ]
wbcd_shuffle
wbcd2 <- wbcd_shuffle[ , -1 ]
str(wbcd2)

normalize <- function(x) {
  return ( (x-min(x)) / ( max(x) - min(x) ) )
}

wbcd_n <- as.data.frame( lapply( wbcd2[ , 2:31], normalize) )
```

```

nrow( wbcd_n ) # 569
train_num <- round( 0.9 * nrow(wbcd_n), 0 )
train_num # 512

wbcd_train <- wbcd_n[ 1:train_num, ]
wbcd_test <- wbcd_n[ (train_num+1) : nrow(wbcd_n), ]
nrow(wbcd_test) # 57

wbcd_train_label <- wbcd2[ 1:train_num, 1 ]
wbcd_test_label <- wbcd2[ (train_num+1) : nrow(wbcd_n), 1 ]
wbcd_test_label

# install.packages("class")
library(class)

result1 <- knn(train=wbcd_train, test=wbcd_test, cl=wbcd_train_label, k=21)
result1

data.frame( result1, wbcd_test_label)
sum( result1 == wbcd_test_label )

x <- data.frame('실제'=wbcd_test_label, '예측'=result1)
table(x)

```

	seed = 1, k= 11	seed = 1, k = 21
정확도	1	1
카파통계량	1	1
정밀도	1	1
재현율	1	1
민감도	1	1
특이도	1	1

문제312. 지난번에 만든 파생변수를 추가해서 파생변수를 추가한 후의 상관관계와 오차를 올리시오.

method = "lm" : 5.441164(승순) - 파생변수 추가후

method = "lm" : ? (내꺼) - 파생변수 추가후

method = "rf" : 3.199271(승순) - 파생변수 추가후

method = "rf" : ? (내꺼) - 파생변수 추가후

문제313. (점심시간 문제) 아무런 파생변수라도 추가해서 step 함수를 쓴 결과를 올리시오.

문제314. (오늘의 마지막 문제 2/19) 주말에 올리기

더 좋은 파생변수를 생각하거나 결측치를 다른 값으로 치환해서 조금 더 순위를 올리시오!