

목차

2021년 1월 14일 목요일 오후 3:57

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	1	R의 자료구조의 종류	바로가기

데이터 관리와 이해			
	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 베이즈의 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
단원	번호	상세목차	수업내용
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 찰린저호 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 ₁ 바로가기 ₂
	3	영어, 수학 점수로 학생 분류	바로가기
	4	SNS 의 글로 같은 성향을 가진 사람들을 분류	바로가기
단원	번호	상세목차	수업내용
10장 성능평가	1	혼동행렬을 사용한 성능척도	
	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	
단원	번호	상세목차	수업내용
7장 서포트 벡터머신	1	벡터와 벡터 머신 이론	바로가기
	2	서포트 벡터 머신 원리	바로가기
	3	서포트 벡터 머신으로 아이리스 데이터 분류	바로가기

	4	서포트 벡터 머신으로 필기체 데이터 분류	바로가기
단원		상세목차	수업내용
11장. 랜덤포레스트	1	랜덤포레스트 이론	바로가기
	2	랜덤포레스트 실습 척추 데이터	바로가기
	3	랜덤포레스트 실습 아이리스 데이터	바로가기
	4	caret 함수로 자동튜닝	바로가기
	5	seeds 값에 대한 실험	바로가기

유방암 데이터를 kmeans 로 테스트 : [바로가기](#)

k-means 함수 날코딩으로 구현하기 : [바로가기](#)

로지스틱회귀와 서포트 벡터머신과의 차이 실험 : [바로가기](#)

서포트 벡터 머신으로 미국 대학원 데이터 분류 실험 : [바로가기](#)

21.01.14

2021년 1월 14일 목요일 오후 3:53

R 설치

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

작업 디렉토리 설정 및 emp.csv를 로드하는 방법

```
> setwd("c:\\data")
> getwd()
```

emp3.csv를 c드라이브 밑에 data 폴더 밑에 둔다.

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

> (emp$sal, col = rainbow(14))
```

R studio 설치

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

```
> setwd ("c:\\data")
> emp <- read.csv("emp3.csv")
> emp
> pie (emp$sal, col=rainbow(14))
```

문제1. centos7에 R과 R studio를 설치하시오.

(root에서 수행하시오.)

<http://cafe.daum.net/oracleoracle/Sg0x/3>

21.01.18

2021년 1월 18일 월요일 오전 9:35

R 이란

- 뉴질랜드의 auckland 대학의 robert gentleman 과 Ross Ihaka 가 1995년에 개발한 소프트웨어이다. 데이터 분석을 위한 통계 및 시각화를 지원하는 무료 소프트웨어이다.

R을 사용하는 이유

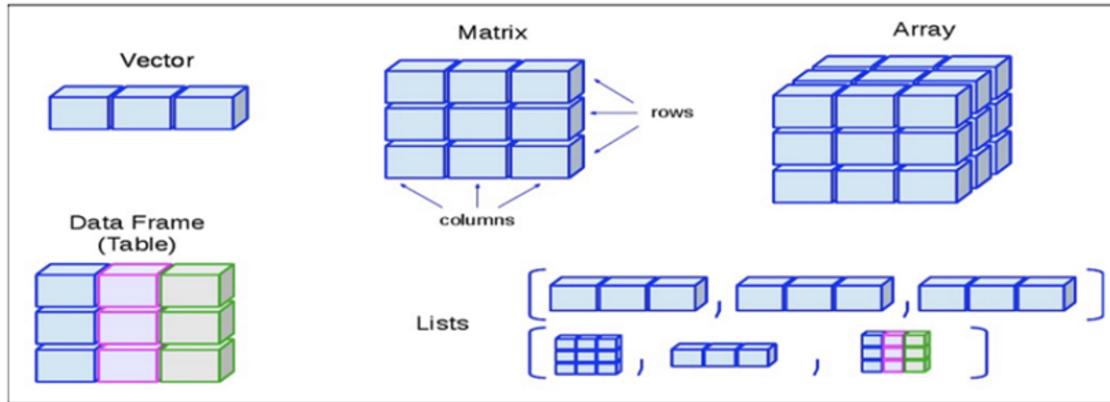
1. 무료이다.
2. data 분석을 위해서 가장 많이 사용하는 통계 플랫폼
3. 복잡한 데이터를 다양한 그래프로 표현할 수 있다.
4. 분석을 위한 데이터를 쉽게 저장하고 조작할 수 있다.
5. 누구든지 유용한 패키지를 성해서 공유할 수 있고 새로운 기능에 대한 전파가 빠르다.
6. 어떠한 os에도 설치가 가능하다.

구글의 코랩을 이용하면 GPU를 사용해서 파이썬 코드를 무료로 수행할 수 있는 장점이 있는데 코랩에서 R 코드를 지원한다.

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
 - rdbms의 테이블과 유사
5. list : 서로 다른 데이터 구조(vector, data frame, matrix, array)인 데이터 타입이 중첩된 구조

■ R 의 자료구조 그림



R을 사용하여 기본 데이터 검색

문제2. emp.csv를 리눅스 /home/scott에 올리고 emp.csv를 로드해서 emp 데이터 프레임을 만드시오.

```
setwd('/home/scott')
emp <- read.csv('emp.csv')
emp
```

문제3. emp 데이터 프레임에서 이름과 월급을 출력하시오.

```
emp[, c('ename', 'sal')]
```

문법 : emp[행, 열]

문제4. 월급이 3000인 사원들의 이름과 월급을 출력하시오.

```
emp[emp$sal == 3000, c('ename', 'sal')]
```

문제5. 월급이 2000 이상인 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 이름, 입사일을 출력하시오.

```
emp[emp$hiredate == '1981-12-11', c('ename','hiredate')]
```

문제8. emp 데이터 프레임의 구조를 확인하시오.

```
str(emp)
```

연산자 총 정리

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

ex)

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

[1] FALSE TRUE FALSE

ex)

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

[1] TRUE

combine을 사용하면 백터화 된다.

연결 연산자

오라클	R
	paste

문제9. 아래의 SQL을 R로 구현하시오.

```
select ename || '의 직업은' || job
```

```
from emp;  
  
paste(emp$ename, '의 직업은', emp$job)
```

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
in	%in%
like	grep
is null	is.na
between .. and	emp\$sal >= 1000 & emp\$sal <= 3000

문제10. 직업이 SALESMAN, ANALYST인 사원들의 이름과 월급과 직업을 출력하시오.

```
emp[emp$job %in% c('SALESMAN','ANALYST'), c('ename','sal','job')]
```

문제11. 직업이 SALESMAN, ANALYST가 아닌사원들의 이름과 월급과 직업을 출력하시오.

```
emp[!emp$job %in% c('SALESMAN','ANALYST'), c('ename','sal','job')]
```

문제12. 부서번호가 10, 20 번인 사원들의 이름과 월급과 부서번호를 출력하시오.

```
emp[emp$deptno %in% c(10,20), c('ename','sal','deptno')]
```

문제13. 커미션이 null인 원들의 이름과 월급과 커미션을 출력하시오.

```
emp[is.na(emp$comm), c('ename','sal','comm')]
```

1. NA (결손값) : is.na()

2. NaN (비수치) : `is.nan()`
 - Not a Number
3. NULL(아무것도 없음) : `is.null()`

머신러닝 학습시에는 결손값을 다른 값으로 대체하는 것이 아주 중요하다.

문제14. 커미션션이 NA가 아닌 사원들의 이름과 월급과 커미션을 출력하시오.

```
emp[!is.na(emp$comm), c('ename','sal','comm')]
```

문제15. 월급이 1000에서 3000 사이인 사원들의 이름과 월급을 출력하시오.

```
select ename, sal  
from emp  
where sal between 1000 and 3000;
```

```
emp[emp$sal >= 1000 & emp$sal <= 3000, c('ename','sal')]
```

문제16. 월급이 1000에서 3000 사이가 아닌 사원들의 이름과 월급을 출력하시오.

```
emp[!(emp$sal >= 1000 & emp$sal <= 3000), c('ename','sal')]
```

문제17. 이름의 첫번째 철자가 A로 시작하는 사원들의 이름과 월급을 출력하시오.

```
emp[grep('^A',emp$ename), c('ename','sal')]
```

^ 는 시작을 나타낸다.

문제18. 이름의 끝글자가 T로 끝나는 사원들의 이름과 월급을 출력하시오.

```
emp[grep('T$',emp$ename), c('ename','sal')]
```

\$ 는 끝을 나타낸다.

문제19. 이름의 두번째 철자가 M인 사원들의 이름과 월급을 출력하시오.

```
emp[grep('^.M',emp$ename), c('ename','sal')]
```

. 은 자릿수 하나를 나타낸다.

문제20. 이름의 세번째 철자가 L인 사원들의 이름과 월급을 출력하시오.

```
emp[grep('^..L',emp$ename), c('ename','sal')]
```

중복 제거

오라클	R
distinct	unique

문제21. 부서번호를 출력하는데 중복을 제거해서 출력하시오.

```
unique(emp$deptno)
```

or

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

```
data.table('부서번호' = unique(emp$deptno))
```

문제22. 직업을 출력하는데 중복을 제거해서 출력하시오.

```
data.table('직업' = unique(emp$job))
```

정렬 작업

오라클	R
order by	1. data frame 에서 order 옵션 2. doBy 패키지를 설치하고 orderBy 함수를 사용

문제23. 이름과 월급을 출력하는데 월급이 높은 사원부터 출력하시오.

```
emp[order(emp$sal, decreasing = T), c('ename','sal')]
```

data frame에 내장 되어 있는 order 옵션을 사용하면 된다.

문제24. 이름과 입사일을 출력하는데 먼저 입사한 사원부터 출력하시오.

```
emp[order(emp$hiredate, decreasing = F), c('ename','hiredate')]
```

문제25. 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하는데 월급이 높은 사원부터 출력하시오.

1. 직업이 SALESMAN인 사원들을 출력해서 result 변수에 담는다.

```
result <- emp[emp$job == 'SALESMAN', c('ename','sal','job')]  
result
```

2. result 변수의 월급을 높은 것 부터 출력한다.

```
result[order(result$sal, decreasing = T), ]
```

문제26. 부서번호가 30번인 사원들의 이름과 월급과 입사일을 출력하는데 먼저 입사한 사원부터 출력하시오.

```
result <- emp[emp$deptno == 30, c('ename','sal','hiredate')]  
result[order(result$hiredate, decreasing = F), ]
```

문제27. doBy 패키지를 이용해서 이름과 월급을 출력하는데 월급이 높은 것부터 출력하시오.

```
install.packages('doBy')  
library(doBy)  
  
orderBy(~-sal, emp[, c('ename','sal')])
```

문제28. 직업이 ANALYST가 아닌 사원들의 이름과 월급과 직업을 출력하는데 월급이 높은 사원부터 출력되게 하시오.

```
orderBy(~-sal, emp[emp$job != 'ANALYST', c('ename','sal','job')])
```

문제29. 범죄 발생요일(crime_day.csv)를 R로 로드해서 토요일에 발

생하는 범죄유형, 범죄건수를 출력하는데 범죄 건수가 높은 것부터 출력하시오.

```
crime_day <- read.csv('crime_day.csv')
result <- crime_day[crime_day$DAY == 'SAT', c('C_T','CNT')]
orderBy(~-CNT, result)
```

현재 R에서 사용되고 있는 result와 같은 변수의 목록 확인방법

```
ls()
```

변수를 지울 때

```
rm(result)
result
```

문제30. 살인이 일어나는 장소와 건수를 출력하는데 건수가 높은 것부터 출력하시오.

```
crime_loc <- read.csv('crime_loc.csv')

result <- crime_loc[crime_loc$범죄 == '살인', c('장소','건수')]
orderBy(~-건수,result)
```

문제31. 강도가 일어나는 장소와 건수를 출력하는데 건수가 높은 것부터 출력하시오.

```
result <- crime_loc[crime_loc$범죄 == '강도', c('장소','건수')]
orderBy(~-건수,result)
```

함수의 종류

1. 문자함수
2. 숫자함수
3. 날짜함수
4. 변환함수
5. 일반함수

문자함수

오라클	R
upper	toupper

lower	tolower
substr	substr
replace	gsub

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.

```
select ename, substr(ename, 1, 3)
  from emp;
```

```
data.table(이름 = emp$ename, 철자 = substr(emp$ename, 1, 3))
```

문제34. 이름, 월급을 출력하는데 월급을 출력할 때 숫자 0을 *로 출력하시오.

```
select ename, replace (sal, 0, '*')
  from emp;
```

```
data.table(이름 = emp$ename, 월급 = gsub(0, '*', emp$sal))
```

문제35. 이름, 월급을 출력하는데 월급을 출력할 때 숫자 0, 1, 2를 *로 출력하시오.

```
select ename, regexp_replace(sal, '[0-2]', '*')
  from emp;
```

```
data.table(이름 = emp$ename, 월급 = gsub('[0-2]', '*', emp$sal))
```

숫자함수

오라클	R
round	round
trunc	trunc
mod	%%
power	2^3 (2의 3승)

문제36. 6의 9승을 출력하시오.

6^9

문제37. 10을 3으로 나눈 나머지값을 출력하시오.

10%3

문제38. 이름과 연봉을 출력하는데 연봉은 월급의 12를 곱해서 출력하고 컬럼명은 둘 다 한글로 이름, 연봉 이라고 출력되게 하시오.

data.table(이름 = emp\$ename, 연봉 = emp\$sal * 12)

문제39. 위의 결과를 다시 출력하는데 round 함수를 이용하여 백의 자리에서 반올림 하시오.

data.table(이름 = emp\$ename, 연봉 = round(emp\$sal * 12, -3))

```
3   5   7   0   0   .   7   3   8  
-5  -4  -3  -2  -1  0  1  2  3
```

문제40. 위의 결과에서 반올림 하지 않고 백자리 이후를 버려서 출력하시오.

data.table(이름 = emp\$ename, 연봉 = trunc(emp\$sal * 12, -3))

trunc는 소수점 이하만 버릴 수 있다. 소수점 이전은 버리지 못하므로 위의 문제는 수행할 수 없다.

```
round(122.5)  
[1] 122
```

```
round(123.5)  
[1] 124
```

날짜함수

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

문제41. 오늘 날짜를 출력하시오.

[Sys.Date\(\)](#)

문제42. 이름, 입사한 날짜부터 오늘까지 총 몇일 근무했는지 출력하시오.

```
select ename, sysdate - hiredate
      from emp;
```

```
data.table(emp$ename, Sys.Date() - as.Date(emp$hiredate))
```

오라클 : to_date

R : as.Date

문제43. 오늘 날짜의 달의 마지막 날짜를 출력하시오.

```
select last_day(sysdate)
      from dual;
```

```
install.packages('lubridate')
library(lubridate)
```

```
ceiling_date(Sys.Date(), 'months') - days(1)
```

floor_date(Sys.Date(), 'months') : 2021-01-01

Sys.Date() : 2021-01-18

ceiling_date(Sys.Date(), 'months') : 2021-02-01

변환함수

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

문제44. 이름, 입사한 요일을 출력하시오.

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

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

format의 옵션

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

문제45. 11월에 입사한 사원들의 이름과 입사일을 출력하시오.

```
select ename, hiredate
      from emp
     where to_char(hiredate, 'mm') = '11';

emp[format(as.Date(emp$hiredate), '%m') == '11', c('ename', 'hiredate')]
```

문제46. 오늘부터 100달 뒤에 돌아오는 날짜를 출력하시오.

```
select add_months(sysdate, 100)
      from dual;
```

[Sys.Date\(\) + months\(100\)](#)

days(숫자), months(숫자), years(숫자)

문제47. 오늘부터 100달 뒤에 돌아오는 날짜의 요일을 출력하시오.

```
select to_char(add_months(sysdate, 100), 'day')
      from dual;

format(Sys.Date() + months(100), '%A')
```

문제48. 내가 무슨 요일에 태어났는지 출력하시오.

```
select to_char(to_date('1993/10/15', 'RRRR/MM/DD'), 'day')
```

```
from emp;  
  
format(as.Date('1993/10/15'), '%A')
```

일반함수

오라클	R
nvl 함수	is.na
decode 함수	ifelse
case 함수	ifelse

문제49. 이름, 월급, 등급을 출력하는데 월급이 1500 이상이면 등급을 A로 출력하고 아니면 B로 출력하시오.

```
data.table(emp$ename, emp$sal, ifelse(emp$sal >= 1500, 'A', 'B'))
```

ifelse는 결측치 데이터를 다른 데이터로 대체할 때 아주 유용한 함수이다.

머신러닝시에 모델이 학습할 때 좋 학습 데이터를 제공하기 위하여 결측치에 값을 채워주는게 중요하다.

문제50. 이름, 월급, 보너스를 출력하는데 보너스가 입사한 년도가 1980년도이면 A로 출력하고 1981년도이면 B 를 출력하고 1982년도 이면 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. 아래의 페이지에서 캐글 데이터 중 student-mat.csv를 내려 받아서 R로 로드하고 나이와 등급을 출력하는데 나이가 15~18은 A, 19~20은 B, 21~22은 C 로 출력하시오.

<http://cafe.daum.net/oracleoracle/Sg0u/4>

```
data.table(studentmat$age,  
          ifelse(studentmat$age >= 15 & studentmat$age <= 18, 'A',  
                ifelse(studentmat$age >= 19 & studentmat$age <= 20, 'B','C')))
```

21.01.19//

2021년 1월 19일 화요일 오전 9:41

그룹함수

오라클	R
max	max
min	min
sum	sum
avg	mean
count	length (세로) table (가로)

워킹 디렉토리 지정

```
setwd('c:/data')
```

지정된 워킹 디렉토리 확인

```
getwd()
```

emp3.csv를 로드하여 emp 데이터 프레임 생성

```
emp <- read.csv('emp3.csv')
```

문제52. 최대월급을 출력하시오.

```
max(emp$sal)
```

문제53. 직업이 SALESMAN인 사원들의 최대월급을 출력하시오.

```
result <- emp[emp$job == 'SALESMAN','sal']
```

```
max(result)
```

또는

```
max(emp[emp$job == 'SALESMAN','sal'])
```

문제54. 20번 부서번호인 사원들 중 최소월급을 출력하시오.

```
min(emp[emp$deptno == 20,'sal'])
```

문제55. 직업, 직업별 최대월급을 출력하시오.

```
select job, max(sal)
  from emp
 group by job;

aggregate(sal~job, emp, max)
```

문제56. 부서번호, 부서번호별 토탈월급을 출력하시오.

```
select deptno, sum(sal)
  from emp
 group by deptno;

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(doBy)

orderBy(~-토탈월급, result)
```

문제59. 직업, 직업별 인원수를 출력하시오.

```
select job, count(*)
  from emp
 group by job;

result <- aggregate(empno~job, emp, length)

names(result) <- c('직업','인원수')

result
```

문제60. 위의 결과를 다시 출력하는데 인원수가 높은 것부터 출력하

시오.

orderBy(~-인원수, result)

문제61. 직업과 직업별 인원수를 가로로 출력하시오.

table(emp\$job)

문제62. 위의 결과를 원형 그래프로 시각화 하시오.

pie(table(emp\$job), col=rainbow(5), density = 100)

문제63. 부서번호, 직업, 부서번호별 직업별 토탈월급을 출력하시오.

```
select deptno, job, sum(sal)
      from emp
     group by deptno, job;
```

aggregate(sal~deptno+job, emp, sum)

문제64. 입사한 년도(4자리), 입사한 년도별 평균월급을 출력하시오.

```
select to_char(hiredate, 'RRRR'), avg(sal)
      from emp
     group by to_char(hiredate, 'RRRR');
```

x <- aggregate(sal~format(as.Date(emp\$hiredate),'%Y'), emp, mean)

names(x) <- c('입사년도', '평균월급')

x

문제65. 위의 결과에서 소수점 이하는 나오지 않게 하시오.

x\$평균월급 <- trunc(x\$평균월급)

x

x 데이터 프레임의 평균월급의 값의 소수점 이하를 버리고 x 데이터 프레임의 평균월급에 넣는다.

책 129 페이지 유방암 데이터에 대한 설명

- 이 데이터는 위스콘신 대학교의 연구원이 기부했으며 유방 종양의 미세침습인물 디지털

이미지에서 측정한 값이 들어있다. 이 값은 디지털 이미지에 존재하는 세포 핵의 특성을 나타낸다. 569개의 암조직 검사 예시가 들어있으며 각 예시는 32개의 특징을 가진다.

문제66. 머신러닝 때 사용할 유방암 데이터를 R로 로드하고 전체 건수가 몇건인지 확인하시오.

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

```
nrow(wisc) # 전체 건수 확인
```

```
ncol(wisc) # 컬럼의 개수 확인
```

```
str(wisc) # 데이터 프레임 구조 확인
```

diagnosis 는 유방암 환자가 양성(B)인지 악성인지(M)를 나타내는 라벨(정답) 컬럼이다.

문제67. 유방암 데이터에서 양성 환자가 몇명이고 악성 환자가 몇명인지 확인하시오.

```
aggregate(id~diagnosis, wisc, length)
```

```
table(wisc$diagnosis)
```

문제68. 위의 결과를 원형 그래프로 시각화 하시오.

```
pie(table(wisc$diagnosis), col = c('green','red'))
```

문제69. 직업, 직업별 토탈월급을 가로로 출력하시오.

세로출력 :

```
aggregate(sal~job, emp, sum)
```

가로출력 :

```
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.

세로출력 :

```
aggregate(sal~format(as.Date(emp$hiredate), '%Y'), emp, sum)
```

가로출력 :

```
tapply(emp$sal, format(as.Date(emp$hiredate), '%Y'), sum)
```

문제72. 위의 결과를 막대그래프로 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.

```
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;
```

```
attach(emp)
```

```
tapply(sal,list(job,deptno),sum)
```

attach(emp)를 사용해서 emp\$sal, emp\$job, emp\$deptno 와 같이 작성하지 않고 sal, job, deptno 로만 작성할 수 있다.

위의 데이터를 가지고 그래프를 그리려면 NA 값을 숫자 0으로 변경해 줘야 한다.

문제74. 위의 결과의 NA를 숫자 0으로 출력되게 하시오.

```
x <- tapply(sal,list(job,deptno),sum)
```

```
x[is.na(x)] <- 0
```

```
x
```

문제75. 위의 결과에서 컬럼명만 출력하고 로우명만 출력하시오.

```
colnames(x)
```

```
rownames(x)
```

문제76. 위의 결과 데이터를 막대 그래프로 시각화 하시오.

```
barplot(x, col = rainbow(5), legend = rownames(x), beside = T, density = 50)
```

legend : 그래프의 설명 박스

beside=T : 직업별로 각각 막대그래프가 생성

문제77. 입사한 년도(4자리), 입사한 년도별 직업별 토탈월급을 출력하시오.

```
tapply(emp$sal,list(emp$job,format(as.Date(emp$hiredate),'%Y'))),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
```

문제79. 위의 과를 막대 그래프로 그리시오.

```
barplot(x,col=rainbow(5),legend=rownames(x),beside=T,density=50)
```

문제80. 입사한 년도(4자리)를 막대 그래프의 x 축으로 구성하고 부서 번호의 평균월급을 y축으로 구성되게 하는데 입사한 년도별로 부서번호 10, 20, 30 번이 각각 그려지게 하시오.

```
x<-tapply(emp$sal,list(emp$deptno,format(as.Date(emp$hiredate),'%Y'))),mean)
```

```
x[is.na(x)]<-0
```

```
x
```

```
barplot(x,col=rainbow(3),legend=rownames(x),beside=T,density=50)
```

문제81. 직업과 직업별 토탈월급을 가지고 원형그래프를 그리시오.

```
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))
```

explode : 벌어짐 정도

문제83. 위의 그래프 결과에 직업 옆에 비율도 같이 출력되게 하시오.

```
x <- tapply(emp$sal, emp$job, sum) # 직업별 토탈월급 가로 출력
```

```
x2 <- aggregate(sal~job, emp, sum) # 직업별 토탈월급 세로 출력
```

```
pct <- round(x2$sal / sum(emp$sal)*100, 1)
```

```
pct
```

```
# [1] 20.7 14.3 28.5 17.2 19.3
```

```
job_label <- paste(x2$job, ':', pct, '%')
```

```
job_label
```

```
# [1] "ANALYST : 20.7 %"  "CLERK : 14.3 %"  "MANAGER : 28.5 %"
```

```
# [4] "PRESIDENT : 17.2 %" "SALESMAN : 19.3 %"
```

```
pie3D(x, explode = 0.1, labels = job_label)
```

```
pie(x, labels = job_label, col = rainbow(5))
```

문제84. 유방암 데이터의 diagnosis의 양성과 악성의 비율을 원형 그래프로 그리시오.

```
x1 <- tapply(wisc$id, wisc$diagnosis, length)  
x2 <- aggregate(id~diagnosis, wisc, length)
```

```
pct <- round(x2$id / nrow(wisc)*100, 1)
```

```
w_label <- paste(x2$diagnosis, ':', pct, '%')
```

```
pie3D(x1, explode = 0.1, labels = w_label)
```

```
pie(x1, col = c('green','red'), labels = w_label)
```

그래프를 그릴때는 가로로 출력되는 결과 데이터를 가지고 그려야 한다.

미율을 나타내는 라벨을 만들때는 세로로 출력되는 결과 데이터를 가지고 만든다.

SQL과 R 비교(조인)

오라클	R
equi join	merge
non equi join	merge
outer join	merge
self join	merge

dept.csv를 내려받아 dept라는 변수에 로드하시오.

```
dept <- read.csv('dept.csv')
dept
```

문제85. 이름과 부서위치를 출력하시오.

```
select e.ename, d.loc
  from emp e, dept d
 where e.deptno = d.deptno;

x <- merge(emp, dept, by = 'deptno')

x[ , c('ename','loc')]
```

x 데이터 프레임에서 ename와 loc만 출력

문제86. 부서위치가 DALLAS인 사원들의 이름과 월급과 부서위치를 출력하시오.

```
x <- merge(emp, dept, by = 'deptno')

x[x$loc == 'DALLAS' , c('ename','sal', 'loc')]
```

문제87. 커미션이 NA인 사원들의 이름과 부서위치와 커미션을 출력 하시오.

```
x <- merge(emp, dept, by = 'deptno')

x[is.na(x$comm), c('ename','loc','comm')]
```

문제88. outer join으로 이름과 부서위치를 출력하는데 아래의 SQL과 동일한 결과가 출력되게 하시오.

```
select e.ename, d.loc
  from emp e, dept d
 where e.deptno (+) = d.deptno;
```

```
x <- merge(emp, dept, by = 'deptno', all.y = T)

x[ , c('ename','loc')]
```

all.y = T : dept 테이블의 데이터가 모두 출력

문제89. 아래의 SQL의 결과를 R로 구현하시오.

```
select e.ename, d.loc
  from emp e, dept d
 where e.deptno = d.deptno (+);

x <- merge(emp, dept, by = 'deptno', all.x = T)

x[ , c('ename','loc')]
```

문제90. 아래의 SQL의 결과를 R로 구현하시오.

```
select e.ename, d.loc
  from emp e full outer join dept d
    on (e.deptno = d.deptno);

x <- merge(emp, dept, by = 'deptno', all = T)

x[ , c('ename','loc')]
```

문제91. 이름을 출력하고 그 옆에 자기의 직속상사의 이름을 출력하시오.

```
select 사원.ename, 관리자.ename
  from emp 사원, emp 관리자
 where 사원.mgr = 관리자.empno;

x <- merge(emp, emp, by.x = 'mgr', by.y = 'empno')

x[ , c('ename.x','ename.y')]
```

문제92. 위의 결과를 다시 출력하는데 자기의 월급이 자기의 직속상사의 월급보다 더 큰 사람들만 출력하시오.

```
x <- merge(emp, emp, by.x = 'mgr', by.y = 'empno')

x[x$sal.x >= x$sal.y, c('ename.x','ename.y')]
```

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
b <- graph.data.frame(a, directed = T)
plot(b)

```

문제94. 위의 그래프를 구글의 googleVis를 이용해서 시각화 하시오.

```

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.

```
x <- merge(emp, dept, by = 'deptno', all = T)
```

세로출력 :

```
aggregate(x$sal~x$loc, x, sum, na.action = na.pass)
```

가로출력:

```
tapply(x$sal, x$loc, sum)
```

문제96. 위의 결과를 막대그래프로 시각화 하시오.

```

x <- merge(emp, dept, by = 'deptno', all = T)
x1 <- tapply(x$sal, x$loc, sum)

```

```
barplot(x1, col = rainbow(4), ylim = c(0, 12000))
```

가로로 출력한 결과 데이터를 이용해서 그래프를 그린다.

문제97. 아래와 같이 부서위치별 년도별 토탈월급을 출력하시오.

	BOSTON	CHICAGO	DALLAS	NEW YORK
1980	NA	NA	800	NA
1981	NA	9400	5975	7450
1982	NA	NA	3000	1300
1983	NA	NA	1100	NA

```
install.packages('lubridate')
library(lubridate)

year(emp$hiredate) # hiredate에서 연도만 추출
```

```
x <- merge(emp, dept, by = 'deptno', all = T)
tapply(x$sal, list(year(x$hiredate), x$loc), sum)
```

문제98. 위의 결과를 다시 출력하는데 NA대신 숫자 0이 출력되게 하시오.

```
x <- merge(emp, dept, by = 'deptno', all = T)
x1 <- tapply(x$sal, list(year(x$hiredate), x$loc), sum)
x1[is.na(x1)] <- 0
x1
```

문제99. 위의 결과를 막대그래프로 시각화 하시오.

```
x <- merge(emp, dept, by = 'deptno', all = T)
x1 <- tapply(x$sal, list(year(x$hiredate), x$loc), sum)
x1[is.na(x1)] <- 0
barplot(x1, col = rainbow(4), beside = T, legend = rownames(x1),
        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)
```

line_no : 호선 (1호선~4호선)
time : 시간
in : 승차 인원수
out : 하차 인원수

문제101. 위의 line에서 line_no를 중복제거해서 출력하시오.

```
unique(line$line_no)
```

문제102. 위의 지하철 승하차수 보를 가지고 구글 모션차트를 그리시오.

```
t1 <- gvisMotionChart(line, idvar = 'line_no', timevar = 'time')
plot(t1)
```

문제103. 부서번호, 부서번호별 토탈월급을 출력하는데 아래의 SQL처럼 전체 토탈월급이 출력되게 하시오.

```
select deptno, sum(sal)
  from emp
 group by rollup(deptno);

rbind(aggregate(sal~deptno, emp, sum), c(' ', sum(emp$sal)))
```

rbind는 두개의 결과를 위 아래로 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)

x1 <- aggregate(sal~deptno, emp, mean)
cbind(aggregate(sal~deptno, emp, sum), x1$sal)
```

문제105. 위의 결과의 컬럼명을 아래와 같이 출력되게 하시오.

부서번호 토탈월급 평균월급

1	10	8750	2916.667
2	20	10875	2175.000
3	30	9400	1566.667

```
x1 <- aggregate(sal~deptno, emp, mean)
x2 <- cbind(aggregate(sal~deptno, emp, sum), x1$sal)
names(x2) <- c('부서번호', '토탈월급', '평균월급')
x2
```

문제106. 아래의 SQL의 결과를 R로 구현하시오.

```
select job, sum(sal) 토탈월급,
       max(sal) 최대월급,
       min(sal) 최소월급,
       avg(sal) 평균월급,
       count(*) 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(empno~job, emp, length)  
z <- cbind(a, b$sal, c$sal, d$sal, e$empno)  
  
names(z) <- c('직업', '토탈월급', '최대월급', '최소월급', '평균월급', '인원수')  
z
```

21.01.20

2021년 1월 20일 수요일 오전 9:45

집합연산자

오라클	R
union all	rbind
union	rbind + unique
intersect	intersect
minus	setdiff

문제107. 아래의 SQL의 결과를 R로 구현하시오.

```
select ename, sal, deptno
      from emp
     where deptno in (10, 20)
union all
select ename, sal, deptno
      from emp
     where deptno = 20;
```

```
a <- emp[emp$deptno %in% c(10, 20), c('ename', 'sal', 'deptno')]
b <- emp[emp$deptno == 20, c('ename', 'sal', 'deptno')]
rbind(a, b)
```

rbind는 두개의 과를 위아래로 어서 출력하는 함수

R과 Rstudio 세팅

```
candidates <- c( Sys.getenv("R_PROFILE"),
                 file.path(Sys.getenv("R_HOME"), "etc", "Rprofile.site"),
                 Sys.getenv("R_PROFILE_USER"),
                 file.path(getwd(), ".Rprofile") )

Filter(file.exists, candidates)
[1] "C:/PROGRA~1/R/R-40~1.3/etc/Rprofile.site"
```

메모장을 관리자 권한으로 실행후 위 경로로 이동

```
setwd('c://data')
```

작성 후 저장

문제108. 아래의 SQL을 R로 구현하시오.

```

select job, sum(sal)
  from emp
 group by job
union all
select null as job, sum(sal)
  from emp;

a <- aggregate(sal~job, emp, sum)
b <- c(' ', sum(emp$sal))

rbind(a, b)

```

문제109. 아래의 SQL의 결과를 R로 구현하시오.

```

select ename, sal, deptno
  from emp
 where deptno in (10, 20)
union
select ename, sal, deptno
  from emp
 where deptno = 10;

a <- emp[emp$deptno %in% c(10, 20), c('ename', 'sal', 'deptno')]
b <- emp[emp$deptno == 10, c('ename', 'sal', 'deptno')]
x <- unique(rbind(a, b))

orderBy(~ename, x)

```

SQL의 union의 경우는 두 SQL의 결과를 위 아래로 연결하면서 첫번째 컬럼인 ename을 기준으로 데이터 정렬하고 중복된 데이터를 제거한다.

문제110. 아래의 SQL의 결과를 R로 구현하시오.

```

select ename, sal, deptno
  from emp
 where deptno in (10, 20)
minus
select ename, sal, deptno
  from emp
 where deptno = 10;

install.packages('dplyr')
library(dplyr)

a <- emp[emp$deptno %in% c(10, 20), c('ename', 'sal', 'deptno')]
b <- emp[emp$deptno == 10, c('ename', 'sal', 'deptno')]
x <- setdiff(a, b)

```

```
orderBy(~ename, x)
```

R에 내장되어 있는 setdiff 함수를 사용하면 안되고 dplyr 패키지의 setdiff를 이용해야 한다.

union처럼 minus도 데이터 정렬을 한다.

문제111. 아래의 SQL의 결과를 R로 구현하시오.

```
select ename, sal, deptno
  from emp
 where deptno in (10, 20)
intersect
select ename, sal, deptno
  from emp
 where deptno = 10;
```

```
a <- emp[emp$deptno %in% c(10, 20), c('ename', 'sal', 'deptno')]
b <- emp[emp$deptno == 10, c('ename', 'sal', 'deptno')]
x <- intersect(a, b)
```

```
orderBy(~ename, x)
```

SQL의 서브쿼리를 R로 구현하기

오라클 서브쿼리의 종류 3가지

1. 단일행 서브쿼리
2. 다중행 서브쿼리
3. 다중 컬럼 서브쿼리

문제112. JONES의 월급보다 더 많은 월급을 받는 사원들의 이름과 월급을 출력하시오.

```
select ename, sal
  from emp
 where sal > (select sal
                  from emp
                 where ename = 'JONES');
```

```
jones_sal <- emp[emp$ename == 'JONES', c('sal')]
emp[emp$sal > jones_sal, c('ename', 'sal')]
```

문제113. 아래의 SQL을 R로 구현하시오.

```
select ename, sal
  from emp
```

```
where sal = (select max(sal)
             from emp)
```

```
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')
univ_max <- max(univ$등록금)
```

```
univ[univ$등록금 == univ_max, c('학교명', '등록금')]
```

문제115. KING에게 보고하는 사원들의 이름과 월급을 출력하시오.

```
select ename, sal
      from emp
     where mgr = (select empno
                   from emp
                  where ename = 'KING');
```

```
king_empno <- emp[emp$ename == 'KING', c('empno')]
emp[emp$mgr == king_empno, c('ename', 'sal')]
```

```
ename sal
NA <NA> NA
2 BLAKE 2850
3 CLARK 2450
4 JONES 2975
```

위의 결과에서 NA 부분을 제거하시오.

```
king_empno <- emp[emp$ename == 'KING', c('empno')]
na.omit(emp[emp$mgr == king_empno, c('ename', 'sal')])
```

문제116. 관리자인 사원들의 이름을 출력하시오.

```
select ename
      from emp
     where empno in (select mgr
                      from emp);
```

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

```
data.table(이름 = emp[emp$empno %in% emp$mgr, 'ename'])
```

문제117. 관리자가 아닌 사원들의 이름을 출력하시오.

```
select ename
  from emp
 where empno not in (select mgr
                        from emp
                      where mgr is not null);
```

```
data.table(이름 = emp[!emp$empno %in% emp$mgr, 'ename'])
```

문제118. 아파트에서 가장 많이 발생하는 범죄유형이 무엇인지 출력하시오.

(crime_loc.csv 사용)

```
crime_loc <- read.csv('crime_loc.csv')
x <- crime_loc[crime_loc$장소 == '아파트', c('범죄', '건수')]

x[x$건수 == max(x$건수), ]
```

문제119. 지하철에서 가장 많이 발생하는 범죄유형이 무엇인지 출력하시오.

```
x <- crime_loc[crime_loc$장소 == '지하철', c('범죄', '건수')]

x[x$건수 == max(x$건수), ]
```

문제120. 지하철에서 발생하는 범죄유형과 건수를 출력하는데 건수가 높은 것 부터 출력하시오.

```
x <- crime_loc[crime_loc$장소 == '지하철', c('범죄', '건수')]
orderBy(~-건수, x)
```

문제121. 위의 결과에서 첫번째 행만 출력하시오.

```
x <- crime_loc[crime_loc$장소 == '지하철', c('범죄', '건수')]
x1 <- orderBy(~-건수, x)

x1[1, ]
```

문제122. 문제120번의 결과에서 첫번째 행부터 세번째 행까지 출력

하시오.

```
x <- crime_loc[crime_loc$장소 == '지하철', c('범죄', '건수')]  
x1 <- orderBy(~-건수, x)  
  
x1[c(1:3), ]
```

문제123. 강력범죄가 가장 많이 발생하는 요일은 언제인지 확인하시오.

(crime_day.csv 사용)

trimws 함수

- 양쪽 공백 제거

```
x <- crime_day[trimws(crime_day$C_C) == '강력범죄', ]  
x1 <- orderBy(~-CNT, x)  
x1 [1, 'DAY']
```

문제124. 살인기수가 많이 발생하는 요일을 1위부터 3위까지 출력하시오.

```
x <- crime_day[trimws(crime_day$C_T) == '살인기수', ]  
x1 <- orderBy(~-CNT, x)  
x1 [c(1:3), 'DAY']
```

순위 출력하기

문법 : rank 함수

ex)

이름, 월급, 월급에 대한 순위를 출력하시오.

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

orderBy(~순위, x)

rank에 '-'를 사용하면 월급이 높은 것 부터 출력된다.

ties.method

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

데이터를 높은 순위로 부여한다.

3. max : 예를 들어 2등이 두면이면 둘 다 3등으로 출력한다.

오라클의 dense_rank와 같은 함수

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

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

```
orderBy(~순위, x)
```

문제125. 월요일에 많이 발생하는 범죄, 건수, 순위를 출력하시오.

(crime_day.csv 사용)

```
c_mon <- crime_day[crime_day$DAY == 'MON', ]  
x <- data.table(범죄 = c_mon$C_T, 건수 = c_mon$CNT,  
                 순위 = dense_rank(-c_mon$CNT))
```

```
orderBy(~순위, x)
```

문제126. 여자들이 많이 걸리는 암과 그 건수와 순위를 출력하는데 모든암은 제외하시오.

(cancer2.csv 사용)

```
cancer2 <- read.csv('cancer2.csv')
```

```
c <- cancer2[cancer2$성별 == '여자' & cancer2$암종 != '모든암', ]  
x <- data.table(c$암종, c$환자수, 순위 = dense_rank(-c$환자수))  
View(unique(orderBy(~순위, x)))
```

막대 그래프 그리기

문제127. emp 테이블의 월급으로 기본적인 막대 그래프를 그리시오.

```
barplot(emp$sal)
```

문제128. 위의 그래프의 제목을 Salary Bar Chart라고 이름을 붙이

시오.

```
barplot(emp$sal, main = 'Salary Bar Chart')
```

문제129. 막대 그래프 x 축에 사원이름을 붙이시오.

```
barplot(emp$sal, main = 'Salary Bar Chart', names.arg = emp$ename)
```

문제130. 막대 그래프의 x축과 y축의 이름을 각각 이름, 월급 이라고 하시오.

```
barplot(emp$sal, main = 'Salary Bar Chart', names.arg = emp$ename,
       xlab = '이름', ylab = '월급')
```

문제131. 막대 그래프의 색깔을 변경 하시오.

```
barplot(emp$sal, main = 'Salary Bar Chart', names.arg = emp$ename,
       xlab = '이름', ylab = '월급', col = 'Green Yellow', density = 80)
```

카페에 R 색상표 확인

<https://cafe.daum.net/oracleoracle/Sg0x/165>

<https://cafe.daum.net/oracleoracle/Sg0x/166>

문제132. 창업건수.csv를 R로 로드하고 치킨집의 창업건수를 막대 그래프로 시각화 하시오.

```
create_cnt <- read.csv('창업건수.csv', header = T)
```

```
barplot(create_cnt$치킨집, main = '년도별 치킨집 창업건수',
       names.arg = create_cnt$X, col = 'Green Yellow', density = 80,
       ylim = c(0, 1600))
```

문제133. 치킨집의 폐업건수로 막대 그래프를 그리시오.

```
drop_cnt <- read.csv('폐업건수.csv', header = T)
```

```
barplot(drop_cnt$치킨집, main = '년도별 치킨집 폐업건수',
```

```
names.arg = drop_cnt$X, col = 'Green Yellow', density = 80,  
ylim = c(0, 4000))
```

문제134. 치킨집의 창업건수와 폐업건수를 같이 막대 그래프로 시각화 하시오.

```
x <- rbind(create_cnt$치킨집, drop_cnt$치킨집)
```

```
barplot(x, main = '년도별 치킨집 창업/폐업건수',  
names.arg = drop_cnt$X, col = c('Green Yellow', 'Hot pink'),  
density = 80, ylim = c(0, 4000), beside = T)
```

문제135. 카페가 얼마나 창업을 하고 얼마나 폐업을 하는지 막대그래프로 시각화 하시오.

```
x <- rbind(create_cnt$커피음료, drop_cnt$커피음료)
```

```
barplot(x, main = '년도별 카페 창업/폐업건수',  
names.arg = drop_cnt$X, col = c('Green Yellow', 'Hot pink'),  
density = 80, ylim = c(0, 4000), beside = T)
```

문제136. 위의 막대그래프를 앞으로 편하게 그릴 수 있도록 함수로 만들어서 실행되게 하시오.

R 함수 생성 문법 :

```
kjm_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 창업/폐업건수',  
  names.arg = drop_cnt$X, col = c('Green Yellow', 'Hot pink'),  
  density = 80, ylim = c(0, 4000), beside = T)  
}
```

```
kjm_auto()
```

문제137. 창업건수.csv를 로드해서 치킨집의 창업건수를 막대그래프로 그리는 함수를 생성하시오.

```
a_bar <- function() {  
  create_cnt <- read.csv('창업건수.csv', header = T)
```

```
barplot(create_cnt$치킨집, main = '년도별 치킨집 창업건수',
       names.arg = create_cnt$X, col = c('Green Yellow'),
       density = 80, ylim = c(0, 3000), beside = T)
}

a_bar()
```

원형 그래프 그리기

문제138. 사원 테이블의 월급을 원형 그래프로 그리시오.

```
pie(emp$sal)
```

문제139. 위의 그래프를 다시 출력하는데 누구의 월급인지 명시되게 하시오.

```
pie(emp$sal, main = 'Salary Pie Chart', labels = emp$ename, col = rainbow(14))
```

문제140. 위의 그래프의 월급에 비율을 붙여서 출력하시오.

```
sal_labels <- round(emp$sal / sum(emp$sal) * 100, 1)
sal_labels1 <- paste(emp$ename, sal_labels, '%')
pie(emp$sal, main = 'Salary Pie Chart',
    labels = sal_labels1, col = rainbow(14))
```

라인 그래프 그리기

- 시간 순서에 따른 데이터의 변화를 볼 때 유용한 그래프

문제141. 아래의 데이터로 plot(점) 그래프를 그리시오.

```
cars <- c(1, 3, 6, 4, 9)
cars
```

```
plot(cars)
```

문제142. 위의 그래프에 파란색 선을 그리시오.

```
plot(cars, type = 'o', col = 'blue')
```

type = 'o' : 선을 그린다.

문제143. 차와 트럭의 판매된 댓수를 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))
lines(trucks, type = 'o', pch = 22, lty = 2, col = 'red')
```

pch = 21 : 동그라미

pch = 22 : 네모

lty = 1 : 직선

lty = 2 : 점선

문제144. 가로축을 월, 화, 수, 목, 금으로 변경하시오.

```
plot(cars, type = 'o', col = 'blue', ylim = c(0, 12), axes = FALSE, ann = FALSE)
lines(trucks, type = 'o', pch = 22, lty = 2, col = 'red')

axes = FALSE : x축과 y 축을 지운다.
ann = FALSE : 축이름을 지운다.
```

새로운 축을 생성하는 방법

```
axis(1, at=1:5, lab = c('mon', 'tue', 'wed', 'thu', 'fri')) # x축
axis(2) # y축
```

레전드 붙이기

```
legend(2, 10, c('cars', 'trucks'), col = c('blue', 'red'), cex = 0.8,
      pch = 21:22, lty = 1:2)
```

cex는 글씨 크기, 레전드 안에 pch 21(동그라미), pch22(네모), lty1(직선), lty2(점선)을 표시 한다.

문제145. 치킨집의 창업건수를 이용해서 라인 그래프를 그리시오.

(x축을 연도로 두고 y축을 창업건수로 두고 만드시오.)

```
create_cnt <- read.csv("창업건수.csv", header=T)
g_range <- range(0, max(create_cnt$치킨집))
plot( create_cnt$치킨집, type='o', col='blue', ylim=g_range, axes=FALSE, ann=FALSE )

axis( 1, at=1:10, lab= create_cnt$년도 )
axis(2) # y 축 생성
legend( 8, g_range[2], '창업' , col= 'blue', cex=0.8, pch=21, lty=1 )
```

문제146. 치킨집의 창업건수와 폐업건수를 같이 출력되게 하시오.
(라인 그래프로 그리시오.)

```
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$X)
axis(2)
legend(5, max(drop_cnt$치킨집), c('폐업','창업'), col = c('red','blue'),
       cex = 0.5, pch = 22:21, lty = 2:1)
```

문제147. 막대 그래프, 원형 그래프, 라인 그래프를 그리는 함수를
만드시오.

21.01.21

2021년 1월 21일 목요일 오전 9:45

R studio를 실행하면 자동으로 emp3.csv가 로드

<https://cafe.daum.net/oracleoracle/Sg0x/190>

막대그래프를 조동화 스크립트로 구현하기

<https://cafe.daum.net/oracleoracle/Sg0x/193>

```
library(data.table)

bar <- 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]
  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)
}

bar()
```

문제148. 위의 막대 그래프 자동화 코드를 이용해서 원형 그래프 자동화 함수를 생성하시오.

```
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) )

}

```

문제149. 아래의 메뉴코드를 실행하시오.

```

x1 <- menu(c('막대그래프', '원형그래프'))
x1

```

문제150. 아래의 switch 코드를 실행하시오.

```

my_func <- function() {
  x1 <- menu(c('막대그래프', '원형그래프'),
             title = '숫자를 선택하시오.')
  switch(x1,
         gp1 = {bar()},
         gp2 = {pie()})
}
my_func()

```

my_func 함수 생성

```

my_func <- function() {

  bar <- 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] # 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')
plot(cars$cnt2, type = 'o', col = 'blue', ylim = c(0,12),
      axes = FALSE, ann = FALSE)
axis(1, at = 1:5, lab = cars$date2)
axis(2)
legend(2, 10, 'car', col = 'blue', cex = 0.8, pch = 21, lty = 1)
```

문제152. 위의 스크립트를 이용해서 윈도우에서 csv파일을 불러와서 라인 그래프를 그리는 함수를 line2라는 이름으로 생성하시오.

```
line2 <- 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]

  y_max <- max(ycol2) + 3
  y_max
  plot( ycol2, type='o', col='blue', ylim=c(0,y_max), axes=FALSE, ann=FALSE )

  axis( 1, at=1:5, lab= xcol2 )
  axis(2)
  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 추가 시키시오.

```
my_func <- function() {

  bar <- 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] # 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) )
```

```
}
```

```
line2 <- 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_max2 <- max(ycol2) -1
y_max <- max(ycol2) +3
y_max
plot( ycol2, type='o', col='blue', ylim=c(0,y_max), axes=FALSE, ann=FALSE )

axis( 1, at=1:5, lab= xcol2 )
axis(2) # y 축 생성
legend( 2, y_max2, 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() }
)
}

```

히스토그램 그래프 그리기

- 히스토그램은 하나의 속성에 대한 데이터의 분포를 시각적으로 표현하는 그래프이다.

```

usedcars <- read.csv('usedcars.csv')
View(usedcars)

```

1. 전체 건수를 확인한다.
- nrow(usedcars)
2. 컬럼이 몇개인지 확인한다.
- ncol(usedcars)
3. 각 컬럼들의 데이터에 대한 통계정보를 확인한다.
- summary(usedcars)
4. 중고차 가격에 대해서 히스토그램 그래프를 그리시오.
- hist(usedcars\$price)
5. 히스토그램 x 축의 간격을 조정하시오.
- hist(usedcars\$price, breaks = seq(0, 26000, by = 2000))

문제154. 중고차의 마일리지로 히스토그램 그래프를 그리시오.

```
hist(usedcars$mileage, main= paste('중고차 마일리지 히스토그램 그래프'),  
      breaks=seq(0, 152000, by=8000), ylim=c(0,50))
```

문제155. 위의 그래프 코드를 함수로 생성해서 윈도우 탐색기가 열리게 하면서 데이터를 불러와서 히스토그램 그래프가 그려지게 하시오.

```
hist2 <- 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  
  
  xcol <- colnames(table[xcol_num]) # x축 컬럼명을 담는 코드  
  
  xcol2 <- table[,xcol] # x 축 컬럼의 데이터를 xcol2에 넣는다.  
  max_m = max(xcol2)*1.1  
  
  hist( xcol2 , main= paste(xcol,'의 히스토그램 그래프') , col='orange', density=80 )  
  
}
```

사분위수 그래프 (박스 플롯 그래프)

- 박스플롯은 많은 데이터를 그림을 이용하여 집합의 범위와 중앙값을 빠르게 확인 할 수 있으며 통계적으로 이상치값이 있는지 빠르게 확인이 가능한 시각화 기법이다.

평균값과 중앙값과 최빈값만으로는 데이터 분석을 하기 부족한 경우가 있다.

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

ex)

어느 농구단의 감독이 아래 3명의 농구선수중에 한명을 선택하려고 하는데 아래의 3명의 선수의 게임별 점수를 가지고 한명을 고른다면 어떤 선수를 골라야 하는지 확인하시오.

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 )
```

문제156. 위의 농구선수 3명의 득점 점수의 평균값, 중앙값, 최빈값을 각각 구하시오.

```
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 )
```

```
mean(x1)  
mean(x2)  
mean(x3)
```

```
median(x1)  
median(x2)  
median(x3)
```

```
names(table(x1))[table(x1) == max(table(x1))]  
names(table(x2))[table(x2) == max(table(x2))]  
names(table(x3))[table(x3) == max(table(x3))]
```

보통은 위 3개의 통계치로 한명의 선수를 선택할 수 있는데 위와 같은 경우는 평균값, 중앙값, 최빈값으로는 특정 선수를 선택하기가 곤란한 경우 이다.

이런 경우에 필요한 데이터 분석 방법

- 각 선수의 점수의 데이터 분포가 어떻게 분포 되었는지 분포 방식을 측정할 수 있으면 결정을 내릴 때 크게 도움을 줄 수 있다.

데이터의 분포를 확인하는 방법

1. 범위
2. 사분위수 범위
3. 히스토그램 그래프

범위 : 데이터의 폭을 확인할 때 사용한다.

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

범위는 그 자체 데이터의 폭만 설명할 뿐 그 안에서 데이터가 분포되는 방식은 설명해 주지 않고 이상치에 민감하다.

3번째 선수의 경우 30점인 이상치 때문에 범위가 넓어져 버렸기 때문에 분석하기가 어려워 졌기 때문에 이상치로 부터 멀어질 필요가 있다.

이상치로 부터 멀어지고 가운데 있는 데이터에만 집중하게 해주는게 사분위수범위 이다.

문제157. 이상치 데이터(30점)을 가지고 있는 3번째 선수의 점수 데이

터로 사분위수 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

문제158. 위와 같이 사분위수 그래프를 그리면서 위의 통계치를 확인하는 이유를 확인하시오.

이상치를 제거하고 가운데 50%의 데이터에만 집중하여 문제를 우회할 수 있다.

머신러닝 모델을 학습 시킬때도 이상치를 제거하고 머신러닝 모델을 학습시킨 결과가 훨씬 더 학습률이 좋아서 모델의 판별 정확도가 더 높은 경우가 많다.

문제159. 3명의 농구 선수들의 점수를 가지고 사분위수 그래프를 그리는데 1번, 2번, 3번 선수의 그래프를 하나의 결과로 나오게 출력하시오.

```
boxplot(x1,x2,x3)
```

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 낮은 점수에 대한 기록도 보인다. 그래서 1, 2번 선수가 더 일관성이 있고 대부분의 경우에 3번 선수보다 더 높은 점수를 기록했다.

그래서 만약 세명의 선수중에 한명의 선수를 고른다면 1, 2번 선수 중에 골라야 한다.

데이터를 통계치로 분석하는 순서

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

평균값 : mean

중앙값 : median

최빈값

범위 : range

사분위수 범위 : boxplot

분산값 : var

표준편차 : sd

문제160. 위의 3명의 선수들 중에 3번째 선수는 제외하고 1번과 2번 선수중 한명을 선택하시오.

분산 : 데이터의 퍼짐정도를 수치화 한것

- 확률변수가 기댓값으로 얼마나 떨어진 곳에 분포하는지 가늠하는 숫자

표준편차

- 분산의 제곱근 값으로 이 값을 통하여 평균에 흩어진 정도를 확인할 수 있다.

```
var(x1) # [1] 3.333333  
var(x2) # [1] 2.444444
```

```
sd(x1) # [1] 1.825742  
sd(x2) # [1] 1.563472
```

두번째 선수가 첫번째 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% : 평균으로 부터 1 표준편차

정규분포 95% : 평균으로 부터 2 표준편차

정규분포 99.7 : 평균으로 부터 3 표준편차

문제161. 중고차 가격(price)에 대한 사분위수 그래프를 그리고 사분위수 통계치를 확인하시오.

```
usedcar <- read.csv('usedcars.csv')  
a <- boxplot(usedcar$price)
```

```
a$stats
```

```
[,1]  
[1,] 5980.0 -> 하한값  
[2,] 10995.0 -> 하한 사분위수값  
[3,] 13591.5 -> 중앙값  
[4,] 14906.0 -> 상한 사분위값  
[5,] 19995.0 -> 상한값
```

a\$out -> 이상치 확인

문제162. 중고차 가격의 가운데 50%에 해당하는 데이터에 집중하기 위해서 사분위수 범위의 Q1과 Q3값을 확인하시오.

quantile(usedcar\$price)

0%	25%	50%	75%	100%
3800.0	10995.0	13591.5	14904.5	21992.0
	Q1(하한사분위수)	Q2(중앙값)	Q3(상한사분위수)	

IQR : InterQuartile Range(사분위수 범위)

- Q1과 Q3의 차이인 사분위수 범위

IQR(usedcar\$price)

[1] 3909.5

$$14904.5 - 10995.0 = 3909.5$$

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

사분위수 범위는 1사분위수와 3사분위수 사이의 차이이다.

그래프 종류

1. 막대 그래프
2. 원형 그래프
3. 라인 그래프
4. 히스토그램 그래프
5. 사분위 그래프(박스 그래프)
6. 기타 그래프
 - a. 지도 그래프
 - b. 워드 클라우드 그래프
 - c. 소리를 시각화

지도 그래프

map 패키지를 설치하고 중국지도만 확대해서 출력하시오.

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

```
map('world')
```

문제163. 미국지도를 시각화 하시오.

```
map('world','usa')
```

워드 클라우드 그리기

감정분석을 위한 시각화로 워드 클라우드를 사용한다.

R로 워드클라우드를 그리려면 R java를 설치해야 한다.

1. 아래의 사이트에서 java 64비트를 다운로드 받는다.
 - <http://www.java.com/en/download/manual.jsp>
2. 자바 설치시 대상 폴더 변경으로 설치
3. 아래와 같이 환경설정을 한다.
 - Sys.setenv(JAVA_HOME="C:\\Program Files\\Java\\jre1.8.0_281")
4. R java를 설치한다.
 - install.packages('rJava')
 - library(rJava)

KoNLP 설치 방법

<https://cafe.daum.net/oracleoracle/Sg0x/220>

<https://cafe.daum.net/oracleoracle/Sg0x/221>

<https://cafe.daum.net/oracleoracle/Sg0x/218>

문제164. jobs.txt를 워드 클라우드로 시각화 하시오.

```
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("c:\\\\data") # 워킹디렉토리를 소환
jobs <- readLines('jobs.txt')

nouns <- extractNoun(jobs) # 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, # F로 하게 되면 큰글씨부터 출력이 되면서  
           family="malgun") # 중앙에서 퍼지게 한다.  
# 맑음 글씨체로 시각화 하겠다.
```

그래프 자동화

- <https://cafe.daum.net/oracleoracle/Sg0x/244>

위의 스크립트를 R studio를 실행 할 때마다 편하게 사용할 수 있도록 구현

1. 그래프 3개 자동화 스크립트를 가져온다. (my_func 함수)
2. <https://cafe.daum.net/oracleoracle/SZTZ/1990> (참고용)
3. Rprofile.site에 추가하여 R 을 실행할 때마다 자동으로 불러오게 한다.
 - 메모장을 관리자 권한으로 열고 아래의 위치로 가서 Rprofile.site를 연다.
 - C:\Program Files\R\R-4.0.3\etc
 - r <- function() {source('my_func2.R')} 추가 후 Rstudio을 다시 실행한다.
4. 히스토그램 그래프와 사분위수 그래프를 my_func.R에 추가한다.(2번에서 가져온다.)
5. View함수를 가지고 v 라는 함수를 만들고 실행한다.
 - v <- function(){View(emp)}
6. v함수를 my_func2.R의 6번째로 추가한다.

R을 활용한 머신러닝 2장

통계기반 분석 모형

1. 기술 통계 : 평균, 분산, 표준편차, 왜도 첨도, 빈도등에 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 통계기법
4. 분산 분석 : 두 개 이상의 집단간 비교를 수행할 때 집단 내의 분산의 비교로 얻은 분포를 이용하여 가설검정을 수행하는 방법
5. 주성분 분석 : 많은 변수의 분산방식의 패턴을 간결하게 표현하는 주성분 변수를 원래 변수의 선형 결합으로 추출하는 통계기법
6. 판별 분석 : 집단에 대한 정보로 부터 집단을 구별할 수 있는 판별 규칙을 만들고 다변량 기법으로 조사된 집단에 대한 정보를 활용하여 새로운 개체가 어떤 집단인지 탐색하는 통계 기법

데이터 마이닝 기반 분석 모형

머신러닝 기반 분석 모형

- 지도 학습 : 정답인 레이블(label)이 포함되어 있는 학습 데이터를 통해 컴퓨터를 학습시키는 방법 (knn(3장), 나이브 베이즈(4장), 의사결정트리와 랜덤포레스트(5장), 신경망과 서포트 벡터머신(7장), 회귀분석(6장), 연관분석(8장))
- 비지도 학습 : 입력 데이터에 대한 정답인 레이블(label)이 없는 상태에서 데이터가 어떻게 구성되었는지 알아내는 기계학습 방법 (k-means(9장))
- 강화 학습 : 환경에 대해 적응해 나가는 경험 데이터를 쌓으면서 학습해 나가는 기계학습 방법 (딥마인드의 알파고)

머신러닝 데이터 분석을 위한 R 기본 문법, 그래프, 함수 소개

1. Factor
2. 이원 교차표 (CrossTable)
3. 데이터를 R로 불러오는 여러가지 방법

Factor

팩터(factor)란 범주 변수나 순위 변수를 나타내기 위해 사용하는 특별한 종류의 벡터(vector)이다.

1. 범주(값의 목록)를 갖는 vector
2. factor() 함수를 통해서 생성
3. factor는 nominal, ordinal 형식 2가지가 존재
4. nominal은 level 순서의 값이 무의미하며 알파벳 순서로 정의된다.
ex)
`a <- c('middle', 'low', 'high')`
`a2 <- factor(a)`
`a2`
5. ordinal은 level 순서의 값을 직접 정의해서 원하는 순서를 정한다.
ex)
`a3 <- (a, order = TRUE, level = c('low', 'middle', 'high'))`
`a3`

R의 자료구조

1. vector : 같은 데이터 타입을 갖는 1차원 배열구조
 - c() 함수를 이용해서 구성할 수 있다.
ex)
`a <- c(1, 2, 3, 4, 5)`
`str(a)`
2. matrix : 같은 데이터 타입을 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

순위 데이터를 모델링하는 머신러닝 알고리즘은 순서 팩터를 기대하기 때문에 팩터를 알아야 한다.

ex)

팩터 = 일반 벡터 + level

```
a <- c('middle', 'low', 'high')
```

```
a
```

```
str(a)
```

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

```
a2
```

```
str(a2)
```

팩터는 벡터와는 다르게 순서라는 개념이 들어가 있는데 위의 a2의 경우 순서가 알파벳 순서로 abcd 순서로 순서의 개념이 들어가 있다.

```
a2[order(a2, decreasing = F)]
```

```
[1] high low middle
```

```
Levels: high low middle
```

```
a[order(a, decreasing = F)]
```

```
[1] "high" "low" "middle"
```

low middle high 순서로 순서를 부여해서 a3 factor를 구성

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

```
a3
```

```
[1] middle low high
```

```
Levels: low < middle < high
```

위와 같이 순서를 부여할 수 있어서 머신러닝 모델 학습시킬때 어떤게 악성이고 양성인지를 알려주면서 머신러닝 모델을 학습 시켜야 학습이 된다.

```
str(a3)
```

```
Ord.factor w/ 3 levels "low" <"middle" <..: 2 1 3
```

ex)

유방암 데이터의 라벨은 factor로 줘야한다.

```
wisc <- read.csv('wisc_bc_data.csv', header=T, stringsAsFactors=F)
str(wisc)
```

```
$ diagnosis      : chr "B" "B" "B" "B" ...
```

`stringsAsFactor=F` 는 `wisc_bc_data.csv`의 데이터 중에 문자형 데이터를 Factor로 변환하지 않고 문자형으로 쓰겠다라는 뜻이다. 그래서 아래와 같이 정답(label) 컬럼인 `diagnosis`가 문자형이다.

```
wisc <- read.csv('wisc_bc_data.csv', header=T, stringsAsFactors=T)
str(wisc)
```

```
$ diagnosis    : Factor w/ 2 levels "B","M": 1 1 1 1 1 1 2 1 1 ...
```

문제165. 아래의 문자 3개를 factor로 만드는데 순서를 만드는데 순서를 SEVERE MODERATE MILD 순서가 되게 만드시오.

```
SEVERE MILD MODERATE
```

```
b <- c('SEVERE','MODERATE','MILD')
```

```
b <- factor(b,order=TRUE,level=c('MILD','MODERATE','SEVERE'))
```

```
b
```

R에서 데이터를 로드하는 방법 4가지

1. csv 파일을 로드하는 방법
- 2.xlsx 파일을 로드하는 방법
3. text 파일을 로드하는 방법
4. database 와 R 과 연동하는 방법

csv 파일을 로드하는 방법

```
setwd('c:\\\\data')
library(utils)
emp <- read.csv('emp.csv', header=T, stringsAsFactor=T)
```

`read.csv` 함수는 `utils` 패키지에 내장되어 있는 함수이다.

xlsx 파일을 로드하는 방법

```
install.packages('xlsx')
library(xlsx)
dept <- read.xlsx('dept.xls', 1)
```

text 파일을 로드하는 방법

```
niv <- readLines('niv.txt')
niv
```

database 와 R 과 연동하는 방법

1. 오라클에 접속이 되는지 확인한다.
 - sys 유저로 접속하는 방법

```
C:\Users\KJM>sqlplus "/as sysdba"
```

- scott유저로 접속하는 방법

```
C:\Users\KJM>sqlplus scott/tiger
```

2. db와 연동하기 위해서 필요한 패키지를 설치한다.

- install.packages('DBI')
- install.packages('RJDBC')

- library('DBI')
- library('RJDBC')

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

오라클과 R을 연동하려면 ojdbc8.jar 파일이 있어야 한다.

3. 도스창을 열고 리스너의 상태를 확인한다.

- C:\Users\KJM>lsnrctl status

포트번호와 서비스 이름을 확인한다.

포트번호 : 1521

서비스 이름 : orcl

4. 도스창에서 위의 정보로 오라클에 접속이 되는지 확인한다.

- sys로 접속

```
C:\Users\KJM>sqlplus sys/oracle@127.0.0.1:1521/orcl as sysdba
```

- scott로 접속

```
C:\Users\KJM>sqlplus scott/tiger@127.0.0.1:1521/orcl
```

5. 아래의 R 명령어로 오라클의 데이터를 쿼리한다.

- sys

```
oracle_db <- dbConnect(driver,  
                      'jdbc:oracle:thin:@//127.0.0.1:1521/orcl',  
                      'sys','oracle')
```

- scott

```
oracle_db <- dbConnect(driver,  
                      'jdbc:oracle:thin:@//127.0.0.1:1521/orcl',  
                      'scott','tiger')
```

- 공통

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

수치변수 탐색 방법

1. summary 함수를 사용한다.

2. 히스토그램 그래프를 사용한다.
3. 산포도 그래프를 사용한다.

수치 데이터 탐색 순서

1. summary 함수를 사용하여 평균, 중앙, 최소, 최대값을 확인한다.

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

이 값들을 살펴보면 데이터의 중심이 어떻게 되는지 확인할 수 있다.

ex)

중고차 가격의 평균값, 중앙값, 최대값, 최소값이 어떻게 되는지 확인

```
car <- read.csv('usedcars.csv')
car
summary(car)
```

```
price
Min. :3800
Mean :12962
Max. :21992
```

2. 이상치가 있는지 확인한다.

이상치가 있으면 머신러닝 학습시 학습이 잘 되지 않기 때문에 확인한다.

이상치를 제거한 보편적이고 일반적인 데이터로 학습 시키는게 중요하다.

```
install.packages('outliers')
library(outliers)
outlier(car$price)
```

또 다른 방법

```
x2 <- boxplot(car$price)
x2$out
```

3. 데이터의 편향여부를 확인한다.

평균값과 중앙값을 비교해서 편향여부를 확인할 수 있다.

평균값 > 중앙값 : 이상치 때문에 평균이 올라간 것이며 데이터가 오른쪽으로 편향 된다.

평균값 < 중앙값 : 데이터가 왼쪽으로 편향 된다.

ex)

중고차 가격 데이터의 편향 여부를 확인한다.

```
summary(car$price)
Min. 1st Qu. Median Mean 3rd Qu. Max.
```

3800 10995 13592 12962 14904 21992

평균값 : 12962 < 중앙값 : 13592

ex)

중고차의 마일리지(주행거리)의 편향여부를 확인한다.

summary(car\$mileage)

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

평균값 : 44261 > 중앙값 : 36385

평균값이 중앙값보다 크다면 데이터가 오른쪽으로 편향되었으므로 이상치가 있는지 확인해 봐야 한다.

outlier(car\$mileage)

[1] 151479

4. 히스토그램 그래프를 확인하여 데이터를 시각화 한다.

ex)

자동화 스크립트 r()을 실행

5. 사분위 그래프를 확인해 본다.

ex)

자동화 스크립트 r()을 실행

머신러닝 학습 시킬때 이상치를 꼭 확인해야 한다.

6. 왜도값과 첨도값을 확인한다.

- 왜도 : 데이터의 좌우로 기울어짐 정도

왜도값 > 0 : 오른쪽으로 꼬리가 길다.

왜도값 < 0 : 왼쪽으로 꼬리가 길다.

- 첨도 : 위 아래 뾰족한 정도

첨도값이 3에 가까울수록 정규분포에 해당하고

3보다 작은 경우는 완만한 곡선

3보다 크면 뾰족한 곡선

가급적 데이터가 정규분포형태를 보여야 학습이 잘 된다.

ex)

중고차의 주행거리의 왜도값을 확인하시오.

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

[1] 1.231805

왜도값이 0 보다 크므로 오른쪽으로 꼬리가 긴 경우이다. (평균값 > 중앙값)

7. 산포도(산점도) 그래프

두 변수간의 관계를 파악할 때 사용하는 그래프이다.

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

빅데이터 기사시험 : 4-51

- 두 데이터(변량)간의 상관관계 유무를 xy 평면상에 시각적으로 나타내는 그림

ex)

사원 테이블의 커미션과 월급과의 관계를 산포도 그래프로 그리시오.

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

그래프를 보면 커미션을 받는 사원들의 월급이 대체적으로 작은 것을 확인할 수 있다. 이 사원들의 직업은 SALESMAN이라서 커시면으로 수익을 올리기 때문이다.

문제166. 중고차의 주행거리가 높으면 중고차의 가격이 낮아지는지 산포도 그래프로 확인하시오.

```
plot (car$mileage, car$price, pch=21, col='blue', bg='blue')
```

mileage(주행거리)가 높을수록 가격이 낮아지는 음의 상관관계를 보이고 있다.

```
cor(car$mileage, car$price) # 상관관계 확인
```

[1] -0.8061494

<https://cafe.daum.net/oracleoracle/Sg0x/277> (참고)

문제167. 산포도 그래프를 자동화 스크립트에 7번에 추가하시오.

이원교차표 확인하는 방법

1. 관계의 관찰 : 이원교차표

두 명목 변수간의 관계를 관찰 하기 위해 이원 교차표를 사용한다.

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

ex)

```
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 해당 셀의 상대적 비율을 나타냄
4. 카이제곱 통계
 - CrossTable(x=emp\$deptno, y=emp\$job, chisq=TRUE)

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 2500이상인 사원들과 월급이 2500보다 작은 사원들의 분포를 확인한다.

```
library(data.table)
data.table(emp$sal, emp$sal >= 2500)
emp$sal_tf <- emp$sal >= 2500
emp
```

emp테이블에 sal_tf라는 새로운 컬럼과 데이터를 만들었다.

이 sal_tf는 기존 데이터로 만든 새로운 파생변수이다.

```
library(gmodels)
CrossTable(emp$job, emp$sal_tf)
```

문제169. 중고차의 모델별로 색깔이 보수적인 색깔이 많은지 아닌지를

이원교차표로 확인하시오.

보수적인 색깔 : Black, Gray, Silver, White

```
car$color_tf <- car$color %in% c('Black', 'Gray', 'Silver', 'White')
```

```
CrossTable(car$model, car$color_tf)
```

2장 내용

1. Factor 이해
2. R 로 데이터를 로드하는 4가지 방법
3. 수치형 데이터를 관찰하는 7가지 방법
4. 이원교차표 확인하는 방법

21.01.25

2021년 1월 25일 월요일 오전 9:40

문제170. plotly패키지로 박스 그래프를 시각화하는데 농구선수 2번과 농구선수 3번을 같이 시각화 하시오.

```
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)
```

```
library(plotly)

subplot(
  add_markers = plot_ly(y=~x2,type='box', name='선수2'),
  add_markers = plot_ly(y=~x3,type='box', name='선수3')
)
```

문제171. 선수2번의 데이터만 박스 그래프로 시각화 하시오.

```
x2 = c(7,9,9,10,10,10,11,11,13)
```

```
plot_ly(y=~x2,type='box', name='선수2')
```

```
plot_ly(y=~x2,type='bx', name='선수2')
```

에러: Trace type must be one of the following:

```
'scatter', 'bar', 'box', 'heatmap', 'histogram', 'histogram2d', 'histogram2dcontour', 'contour',
'scatterternary', 'violin', 'funnel', 'waterfall', 'image', 'pie', 'sunburst', 'treemap', 'funnelarea', 'scatter3d',
'surface', 'isosurface', 'volume', 'mesh3d', 'cone', 'streamtube', 'scattergeo', 'choropleth', 'scattergl',
'splom', 'pointcloud', 'heatmapgl', 'parcoords', 'parcats', 'scattermapbox', 'choroplethmapbox',
'densitymapbox', 'sankey', 'indicator', 'table', 'carpet', 'scattercarpet', 'contourcarpet', 'ohlc', 'candlestick',
'scatterpolar', 'scatterpolargl', 'barpolar', 'area'
```

bar : 막대 그래프

pie : 원형 그래프

scatter : 산포도 그래프

histogram : 히스토그램 그래프

문제172. plot_ly로 농구 선수2번의 점수 데이터로 라인 그래프를 그리시오.

```
x2 = c(7,9,9,10,10,10,10,11,11,13)
```

```
plot_ly(y=~x2,type='scatter', name='선수2', mode='dot')
```

문제173. 2019년 특정 기업의 주가 데이터를 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
)
```

머신러닝 수업을 위해 알아야하는 기본 R 문법

1. 팩터(factor)

- 머신러닝 학습을 위해서는 반드시 데이터를 문자형이 아닌 팩터로 변환해줘야 하기 때문에 팩터에 대해 이해하고 있어야 한다.
 - 팩터 : 벡터(vector) + 순서(순위)
2. R 로 데이터를 로드하는 4가지 방법
 3. 수치형 데이터를 관찰하는 7가지 방법
 4. 이원교차표 확인하는 방법

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

```
CrossTable(car$model,car$conservation)
```

카이제곱 검정 정의 : 카이제곱 검정값은 편차의 제곱값을 기대 빈도로 나눈값들의 합이다.

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

ex)

아래의 2x2 크로스 집계표를 보고 두 변수가 연관성이 있는지 살펴보시오.

관측빈도	당뇨	정상	전체
비만	10	10	20
정상	10	10	20
전체	20	20	40

정상체중	15	65	80
전체	25	75	100

귀무가설 : 두 변수는 연관성이 없다.

대립가설 : 두 변수는 연관성이 있다.

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

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

카이제곱값 : 8.33, 자유도 1일 때 p-value

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

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

카이제곱값이 높을수록 확률이 낮아지고 두 변수의 연관성이 많다.

카이제곱값이 낮을수록 확률이 높아지고 두 변수의 연관성이 적다.

```
car$conservation<-car$color %in% c("Black","Gray","Silver","White")
```

```
CrossTable(car$model,car$conservation, chisq=TRUE)
```

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.

위의 경우는 확률이 93%로 셀 개수의 변화는 단지 우연때문이고 차 모델과 색깔은 실제 연관성이 있지 않을 가능성이 높다.

R에서의 if 문 사용법

```
if(조건식) {
```

조건식이 True일 때 실행되는 식

}

```
else if(조건식) {
```

조건식이 True일 때 실행되는 식

```
    }
else {
    위의 조건식들이 만족하지 않는 경우 실행되는 식
}
```

ex)
if문을 이용해서 피타고라스의 직각 삼각형 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 loop 문 사용법

for(루프변수 in 반복할 리스트) {반복할 실행문}

ex)
`for (i in 1:10) {print(i)}`

ex)
위의 for 문을 함수로 만들어서 실행하시오.

```
aaa <- function(x) {for (i in 1:x) {print(i)}}
aaa(5)
```

ex)
별을 3개 출력하시오.

`rep('★',3)`

문제174. 아래와 같이 별이 출력되는 함수를 생성하시오.

```
star <- function(x) {for (i in 1:x) {print(rep('★',i))}}
star(5)
```

문제175. 우리나라에 알려진 흡연율이 20% 일 경우 300명을 표본 추출하여 300명의 설문 조사를 통하여 관측된 흡연자의 비율이 22.4%인

지를 검정하는 예제이다. 설문 결과 흡연자는 68명이고 비흡연자는 232명이라 가정하면 귀무가설은 "흡연율은 22.4%이다." 이고 대립가설은 "흡연율은 22.4%가 아니다." 이다. 둘중에 어떤것을 채택해야 할지 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
```

카이제곱 검정 결과 도출된 p-value 값이 0.9118로써 0.05 보다 크므로 귀무가설을 채택한다. 설문결과 관측된 흡연율을 가정된 흡연율(22.4%)를 따른다고 할 수 있다.

문제176. 다음 중 관찰된 빈도가 기대되는 빈도와 유의미하게 다른지 검정하기 위해 사용되는 검정으로 가장 옳은 것을 고르시오.

1. Z-검정
2. T-검정
3. ANOVA
4. 카이제곱 검정

4번

Knn 알고리즘



knn 알고리
즘

- k nearest neighbor 의 약자로 k개의 최근접 이웃이라는 뜻이다.
머신러닝 지도학습에 분류에 해당하는 알고리즘이다.

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

Knn 알고리즘의 장단점

- 장점 : 단순하고 효율적이다. 모델을 훈련시키지 않는다.
- 단점 : 모델을 생성하지 않아서 특징과 클래스 간의 관계를 이해하는 능력이 제한된다.
적절한 k값을 모델 개발자가 직접 알아내야 한다.

Knn 의 원리

- 새로 들어온 데이터가 기존 데이터중에서 (악성종양, 양성종양) 어느 데이터에 더 인접해 있는지 거리를 계산해서 가장 가까운 거리에 있는 데이터를 자기 이웃으로 선택한다.

거리를 계산할 때 사용하는 수학식 : 유클리드 거리 계산식

군집간의 거리 계산

연속형 변수 거리 :

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

유클리드 거리 공식을 R로 구현하기

1. 두 점의 자표를 지정한다.

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

2. 두 점 사이의 거리를 구한다.

- a. $\sqrt{(5-2)^2 + (6-4)^2}$ (그래프)
- b. a = c(2,4)
b = c(5,6)

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

3. 위의 거리를 구하기 쉽도록 distance라는 함수를 만들어서 아래와 같이 실행되게 한다.

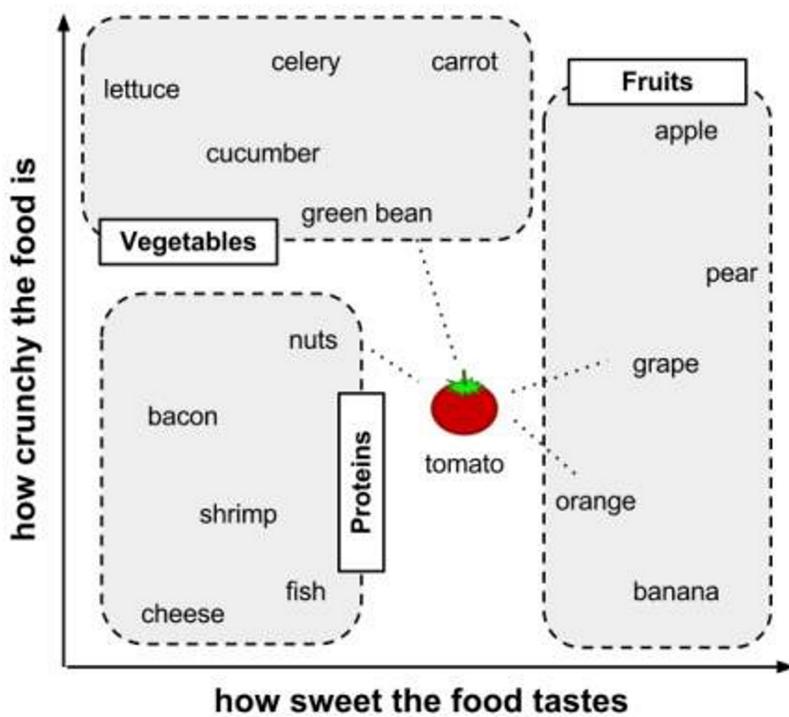
```
distance <- function(a,b) {  
    return (sqrt(sum((a-b)^2)))  
}  
  
distance(a,b)
```

문제177. 아래의 3차원 데이터의 a와 b 지점 사이의 거리를 구하시오.

```
a = c(0, 3, 2)  
b = c(2, 0, 0)
```

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

문제178. 위에서 만든 distance 함수를 이용해서 여러개의 지점과
c(4,4) 지점과의 거리를 각각 비교하시오.



$a = c(1,5)$ 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
```

```
[1] 3.162278 2.828427 1.000000 2.236068 2.236068 4.242641
```

문제179. 위의 거리 중에서 가장 작은 값을 출력하시오.

```
min(temp)
```

문제180. 토마토와 가장 가까운 거리에 있는 음식종류가 무엇인지 확인하기 위해 거리를 계산하여 fruits 데이터 프레임에 추가하시오.

```
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)

```
temp <- c() # 거리를 담을 변수
for (i in 1:6) {
  temp <- append(temp, distance(c(fruits$단맛[i],fruits$아삭한맛[i])), 토마토))
}
temp
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)
View(fruits)
```

문제182. 위의 결과에서 순위가 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))]
```

위에서 숫자 3이 Knn의 k의 갯수이다.

Knn 머신러닝 알고리즘을 이용하여 유방암 데이터의 악성과 양성 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(유클리드 거리계산 공식)으로 모델 생성

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

5단계 : 모델 성능 개선 : 정확도가 제일 좋은 k값을 알아내기

1단계 : 데이터 수집

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

반지름 / 질감 / 둘레 / 넓이 / 매끄러움 / 조밀성 / 오목함 / 오목점 / 대칭성 / 프랙탈 차원

지도학습이므로 정답이 있는 라벨 컬럼이 있는데 이 라벨컬럼이 diagnosis 이다.

라벨 : 양성(B), 악성(M)

2단계 : 데이터 탐색

1. 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

wisc_bc_data.csv

컬럼 번호: 2

2. 이상치가 얼마나 있는지 확인

<https://cafe.daum.net/oracleoracle/Sg0x/327>

```
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) ) )  
  
}
```

모델의 정확도가 잘 나오지 않으면 이상치 데이터를 삭제하고 학습시킬 것을 고려해야 한다.

3. 결측치가 많은 컬럼이 무엇인지 확인한다.

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

만약 결측치가 많은 데이터가 있다면 결측치를 다른 값으로 치환하거나 삭제를 고려해야 한다.

3단계 : 모델 훈련

1. 데이터를 로드한다.

```
wbcd <- read.csv('wisc_bc_data.csv', header=T, stringsAsFactors=FALSE)
```

```
str(wbcd)
```

2. diagnosis(정답컬럼)을 factor로 변환한다.

```
wbcd$diagnosis <- factor(wbcd$diagnosis,  
  levels = c('B','M'),
```

```
labels = c('Benign', 'Malignant')  
str(wbcd)
```

3. 데이터를 shuffle 시킨다.

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

4. 데이터에서 환자번호id컬럼을 제외 시킨다.

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

5. 데이터를 정규화 한다.

정규화 : 몸무게와 키와 같이 서로 단위가 다른 데이터를 전부 0~1사이로 데이터로 맞춰주는 것

예를들어 몸무게는 80인데 키가 172이면 키가 더 큰 숫자를 가지고 있기 때문에 데이터의 영향도가 커지기 때문에 모든 컬럼을 0~1 사이의 데이터로 변경해주어야 한다.

정규화 방법 2가지 : 1. scale 함수 사용 / 2. min/max 함수 사용

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

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

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

shuffle 시킨 wbcd2 데이터의 2번 컬럼부터 31번째 컬럼까지의 데이터를 normalize 함수에 적용 시켜서 데이터를 변환하고 데이터 프레임으로 구성

6. 훈련 데이터와 테스트 데이터로 wbcd_n 데이터를 9대1로 나눈다.

```
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
```

훈련 데이터를 모델 학습 시키기 위해서 필요한 데이터이고 테스트 데이터는 학습된 모델이 잘 학습했는지 테스트하기 위해서 필요한 데이터이다.

7. 훈련데이터 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=wbcd_train, test=wbcd_test, cl=wbcd_train_label, k=21)

result1
```

train=훈련 데이터, test=테스트 데이터, cl =훈련데이터의 라벨

result1에 테스트 데이터에 대한 예상 정답이 들어간다.

k를 21로 줘서 만들었는데 이숫자의 제일 좋은 숫자는 우리가 알아내야 한다.

```
data.frame(result1, wbcd_test_label) # 2개를 서로 비교
```

```
sum(result1 == wbcd_test_label) # 53
```

문제184. 아래의 결과를 캡처해서 올리시오.

```
> table(x)
예측
실제      Benign Malignant
Benign      37      3
Malignant    1     16
```

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

```
table(x)
```

21.01.26

2021년 1월 26일 화요일 오전 9:43

1단계 : 데이터 수집 : 데이터 출처, 데이터 설명

2단계 : 데이터 탐색과 준비

- a. 정답 라벨 데이터의 분포를 확인 (원형그래프)
- b. 이상치가 존재하는지 확인 : 이상치 제거 고려
- c. 결측치가 존재하는지 확인 : 다른 데이터로 치환을 고려
- d. 문자형 데이터인지 숫자형 데이터인지 문자형과 숫자형이 섞여있는지 확인

3단계 : 데이터로 모델 훈련 : Knn 알고리즘(유클리드 거리계산 공식)으로 모델 생성

- a. 데이터를 로드한다 (stringsAsFactors=FALSE)
- b. 라벨(정답)컬럼만 팩터로 변환
- c. 데이터를 shuffle 시킨다
- d. 학습에 도움이 안되는 데이터를 제외 (환자번호)
- e. 데이터를 정규화 한다 (1. scale 함수, minmax 함수)
- f. 데이터를 훈련 데이터와 테스트 데이터로 나눈다.

데이터 분할을 하는 이유 : 모델(모형)이 주어진 데이터에 대해서만 높은 성능을 보이는 과대적 합의 문제를 예방하여 2종 오류인 잘못된 귀무가설을 채택하는 오류를 방지하는데 목적이 있다.

데이터 분할 3가지

1. 훈련용 데이터
 2. 검증용 데이터 : 모형의 학습과정에서 모형이 제대로 학습되었는지 중간에 검증을 실시하고, 과대적합과 과소적합의 발생여부를 확인하여 모형의 튜닝에도 사용한다.
 3. 테스트용 데이터
- g. 훈련 데이터를 가지고 Knn 모델을 생성한다.

Knn의 값을 우리가 직접 알아내야 한다. 위의 유방암 데이터를 잘 분류하는 즉 데이터에 맞는 적절한 k 값을 우리학 실험으로 알아내야 한다.

이러한 k 값을 '하이퍼 파라미터' 라고 한다.

머신러닝 모델을 학습 시킬때 필요한 파라미터 2가지

1. 파라미터 : 모델 내부에서 확인이 가능한 변수로 데이터를 통해서 산출이 가능한 값
 - ex) 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 설정해주는 값.

- ex) knn 에서의 k 값, 신경망에서의 학습률, 의사결정트리에서의 깊이

데이터를 잘 분할해주는 R 패키지 caret 소개

- createPartition 함수를 이용하면 shuffle과 데이터 분할을 동시에 해주기 때문에 shuffle를 따로 하지 않아도 된다.

```
install.packages("caret")
library(caret)
wbcn <- read.csv('wisc_bc_data.csv')
nrow(wbcn) # 569

train_num <- createDataPartition(wbcn$diagnosis, p=0.9, list=F)

train_data <- wbcn[train_num, ]
test_data <- wbcn[-train_num, ]

nrow(train_data) # 513
nrow(test_data) # 56

table(train_data$diagnosis)
table(test_data$diagnosis)

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

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

5단계 : 모델 성능 개선 : 정확도가 제일 좋은 k값을 알아내기

유방암 데이터 분류에 Knn알고리즘을 사용한 이유 : 데이터를 확인했을 때 라벨컬럼을 제외하고는 전부 수치형 데이터로 구성되어있기 때문에 Knn알고리즘을 사용한다.

문제185. 다음은 분석 모형을 정의할 때 설정하는 것이다. 이를 설명한 것으로 가장 적절한 것은 무엇인지 고르시오.

- 모델 내부에서 확인이 가능한 변수로 데이터를 통해서 산출이 가능한 값이다.
- 예측을 수행할때 모델에 의해 요구되는 값들이다.
- 주로 예측자에 의해 수작업으로 측정되지 않는다.

1. 신경망 학습에서의 학습률
2. 파라미터
3. 신경망 학습의 배치사이즈
4. 정규화 파라미터

문제186. seed 값을 설정한 상태에서 모든 자리에서 동일한 결과가 나오도록 유방암 판별 모델을 생성하시오.

```
# 1. 데이터 로드
wbcn <- read.csv("wisc_bc_data.csv", header=T, stringsAsFactors=FALSE)

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

# 2. 데이터 shuffle
set.seed(1)
wbcn_shuffle <- wbcn[sample(nrow(wbcn)), ]

# 3. 환자번호 제외
wbcn2 <- wbcn_shuffle[-1]

# 4. 데이터를 min/max 정규화
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 9대1로 나눈다.
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 예측
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 확인
# 7-1
x <- data.frame('실제'=wbcn_test_label, '예측'=result1)
table(x)

# 7-2 (이원교차표)
library(gmodels)
g2 <- CrossTable(x=wbcn_test_label, y=result1)
print (g2$prop.tb[1] + g2$prop.tb[4]) # 정확도
```

다른 데이터를 Knn 모델에 학습시키기(iris 데이터셋)

<https://cafe.daum.net/oracleoracle/Sg0x/353>

문제187. 아이리스의 종, 아이리스의 종별 건수를 출력하시오.

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

문제188. createDataPartition 함수를 이용해서 아이리스 데이터를 훈련 데이터와 테스트 데이터로 나누시오.

(훈련 데이터 80%, 테스트 데이터 20%)

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

```
train_num <- createDataPartition(iris$Species, p=0.8, list=F)
```

```
train_data <- iris[train_num, ]
test_data <- iris[-train_num, ]
```

```
nrow(train_data)
nrow(test_data)
```

문제189. 위의 아이리스 데이터를 오스트리아 빈대학교에서 만든 knn 함수에 입력하여 테스트 데이터의 라벨을 예측하고 이원 교차표를 출력하시오.

문제190. wine 데이터를 R로 불러와서 전체 건수가 몇건인지 확인하시오.

```
wine <- read.csv('wine.csv')
```

```
nrow(wine)
summary(wine)
```

문제191. 와인 데이터 분류 모델을 생성하시오.

나이브 베이즈 알고리즘

나이브 베이즈 알고리즘이 사용되는 분야

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



나이브 베
이즈 설명...

독버섯을 분류하는 나이브 베이즈 알고리즘

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

1. 버섯 데이터를 R로 로드한다.

```
# stringsAsFactors=TRUE로 설정해서 문자형 데이터를 전부 factor로 변환
# 전부 문자형 데이터로 이루어져 있어서 knn 알고리즘으로 분류 할 수 없고
# 나이브 베이즈 알고리즘으로 분류를 해야한다.
mushroom <- read.csv("mushrooms.csv", header=T, stringsAsFactors=TRUE)
View(mushroom)
str(mushroom)
unique(mushroom$type) # 라벨
prop.table(table(mushroom$type)) # 독버섯과 정상버섯의 비율 확인
```

2. 8123 독버섯 데이터만 따로 빼서 mush_test.csv로 저장한다.

```
# write.csv로 mush_test 데이터를 mush_test.csv로 저장한다.
# 학습을 끝내고 만든 모델이 잘 맞추는지 확인하기 위해 분리
mush_test <- mushroom[8123, ]
mush_test
write.csv( mush_test, "mush_test.csv", row.names=FALSE )
```

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) # 8123개의 행에 23개의 열로 구성
dim(mushrooms)[1] # 8123
dim(mushrooms)[2] # 23
```

```
# 8123에서 75%에 해당하는 숫자를 train_cnt에 담는다
train_cnt <- round( 0.75*dim(mushrooms)[1] )
```

```

train_cnt
# 전체(1~8123) 중에 6092개의 숫자를 랜덤으로 추출(비복원추출)
train_index <- sample( 1:dim(mushrooms)[1], train_cnt, replace=F)

mushrooms_train <- mushrooms[ train_index, ] # 훈련데이터 구성
mushrooms_test <- mushrooms[- train_index, ] # 테스트 데이터 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 생성

# 6. 위에서 만든 모델과 테스트 데이터를 가지고 독버섯과 일반버섯을 잘 분류하는지 예측해 본다.
# mushrooms_test[, -1] <- 정답을 뺀 테스트 데이터
result1 <- predict(model1, mushrooms_test[, -1])
result1

# 7. 이원 교차표를 그려서 최종 분류 결과를 확인한다.
library(gmodels)
a1 <- CrossTable(mushrooms_test[, 1], result1)
# ↑ ↑
# 실제 예측
print(a1$prop.tb[1] + a1$prop.tb[4]) # 정확도

# 8. 위의 모델의 성능을 올리시오 !
# 나이브베이즈 함수의 하이퍼 파라미터인 라플라스값에 아주 작은 값을 임의로
# 주어서 성능을 올리면 된다.
model2 <- naiveBayes(type ~ ., data=mushrooms_train, laplace=0.0004)
result2 <- predict(model2, mushrooms_test[, -1])
CrossTable(mushrooms_test[, 1], result2)

a2 <- CrossTable(mushrooms_test[, 1], result2)
print(a2$prop.tb[1] + a2$prop.tb[4]) # 정확도

# 위의 모델에 별도로 구분해 놓은 테스트 데이터 한개(독버섯) 8123 번 데이터를 넣어서 독버섯인지 정상인지 확인하시오 !
mush_test <- read.csv('mush_test.csv', stringsAsFactors=TRUE)

```

```
result3 <- predict( model2, mush_test[ , -1] )
```

```
result3
```

문제192. 라플라스값 0.0004의 경우 [1] 0.9901526 의 정확도를 보이는 모델인데 정확도를 더 올릴수 있는 라플라스 값을 알아내시오.

```
model2 <- naiveBayes(type~ . , data=mushrooms_train, laplace=0.000005)
result2 <- predict( model2, mushrooms_test[ , -1] )
CrossTable( mushrooms_test[ ,1], result2)
```

```
a2 <- CrossTable( mushrooms_test[ ,1], result2)
print(a2$prop.tb[1] + a2$prop.tb[4])
```

<https://cafe.daum.net/oracleoracle/Sg0x/389> (참조)

21.01.27

2021년 1월 27일 수요일 오전 9:42

나이브 베이즈 머신러닝 실습2 (영화 장르 예측)

1. 워킹 디렉토리에 영화(movie.csv)를 로드한다.

```
movie <- read.csv('movie.csv', header=T, stringsAsFactors=TRUE)
View(movie)
```

한글명인 컬럼을 영어로 변환

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

2. 39행의 데이터를 빼서 test_data로 저장

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

3. 훈련 데이터를 가지고 나이브 베이즈 모델을 생성

```
library(e1071)
model2 <- naiveBayes(train_data[, 1:5], train_data$m_type, laplace=0)
```

4. test_data를 예측

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

문제193. 나이가 20대이고 성별이 여자이며 직업이 IT이고 결혼을 하지 않았으며 이성친구가 없는 사람이 선택할 가능성이 높은 영화의 장르를 확인하시오.

1. 워킹 디렉토리에 영화(movie.csv)를 로드한다.

```
movie <- read.csv('movie.csv', header=T, stringsAsFactors=TRUE)
View(movie)
```

한글명인 컬럼을 영어로 변환

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

2. 39행의 데이터를 빼서 test_data로 저장

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

3. 훈련 데이터를 가지고 나이브 베이즈 모델을 생성

```
library(e1071)
model2 <- naiveBayes(train_data[, 1:5], train_data$m_type, laplace=0)
```

4. test_data를 예측

```
test_data3 <- data.frame(age='20대', gender='여', job='IT',
                         marry='NO', friend='NO')
```

```
result3 <- predict(model2, test_data3)
result3
```

문제194. 직업이 학생이고 결혼을 하지 않았으며 이성친구가 없는 20 대 남자가 선호하는 영화 장르를 확인하시오.

1. 워킹 디렉토리에 영화(movie.csv)를 로드한다.

```
movie <- read.csv('movie.csv', header=T, stringsAsFactors=TRUE)
View(movie)
```

한글명인 컬럼을 영어로 변환

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

2. 39행의 데이터를 빼서 test_data로 저장

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

3. 훈련 데이터를 가지고 나이브 베이즈 모델을 생성

```
library(e1071)
model2 <- naiveBayes(train_data[, 1:5], train_data$m_type, laplace=0)
```

4. test_data를 예측

```
test_data4 <- data.frame(age='20대', gender='남', job='학생',
                           marry='NO', friend='NO')
```

```
result4 <- predict(model2, test_data4)
result4
```

나이브 베이즈 머신러닝 실습2 (독감 환자 예측)

문제195. 독감 데이터를 가지고 독감환자인지 아닌지를 판별하는 머신러닝 모델을 생성하시오.

1. 워킹 디렉토리에 독감(flu.csv)을 로드한다.

```
flu <- read.csv('flu.csv', header=T, stringsAsFactors=TRUE)
View(flu)
```

2. test_data 생성

```
flu2 <- flu[-1]
train_data <- flu2[1:7, ]
test_data <- flu2[8, ]
```

3. 훈련 데이터를 가지고 나이브 베이즈 모델을 생성

```
library(e1071)
```

```

model2 <- naiveBayes(train_data[ , 1:4], train_data$flue, laplace=0)

# 4. 예측값 확인
result2 <- predict(model2, test_data[ , 1:4])
result2

# 실제값 확인
test_data[ , 5]

```

문제196. 위의 독감환자여부를 알아내는 모델인 model2를 활용할 수 있도록 R 함수를 생성하시오.

```

naive_func <- function() {
  # 1. 워킹 디렉토리에 독감(flu.csv)을 로드한다.
  flu <- read.csv('flu.csv', header=T, stringsAsFactors=TRUE)

  # 2. 환자 id 제거
  flu2 <- flu[-1]
  train_data <- flu2[1:8, ]

  # 3. 훈련 데이터를 가지고 나이브 베이즈 모델을 생성
  library(e1071)
  model2 <- naiveBayes(train_data[ , 1:4], train_data$flue, laplace=0)

  # 4. 환자 증상 확인
  v_chills <- readline('오한 (Y/N)')
  v_runny <- readline('콧물 (Y/N)')
  v_headache <- readline('두통 (STRONG/MILD/NO)')
  v_fever <- readline('열 (Y/N)')

  test <- data.frame(chills=v_chills, runny_nose=v_runny,
                      headache=v_headache, fever=v_fever)

  result <- predict(model2, test)
  if(result == 'Y') {print('독감 환자 입니다.')}
  else {print('독감 환자가 아닙니다.')}
}

naive_func()

```

문제197. 영화장르를 예측하는 나이브 베이즈 모델을 이용해서 질문을 하고 결과를 출력하는 movie_fun 함수를 생성하시오.

```

movie_fun <- function() {
  # 1. 워킹 디렉토리에 영화(movie.csv)를 로드한다.
  movie <- read.csv('movie.csv', header=T, stringsAsFactors=TRUE)

```

```

# 한글명인 컬럼을 영어로 변환
colnames(movie) <- c('age','gender','job','marry','friend','m_type')

# 2. train_data 생성
train_data <- movie[1:39,]

# 3. 훈련 데이터를 가지고 나이브 베이즈 모델을 생성
library(e1071)
model2 <- naiveBayes(train_data[, 1:5], train_data$m_type, laplace=0)

# 4. 개인정보 확인
v_age <- readline('나이대 (20대/30대/40대)')
v_gender <- readline('성별 (여/남)')
v_job <- readline('직업 (IT/디자이너/무직/언론/영업/자영업/학생/홍보마케팅)')
v_marry <- readline('결혼여부 (YES/NO)')
v_friend <- readline('이성친구여부 (YES/NO)')

test <- data.frame(age=v_age, gender=v_gender, job=v_job,
                     marry=v_marry, friend=v_friend)

result <- predict(model2, test)

print(result)
}

moive_fun()

```

의사결정트리

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

ex)

물고기인지 아닌지 분류하는 의사결정트리 모델을 생성

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

의사결정 나무 기법은 분석의 대상을 분류함수를 활용하여 의사결정 규칙으로 이루어진 나무 모양으로 그리는 기법이다.

의사결정트리 나무를 만드는 방법

- 부모마다의 순수도에 비해서 자식 마디들의 순수도가 증가하도록 자식 마디를 형성해 나

가면서 만든다.

순수도 : 목표변수의 특정범주에 개체들이 포함되어있는 정도를 의미한다.

정보 획득량 확인 방법

```
skin <- read.csv('skin.csv', header=T, stringsAsFactors=T)
View(skin)
```

```
install.packages('FSelector')
library(FSelector)
```

```
wg <- information.gain(cupon_react ~., skin, unit='log2')
print(wg)
```

확인결과 결혼유무가 가장 정보획득량이 높은 것으로 확인된다.

문제198. 지방간을 일으키는 원인중에 가장 큰 영향력을 보이는 요인은 무엇인지 정보획득량을 구해서 알아내시오.

```
fat <- read.csv('fatliver2.csv', header=T, stringsAsFactors=T)
```

```
wg <- information.gain(FATLIVER ~., fat, unit='log2')
print(wg)
```

불순도를 측정하는 척도 3가지

1. 엔트로피 지수 : 열역학에서 쓰는 개념으로 무질서 정도에 대한 측도이고 엔트로피 지수의 값이 클 수록 순수도가 낮다고 볼 수 있다.
2. 카이제곱 통계량 : 데이터의 분포와 사용자가 선택한 기대 또는 가정된 분포 사이의 차이를 나타내는 측정값
3. 지니지수 : 노드의 불순도를 나타내는 값

의사결정트리 실습1

<https://cafe.daum.net/oracleoracle/SZTZ/2037>

1. 의사결정 패키지인 C50 패키지를 설치한다.

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

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 분리
skin2 <- skin[ 1:29, ] # 29개의 데이터로 학습시켜서 의사결정 나무를 생성
nrow(skin2)
```

```

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.
skin_model <- C5.0(skin2_train[, -6], skin2_train$cupon_react )
#           ↑          ↑
#       라벨을 뺀 train 전체 data  train 데이터의 라벨

# 5. 위에서 만든 skin_model 를 이용해서 테스트 데이터의 라벨을 예측하시오!
skin2_result <- predict(skin_model, skin2_test[, -6])
#           ↑
#       라벨을 뺀 테스트 데이터 전체

# 6. 이원 교차표로 결과를 확인하시오 !
# CrossTable(x=실제값, y=예측값)
# 아래의 결과를 보면 전체 9개 중 2개만 맞췄으므로 성능을 높여야 한다.
library(gmodels)
CrossTable(skin2_test[, 6], skin2_result)

```

의사결정나무 알고리즘 4가지와 분리기준

1. CART : 지니지수
2. C4.5, C5.0 : 엔트로피 지수
3. CHAID : 카이제곱 통계량과 F 검정
4. QUEST : 카이제곱 통계량과 F 검정

문제199. 위의 의사결정트리의 성능을 높이시오.

```

# 1. 의사결정 패키지인 c50 패키지를 설치한다.
install.packages("C50")
library(C50)

# 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 분리
skin2 <- skin[1:29, ] # 29개의 데이터로 학습시켜서 의사결정 나무를 생성
nrow(skin2)

```

```

skin2 <- skin2[, -1] # 고객번호를 제외시킨다.
set.seed(20)
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.
skin_model2 <- C5.0(skin2_train[, -6], skin2_train$cupon_react, trials=10)
#           ↑           ↑
#       라벨을 뺀 train 전체 data   train 데이터의 라벨

# 5. 위에서 만든 skin_model 를 이용해서 테스트 데이터의 라벨을 예측하시오!
skin2_result <- predict(skin_model2, skin2_test[, -6])
#           ↑
#       라벨을 뺀 테스트 데이터 전체

# 6. 이원 교차표로 결과를 확인하시오 !
# CrossTable(x=실제값, y=예측값)
# 아래의 결과를 보면 전체 9개 중 2개만 맞췄으므로 성능을 높여야 한다.
library(gmodels)
CrossTable(skin2_test[, 6], skin2_result)

```

knn 하이퍼 파라미터 : k 값, seed 값
naivebayes 하이퍼 파라미터 : laplace 값, seed 값
decision tree 하이퍼 파라미터 : trials 값, seed 값

trials 파라미터로 나무의 갯수를 정한다. 여러개의 나무를 만들어서 그 나무들 중에 가장 분류를 잘하는 나무를 투표를 통해서 선택 한다.

의사결정트리 실습2

아이리스 데이터를 가지고 의사결정트리 실습

```

# 1. party 패키지 설치
install.packages("party")
library(party)

# 2. iris 데이터를 불러온다.
data(iris)

# 3. iris 데이터의 전체 행 수 확인
nrow(iris)

```

```

# 4. iris 데이터의 통계정보 확인
summary(iris)

# 5. iris 데이터의 라벨 컬럼의 종류와 건수를 확인
# 종류가 3가지가 있고 각각 50개씩 구성되어 있다.
table(iris$Species)

# 6. sample 함수를 사용해서 shuffle을 하면서 데이터를 7:3으로 나눈다.
# replace=T 를 사용하여 복원추출한다.
# sample(데이터 수, 행 수, 복원/비복원, 확률)
set.seed(11)
ind <- sample(2,nrow(iris),replace=T,prob=c(0.7,0.3))

# 7. ind == 1 과 ind == 2 를 이용해서 훈련 데이터와 테스트 데이터를 만든다.
traindata <- iris[ind==1,] # 70%에 해당하는 데이터를 traindata로 구성
testdata <- iris[ind==2,] # 30%에 해당하는 데이터를 testdata로 구성

# 8. 의사결정트리 모델을 생성
# 변수 <- 라벨컬럼 ~ 독립변수 컬럼1 + 독립변수 컬럼2 + ...
# myformula <- Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
myformula <- Species ~ .

# party 패키지의 ctree 함수를 이용하여 의사결정 모델을 생성한다.
iris_ctree <- ctree(myformula,data=traindata)

# 9. 모델이 예측한 결과를 확인
predict(iris_ctree)

# 10. 예측결과와 실제 훈련 데이터의 정답을 비교
table(predict(iris_ctree),traindata$Species)

# 11. 모델을 print 하여 질문을 확인한다.
print(iris_ctree)

# 12. iris 데이터의 의사결정트리 모델을 시각화 한다.
# 12.1
plot(iris_ctree)
# 12.2
plot(iris_ctree,type="simple")

# 13. 테스트 데이터 30개에 대한 예측 결과 확인
testpred <- predict(iris_ctree, newdata=testdata)
table(testpred,testdata$Species)

```

문제200. 와인 데이터를 분류하는 의사결정트리 모델을 생성하시오.

데이터 : wine.csv

의사결정트리 패키지 : party 패키지의 ctree 함수

21.01.28

2021년 1월 28일 목요일 오전 9:42

sample함수에 대한 설명

```
# 1. sample 함수로 데이터를 랜덤 추출하는 방법  
# 데이터가 1과 2 두개가 있는 상태에서 복원추출 하는데 3번 실행한다.  
set.seed(1)  
  
sample(c(1,2), size=3, replace=TRUE)
```

문제201. 숫자 1번 부터 10번까지의 숫자들을 주머니속에 넣고 랜덤 추출하는데 9번 추출하고 결과를 출력하시오 (복원추출)

```
set.seed(1)  
  
sample(c(1:10), size=9, replace=TRUE)
```

문제202. 숫자 1번 부터 10번까지의 숫자들을 주머니속에 넣고 랜덤 추출하는데 9번 추출하고 비복원 추출하시오.

```
sample(c(1:10), size=9, replace=FALSE)
```

문제203. 숫자 1번부터 10번까지의 숫자들을 랜덤 추출하는데 10번 추출하고 비복원 추출하시오.

```
sample(c(1:10), size=10, replace=FALSE)
```

replace=FALSE, replace=TRUE 의 차이

```
> sample(c(1:10), size=11, replace=FALSE)  
Error in sample.int(length(x), size, replace, prob) :  
'replace = FALSE' 일때는 모집단보다 큰 샘플을 가질 수 없습니다
```

```
> sample(c(1:10), size=11, replace=TRUE)  
[1] 7 6 9 8 9 7 8 6 10 7 3
```

문제204. 아래의 코드를 해석하시오.

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

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

위의 코드를 prob 옵션 없이 다시 쓰면 아래와 같다.

```
sample(c(1,2), 150, replace=T)
```

prob 옵션을 주고 다시 수행

```
sample(c(1,2), 150, replace=T, prob=c(0.7,0.3))
```

숫자 1과 2를 150번 복원 추출하는데 숫자 1은 70%의 확률로 복원추출하고 숫자2는 30%의 확률로 복원추출한다.

문제205. 아래의 코드로 수행해서 출력된 150개의 숫자 1과 2가 각각 몇개씩 있는지 카운트 하시오.

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

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

```
table(ind)
```

문제206. 위의 1과 2의 비율을 출력하시오.

```
prop.table(table(ind))
```

문제207. 150개의 아이리스 데이터에서 2, 5, 73, 121 번째 행을 출력 하시오.

```
iris[c(2, 5, 73, 121), ]
```

문제208. 아래의 ind 변수에서 숫자 1이 몇 개 있는지 확인하시오.

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

```
ind == 1
```

결과를 보면 1인것은 TRUE이고 아닌 것은 FALSE로 출력 되기 때문에 TRUE 인것만 sum 하면 된다.

```
sum(ind == 1)
```

문제209. 아이리스 데이터에서 위의 150개의 TRUE, FALSE 중에 TRUE

행만 출력되게 하시오.

```
iris[ind == 1,]  
nrow(iris[ind == 1,])
```

문제210. 아이리스 데이터에서 ind가 2인 자리의 행들만 출력하시오.

```
iris[ind == 2,]  
nrow(iris [ind == 2,])
```

문제211. 아이리스 데이터를 검색하는데 Sepal.Length (꽃받침 너비) 가 5.0 이상인 꽃들만 출력하시오.

```
iris[iris$Sepal.Length >= 5.0, ]
```

wine 데이터를 의사결정트리 알고리즘을 이용하여 데이터의 품질(Type) 분류

1. party 패키지를 불러온다.

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

```
library(gmodels) # 이원교차표를 보기 위한 패키지
```

2. 데이터를 불러온다.

```
wine <- read.csv('c:/data/wine.csv')
```

```
View(wine)
```

3. 데이터에 대한 정보를 확인

```
nrow(wine) #178
```

```
ncol(wine) #14
```

```
summary(wine)
```

4. 와인 데이터의 종속변수 컬럼의 종류와 건수를 확인

```
table(wine$type)
```

5. Type 컬럼을 펙터로 만들어줌 (chr 타입)

```
wine$type <- factor(wine$type,  
level=c("t1", "t2", "t3"))
```

6. sample 함수를 사용하여 데이터 분할 및 무작위 정렬(7대3 확률)

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

7. 위에서 만든 ind로 훈련데이터와 테스트 데이터를 만듦

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

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

```

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

# 8. 의사결정트리 모델을 생성
# ctree는 기계에서 의사결정트리 알고리즘으로 데이터를 분류하도록
# 구현된 코드가 있는 R 함수이다.
myformula <- Type ~ .
wine_ctree <- ctree(myformula, data=train_data)

# 9. 모델이 예측한 결과를 확인
predict(wine_ctree)

# 10. 예측 결과와 실제 훈련 데이터의 정답과 비교
table(predict(wine_ctree), train_data$Type)

# 11. 어떠한 질문을 기준으로 나누는지 확인
print(wine_ctree)
print(wine_ctree, type='simple')

# 12. 테스트 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$Type, test_pred)
sum(g2$prop.tbl*diag(3)) # 정확도: 0.8363636
plot(wine_ctree)

```

의사결정트리 실습3

은행 대출 채무를 불 이행할 것 같은 고객

1단계 : 데이터 수집

데이터 설명

- 독일의 한 신용기관에서 얻은 대출 정보가 들어있다. 신용 데이터셋은 1000개의 대출예시와 대출금액과 대출 신청자의 특성을 나타내는 일련의 수치특징과 명목특징을 포함하고 있다.

<https://m.blog.naver.com/PostView.nhn?blogId=swkim4610&logNo=220510602767&proxyReferer=https%3A%2F%2Fwww.google.com%2F>

```

credit <- read.csv('credit.csv', stringsAsFactors = T)
str(credit)

prop.table(table(credit$default))

# default : 대출금 상환 여부 (라벨컬럼)
# checking_balance : 예금계좌

```

```

# saving_balance : 적금계좌
# amount : 대출금액

# amount 컬럼의 데이터를 히스토그램 그래프로 그리시오.
r()

# amount 컬럼에 대한 통계정보를 확인하시오.
# 최소 대출금액은 250마르크, 최대 대출금액은 18424마르크로 구성되어있으며
# 평균이 3271마르크 이다.
summary(credit$amount)

```

2단계 : 데이터 탐색

```

# 데이터가 명목형 데이터인지 수치형 데이터인지 확인
# 수치형 데이터는 Knn, 명목형 데이터는 naivebayes를 사용하면 되는데
# 문자와 숫자가 섞여 있는 데이터는 decesion tree를 사용하여 기계학습시킨다.
str(credit)
View(credit)

# 데이터 shuffle 시키고 9대1로 나눈다. (훈련데이터 : 9, 테스트데이터 : 1)
set.seed(123)
credit_shuffle <- credit[sample(nrow(credit)), ]
nrow(credit_shuffle)

train_num <- 0.9 * 1000
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

```

3단계 : 모델 생성

```

library(C50) # 엔트로피지수를 이용해서 순수도를 구하고 분류하는 패키지
str(credit_train) # 라벨컬럼이 몇번째 컬럼인지 확인
ncol(credit_train) # 17

# 900개의 데이터로 학습한 모델을 생성했다.
credit_model <- C5.0(credit_train[, -17], credit_train[, 17])

```

4단계 : 모델 평가

```

# 위에서 만든 모델을 이용하여 테스트 데이터 100개를 예측한다.
credit_result <- predict(credit_model, credit_test[, -17])
# 머신러닝이 예측한 결과
table(credit_result)

```

```

# 정답 확인
table(credit_test[,17])
# 예측 결과와 실제 정답 비교
table(credit_test[,17], credit_result)

credit_result no yes
no 61 21
yes 11 7

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

# 정확도 확인
sum(x$prop.tbl * diag(2))

```

5단계 : 모델 성능 개선

```

# 의사결정트리의 하이퍼 파라미터를 조정
# 1. trials : 의사결정 나무의 갯수를 결정하는 파라미터 (최대 0 ~ 100)
# 2. seed 값 : shuffle 할때 섞는 규칙을 결정하는 파라미터 (정수형 범위)
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)
sum(x$prop.tbl * diag(2))

```

문제212. 은행 채무 불이행자를 예측하는 머신러닝 모델을 의사결정 트리 알고리즘으로 구현했는데 엔트로피 지수로 순수도를 계산하고 정보획득량을 계산하는 C50 패키지로 모델을 생성했다. 이번에는 party 패키지의 ctree 함수를 이용해서 의사결정 분류모델을 만드시오.

```

# 1. party 패키지를 불러옴
library(party)
library(gmodels)

# 2. 데이터를 불러옴
credit <- read.csv("c:\\\\data\\\\credit.csv", stringsAsFactors=TRUE)
str(credit)

# 3. 데이터에 대한 정보를 확인
nrow(credit) #1000
ncol(credit) #17
summary(credit)

```

```

# 데이터의 결측치가 있는지 반드시 확인
colSums(is.na(credit))

# 4. 은행 데이터의 종속변수 컬럼의 종류와 건수를 확인
table(credit$default)

#5. 데이터를 shuffle 시킵니다.
set.seed(659)
credit_shuffle <- credit[ sample(1000), ]
nrow(credit_shuffle)

#6. 데이터를 9대 1로 나눕니다. (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
myformula <- default ~ .
credit_ctree <- ctree(myformula, data=credit_train)

# 8. 모델이 예측한 결과를 확인
predict(credit_ctree)
length(predict(credit_ctree))

# 9. 예측 결과와 실제 훈련 데이터의 정답과 비교
table(predict(credit_ctree), credit_train$default)
#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)
#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))

#13. 시각화를 합니다.
plot(credit_ctree)
plot(credit_ctree, type='simple')

```

의사결정트리 패키지 2가지

1. C50 : 엔트로피 지수로 정보획득량을 구해서 의사결정 나무를 구성
 - 성능 개선 하이퍼 파라미터 : trials, seed
2. party : 카이제곱검정으로 정보획득량을 구해서 의사결정나무를 구성
 - 성능 개선 하이퍼 파라미터 : seed

C50은 trials를 이용해서 바로 성능개선을 할 수 있는게 장점이고 party는 의사결정트리를 시각화 할 수 있는 장점이 있다.

최적의 seed 값을 루프문으로 돌려서 알아내는 방법

```
library(party)
```

#2.wine을 데이터를 불러온다.

```
wine<-read.csv('wine.csv',stringsAsFactors = T)
```

#3.와인 데이터 라벨 컬럼 종류와 건수를 확인합니다.

```
#summary(wine$type)
```

#4.repeat문 돌려서 seed값찾기

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

#5.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
```

#6.의사결정트리 모델(나무)를 생성한다.

```
myformula <- Type ~ .
```

#7. party패키지의 ctree함수를 이용해서 의사결정모델을 생성한다.

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

#8.예측결과를 확인한다.

```
#predict(wine_ctree)
```

#9.예측결과와 실제 테스트 데이터의 정답과 비교한다.

```
#table(predict(wine_ctree),traindata$type)
```

#12.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)
```

문제213. party 패키지를 이용하여 채무불이행자 예측 알고리즘을 구현한 코드의 seed 값을 repeat코드를 이용해서 장확도 0.86일때 seed 값이 무엇인지 출력되게하시오.

21.02.01

2021년 2월 1일 월요일 오전 9:42

oneR 알고리즘

규칙 기반 알고리즘

1. oneR 알고리즘

- 하나의 사실(조건)만 가지고 간단하게 분류하는 알고리즘
- 하나의 사실만 가지고 분류하다보니 간단하지만 오류가 많다.

2. Riper 알고리즘

- 복수개의 사실(조건)을 가지고 분류하는 알고리즘
- 알고리즘이 데이터를 보고 패턴을 발견한다.(조건을 발견한다.)

규칙기반 알고리즘 실습(oneR)

<https://cafe.daum.net/oracleoracle/SZTZ/2083>

1. 버섯 데이터를 R로 로드한다.

```
mushroom <- read.csv("mushrooms.csv", stringsAsFactors=T)
nrow(mushroom) # 8124
ncol(mushroom) # 23
str(mushroom) # type이 정답라벨 컬럼이며 전부 factor로 변경되어 있다.
```

2. mushroom 데이터를 훈련 데이터와 테스트 데이터로 나눈다.

```
# (훈련 데이터 75%, 테스트 데이터 25%)
set.seed(11) # 하이퍼 파라미터
dim(mushroom) # 8124 23
train_cnt <- round(0.75 * dim(mushroom)[1])
train_index <- sample(1:dim(mushroom)[1], train_cnt, replace=F)
mushroom_train <- mushroom[train_index, ]
mushroom_test <- mushroom[-train_index, ]
```

3. 규칙기반 알고리즘인 oneR을 이용해서 독버섯과 일반버섯을 분류하는

모델을 생성한다.

```
install.packages("OneR")
library(OneR)
# 버섯의 냄새만 가지고 버섯을 분류한다.
model1 <- OneR(type~., data=mushroom_train)
model1
summary(model1)
```

4. 위에서 생성한 모델을 가지고 테스트 데이터로 결과를 확인한다.

```
# -1을 사용하여 테스트 데이터에서 정답컬럼을 제외한다.
result1 <- predict(model1, mushroom_test[ , -1])
library(gmodels)
CrossTable(mushroom_test[ , 1], result1)
```

문제214. 위의 모델의 정확도를 출력하시오.

```
g1 <- CrossTable(mushroom_test[, 1], result1)
print(sum(g1$prop.tbl * diag(2)))
```

독버섯 데이터에서 정보획득량이 가장 높은것이 order(버섯향) 이기 때문에 oneR 알고리즘에서 냄새로 버섯을 식용과 독버섯으로 분류하였다.

ex)

독버섯 데이터의 컬럼들의 정보획득량을 확인하시오.

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

문제215. 위의 결과를 다시 출력하는데 정보획득량이 높은것 부터 출력되게 하시오.

```
library(doBy)
orderBy(~ -attr_importance, w1)
```

규칙기반 알고리즘 실습(Riper)

<https://cafe.daum.net/oracleoracle/SZTZ/2084>

```
install.packages("RWeka")
library(RWeka)
mushroom <- read.csv("mushrooms.csv", stringsAsFactors=T)

set.seed(11)
train_cnt <- round(0.75 * dim(mushroom)[1])
train_index <- sample(1:dim(mushroom)[1], train_cnt, replace=F)
mushroom_train <- mushroom[train_index, ]
mushroom_test <- mushroom[-train_index, ]
```

JRip 함수는 oneR 패키지와는 다르게 여러개의 조건을 가지고 분류한다.
model2 <- JRip(type~, data=mushroom_train)
model2 # p245 참고

```
summary(model2)

# 작은 이원교차표가 하나 보임
result2 <- predict(model2, mushroom_test[, -1])
library(gmodels)
CrossTable(mushroom_test[, 1], result2)
```

문제216. 와인 데이터를 Riper 알고리즘으로 분류하는 머신러닝 모델

을 생성하고 정확도를 확인하시오.

```
wine <- read.csv("wine.csv", stringsAsFactors=T)

set.seed(11)
dim(wine)
train_cnt <- round(0.75 * dim(wine)[1])
train_index <- sample(1:dim(wine)[1], train_cnt, replace=F)
wine_train <- wine[train_index, ]
wine_test <- wine[-train_index, ]

install.packages("OneR")
library(OneR)

model1 <- JRip(Type~., data=wine_train)
summary(model1)
result <- predict(model1, wine_test[, -1])
ct <- CrossTable(wine_test[, 1], result)
print(ct$prop.tb[1]+ct$prop.tb[5]+ct$prop.tb[9])
```

<https://cafe.daum.net/oracleoracle/Sg0x/469> (참고)

문제217. seed값을 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)

##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. iris 데이터를 규칙 기반 알고리즘으로 분류하시오. 가장 적절한 seed 값을 알아내시오.

회귀분석

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

ex)

집값에 가장 영향을 주는 요소가 무엇인지 확인하시오.

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

최소 제곱 추정법(p257)

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

문제219. 어느 실험실에서 10시간, 20시간, 30시간, 40시간마다 물질의 방사능 수치를 측정한 자료가 있을 때 35시간에 물질의 방사능 수치는 무엇인지 확인하시오.

(x 축 : 시간, y 축 : 방사능 수치)

```
func_nuclear <- function(x_num) {  
  x = c(10, 20, 30, 40)  
  y = c(300, 250, 200, 150)  
  a = cov(x,y)/var(x)  
  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)
```

문제220. 탄닌 함유량과 애벌래의 성장간의 실험표를 이용해서 탄닌 함유량이 9 일 때 성장률이 어떻게 되는지 알아내는 함수를 생성하시오.

```

reg <- read.table('regression.txt', header = T)

reg_func <- function(x_num) {
  x = reg$growth
  y = reg$tannin
  a = cov(x,y)/var(x)
  x_mean = mean(x)
  y_mean = mean(y)
  b = y_mean - a * x_mean
  y_hat = b + a * x_num
  print(y_hat)
}

reg_func(9)

```

회귀분석 유형

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

$$y = ax + b$$

이중에서 회귀계수는 a 와 b

회귀계수를 최소제곱법을 사용해서 추정한다.

R 함수 lm을 이용해서 단순회귀분석 실습

탄닌 함유량과 애벌레 성장간의 관계에 대한 회귀식을 도출하기

1. 데이터를 로드한다.

```

reg <- read.table('regression.txt', header = T)
reg

```

2. 데이터를 산포도 그래프로 시각화 한다.

```

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

```

3. 회귀분석을 해서 회귀계수인 기울기와 절편을 구한다.

```

model <- lm(growth ~ tannin, data=reg)
model

```

4. 2번에서 시각화한 산포도 그래프에 회귀직선을 겹쳐서 그린다.

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

5. 그래프 제목을 회귀 직선의 방정식으로 출력되게 한다.

```

model$coefficients[1] # 절편
model$coefficients[2] # 기울기

title(paste('성장률=', model$coefficients[2],
'x탄닌 +', model$coefficients[1]))

```

문제221. 광고비가 매출에 미치는 영향을 조사하기 위한 회귀분석을 하시오.

(고객에게 보고하기 위한 회귀식과 그래프를 시각화 하시오.)

1. 데이터를 로드한다.

```

hg <- read.csv('simple_hg.csv')
hg

```

2. 데이터를 산포도 그래프로 시각화 한다.

```

# plot(y ~ x, data=데이터프레임명)
attach(hg)
plot(input ~ cost, data=hg, pch=21, col='blue', bg='blue')

```

3. 회귀분석을 해서 회귀계수인 기울기와 절편을 구한다.

```

model <- lm(input ~ cost, data=hg)
model

```

4. 2번에서 시각화한 산포도 그래프에 회귀직선을 겹쳐서 그린다.

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

5. 그래프 제목을 회귀 직선의 방정식으로 출력되게 한다.

```
model$coefficients[1] # 절편
```

```
model$coefficients[2] # 기울기
```

```
title(expression(italic(input == 2.18649 %*% cost + 62.92913)))
```

오차 그리는 코드

```

yhat <- predict(model, cost=cost)
join <- function(i)
  lines(c(cost[i],cost[i]),c(input[i],yhat[i]), col="green")
  sapply(1:19,join)

```

회귀분석에서의 용어 2가지

1. 오차 : 모집단에서 실제값이 회귀선과 비교해볼 때 나타나는 차이 (정확치와 관측치와의 차이)
2. 잔차 : 표본에서 나온 관측값이 회귀선과 비교해 볼때 나타나는 차이

문제222. 우주왕복선 챠린저호의 폭파원인을 분석하시오.

(우주왕복선 데이터의 o형링 파손수를 y축으로 두고 온도를 x축으로 하여 단순회귀 분석하시

오.)

distress_ct : o형링 파손수
temperature : 온도
field_check_pressure : 압력
flight_num : 비행기 번호

```
# 1. 데이터를 로드한다.  
ch <- read.csv('challenger.csv')  
ch  
  
# 2. 데이터를 산포도 그래프로 시각화 한다.  
# plot(y ~ x, data=데이터프레임)  
attach(ch)  
plot(distress_ct ~ temperature, data=hg, pch=21, col='blue', bg='blue')  
  
# 3. 회귀분석을 해서 회귀계수인 기울기와 절편을 구한다.  
model <- lm(distress_ct ~ temperature, data=ch)  
model  
  
# 4. 2번에서 시각화한 산포도 그래프에 회귀직선을 겹쳐서 그린다.  
abline(model, col='red')  
  
# 5. 그래프 제목을 회귀 직선의 방정식으로 출력되게 한다.  
model$coefficients[1] # 절편  
model$coefficients[2] # 기울기  
  
title( paste('o형링 파손수=', model$coefficients[2],  
'x 온도 + ', model$coefficients[1]) )
```

다중공선성(variance inflation factor)



- 다중공
선성...

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

현업기준 :

팽창계수(VIF)가 보통 10보다 큰것을 골라내고
엄격하게 하려면 5보다 큰것을 골라내고
느슨하게 하려면 15또는 20으로 주로 골라낸다.

다중공선성 확인 실습

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

```
head(Boston)
```

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

```
model <- lm(medv ~. , data=Boston)
model
vif(model) > 10 # 다중 공선성을 보이는 변수들 확인
```

나이브베이즈 수학식 복습 (빅데이터 기사시험 대비)

문제223. 나이브베이즈 공식을 이용해서 아래의 문제를 해결하시오.

<https://cafe.daum.net/oracleoracle/Sg0x/542>

<https://cafe.daum.net/oracleoracle/SgAy/7>

21.02.02

2021년 2월 2일 화요일 오전 9:40

회귀분석

상관관계 데이터 분석 p260

두 변수 간의 상관관계는 변수들의 관계가 직선에 가깝게 따르는 정도를 나타내는 숫자이다.

상관관계는 -1에서 +1 사이의 범위에 있다.

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

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

문제224. 챌린저호의 온도와 O형링 파손수간의 상관계수를 구하시오.

```
launch <- read.csv('challenger.csv', header=T)  
cor(launch$temperature, launch$distress_ct)  
  
# [1] -0.5111264
```

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 상관관계가 높은 컬럼은 아래의 3개중에 어떤 것인지 확인하시오.

1. 온도
2. 압력
3. 비행기 번호(비행기의 노후화와 연관이 있는 번호)

```
cor(launch)
```

	distress_ct	temperature	field_check_pressure	flight_num
--	-------------	-------------	----------------------	------------

distress_ct	1.0000000	-0.51112639	0.28466627	0.1735779
temperature	-0.5111264	1.00000000	0.03981769	0.2307702
field_check_pressure	0.2846663	0.03981769	1.00000000	0.8399324
flight_num	0.1735779	0.23077017	0.83993237	1.0000000

문제226. 위의 상관관계를 R로 시각화 해서 출력하시오.

p274

```
launch <- read.csv('challenger.csv', header=T)

install.packages('psych')
library(psych)

pairs.panels(launch)
```

상관계수 이론(빅데이터 시험)

1. 피어슨 상관계수

- 등간척도나 비례척도의 데이터에서 두 변수의 공분산의 표준편차의 곱으로 나눈 값이다.
- 두 변수간의 선형관계의 크기를 측정하는 값으로 비선형적인 상관관계를 나타내지 못한다.
- 피어슨 상관계수는 모집단, 표본에 적용할 수 있다.

2. 스피어만의 상관계수

- 스피어만 상관계수는 두 변수간의 비선형적인 관계도 나타낼 수 있는 값이다.
- 두 변수를 모두 순위로 변환시킨 후 두 순위 사이의 스피어만 상관계수를 구한다.

문제277. 삼성전자와 현대자동차 둘중에 코스피 등락율과 더 상관관계가 높은 주식이 어떤건지 알아내시오.

데이터 3가지

1. K_index.csv (코스피 등락율)
2. S_stock.csv (삼성전자)
3. H_stock.csv (현대자동차)

```
k <- read.csv('K_index.csv', header=T, stringsAsFactors=F)
s <- read.csv('S_stock.csv', header=T, stringsAsFactors=F)
h <- read.csv('H_stock.csv', header=T, stringsAsFactors=F)
```

```
cor(na.omit(k$k_rate), na.omit(s$s_rate))
cor(na.omit(k$k_rate), na.omit(h$h_rate))
```

문제228. 코스피 등락율과 삼성 수익률 등락율을 plot 그래프로 그리고 그 그래프에 회귀직선을 그으시오.

(x축 : 코스피 등락률, y축 : 삼성수익 등락률)

```
k <- read.csv('K_index.csv', header=T, stringsAsFactors=F)
s <- read.csv('S_stock.csv', header=T, stringsAsFactors=F)
h <- read.csv('H_stock.csv', header=T, stringsAsFactors=F)

all_data <- merge(merge(k,s), h)
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')
```

문제229. 위의 그래프의 제목에 회귀계수를 사용한 직선의 방정식이 들어가게 하시오.

```
model_s$coefficients[1] # 절편
model_s$coefficients[2] # 기울기

title( paste('삼성등락률=', model_s$coefficients[2],
  'x 코스피등락률 + ', model_s$coefficients[1] ) )
```

다중선형회귀분석 (p262)

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

집값(종속변수) : 평수, 교통, 학군, 범죄율(독립변수)

1. 집값에 영향을 미치는 요소가 위의 여러가지 독립변수들 중에서 어떤것인지 확인하시오.
2. 집값을 예측하기 위한 다중회귀식은 어게 되는지 확인하시오.

ex)

태양열전지를 만드는데 있어서 전력효율을 최대화 할 수 있는 가장 좋은 태양열 전지 재료의 조합이 어떻게 되는지 확인하시오.

전력양 = $a_1x_{재료1} + a_2x_{재료2} + \dots + b$

다중선형 회귀식의 회귀계수를 도출하는 방법 (p264)

다중선형 회귀식을 이해하기 위해 사전에 알아할 내용

1. 전치행렬
2. 단위행렬
3. 역행렬
4. 의사역행렬

1. 전치행렬

- 행과 열을 교환하여 얻은 행렬

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}^T = \begin{pmatrix} a & d & g \\ b & e & h \\ c & f & i \end{pmatrix}$$



[R 분석과 프로그래밍] <http://rfriend.tistory.com>

문제230. 아래의 행렬을 R로 구현하고 아래의 행렬의 전치 행렬을 출력하시오.

```
[,1] [,2]
[1,] 1 2
[2,] 3 4
[3,] 5 6
```

```
a <- matrix(c(1,2,3,4,5,6), nrow=3, ncol=2, byrow=T)
a
```

```
t(a)
```

2. 단위행렬

- 주 대각선의 원소가 모두 1이며 나머지 원소는 모두 0인 정사각 행렬

$$I_4 = U_4 = \text{diag}(1,1,1,1) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

[R 분석과 프로그래밍] <http://rfriend.tistory.com>

문제231. 아래의 단위 행렬을 만드시오.

```
[,1] [,2] [,3]
[1,] 1 0 0
[2,] 0 1 0
[3,] 0 0 1
```

`diag(3)`

문제232. 아래의 행렬의 곱을 R로 구현하시오.

[,1] [,2] [,3]	[,1] [,2] [,3]
[1,] 1 2 3	[1,] 1 0 0
[2,] 4 5 6	x [2,] 0 1 0
[3,] 7 8 9	[3,] 0 0 1

```
a <- matrix(c(1,2,3,4,5,6,7,8,9), nrow=3, ncol=3, byrow=T)
b <- diag(3)
a%*%b
```

```
[,1] [,2] [,3]
[1,] 1 2 3
[2,] 4 5 6
[3,] 7 8 9
```

문제233. 아래의 행렬의 곱의 결과를 확인하시오.

[,1] [,2]	[,1] [,2]
[1,] 1 2 x [1,] 1 0	
[2,] 3 4	[2,] 0 1

```
a <- matrix(c(1,2,3,4), nrow=2, ncol=2, byrow=T)
b <- diag(2)
a%*%b
```

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

```
[1,] 1 2  
[2,] 3 4
```

3. 역행렬

- 원래 행렬과 곱했을 때 단위행렬이 되는 행렬

When A is an n by n square matrix,

$$A B = B A = I_n$$

B is called the **inverse** of A, denoted by A^{-1}

[R 분석과 프로그래밍] <http://rfriend.tistory.com>

아래에 예를 하나 들어봤습니다.

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$



$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} -2 & 1 \\ 1.5 & -0.5 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

[R 분석과 프로그래밍] <http://rfriend.tistory.com>

문제234. 아래의 a 행렬의 역행렬을 R로 구하시오.

```
[,1] [,2]  
[1,] 1 2  
[2,] 3 4
```

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

```
solve(a)
```

```
[,1] [,2]  
[1,] -2.0 1.0  
[2,] 1.5 -0.5
```

a 와 a의 역행렬을 행렬곱하면 단위행렬이 출력된다.

```
round(a %*% solve(a))
```

```
[,1] [,2]  
[1,] 1 0  
[2,] 0 1
```

4. 의사 역행렬(Pseudo inverse)

- 기본적으로 역행렬은 정방행렬일 때만 구할 수 있다.
(직사각형 행렬이면 역행렬을 구할 수 없다.)
직사각형 행렬의 역행렬을 구하려면 의사역행렬을 구해야한다.

4.1 정방행렬일 때

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)
```

4.2 정방행렬이 아닐 때

문제236. 아래의 a행렬을 만들고 a행렬의 전치행렬을 곱하면 정방행렬이 되는지 확인하시오.

```
[,1] [,2] [,3] [,4]  
[1,] 1 2 3 4  
[2,] 5 6 7 8  
[3,] 9 10 11 12
```

```
a <- matrix(c(1:12), nrow=3, ncol=4, byrow=T)  
a%*%t(a)
```

```
[,1] [,2] [,3]  
[1,] 30 70 110  
[2,] 70 174 278  
[3,] 110 278 446
```

문제237. p264에 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 번호인지 알아내시오.

```
launch <- read.csv('challenger.csv', header=T)
launch
reg(y=launch$distress_ct, x=launch[ ,2:4])
```

문제239. 스마트폰 만족도(종속변수)에 영향을 미치는 요소중 가장 영향력이 있는 독립변수는 무엇인지 확인하시오.

1.

```
m <- read.csv('multi_hg.csv', header=T)
reg(y=m$만족감,x=m[ , 1:3])
```

2.

```
m <- read.csv('multi_hg.csv', header=T)
attach(m)
lm(만족감 ~ 외관 + 편의성 + 유용성, data=m)
```

회귀분석에서도 정규화를 해야하는지 확인하시오.

보험회사에서 미국 국민의 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 안해야 하는 경우
 - 나이가 한 살 늘어날때 보험료가 얼마나 인상되어야 하는지 예측해야 할 때
 - 부양가족이 한명 더 늘어날 때마다 보험료가 얼마나 인상되어야 하는지 예측해야 할 때

표준화 정규화의 차이

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

ex)

scale 함수

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

ex)

min/max 함수

머신러닝에서는 표준화보다 정규화가 더 좋은 결과가 나와서 정규화를 더 선호한다.

문제240. 미국 대학교 입학에 가장 크게 영향을 미치는 요소가 무엇인지 출력하시오.

(정규화 하지 않고 수행)

academic : 학과점수

sports : 체육점수

music : 음악점수

acceptance : 입학기준점수(종속변수)

```
sports <- read.csv('sports.csv', header=T)
```

```
summary(sports)
```

```
reg(y=sports$acceptance, x=sports[, 2:4])
```

estimate

Intercept 11.4902799

academic 0.1557737

sports 0.5726859

music 0.1046008

정규화를 하지 않고 영향도를 확인하면 sports가 가장 높게 나온다.

문제241. min/max 정규화를 하고 영향도를 확인하시오.

```
sports <- read.csv('sports.csv', header=T)
```

```
sports <- sports[, c(2:5)] # 학생번호 제외
```

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

```
sports_n <- as.data.frame(lapply(sports, normalize)) # 정규화
```

```
summary(sports_n)
```

```
reg(y=sports_n$acceptance, x=sports_n[, 1:3])
```

estimate

Intercept 0.06121748

academic 0.48963854

sports 0.30194528

music 0.11432339

다중 회귀분석 실습(미국 국민의 의료비 데이터 예측) p268

<http://cafe.daum.net/oracleoracle/SZTZ/2156>

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 선능 개선

1. 보험 데이터를 R로 로드한다.

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

# 결측치가 있는지 확인
colSums(is.na(insurance))

# 종속변수의 데이터가 정규성을 띠는지 히스토그램으로 확인
hist(insurance$expenses)
```

2. 정규화 작업을 수행한다.

```
# 종속변수에 어떤 독립변수의 영향이 큰지 확인하기 위해서 정규화 작업을 수행
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)
```

3. 독립변수와 종속변수간의 상관관계 분석

```
# 회귀분석하기 전에 상관관계를 확인한다.
# (독립변수들간의 강한 상관 관계를 보이는 다중공선성 여부를 확인해야
# 회귀분석 결과에 가장 중요한 결정계수에 결과를 신임할 수 있다.)
cor( insurance[ , c("age","bmi","children","expenses")] )
```

상관관계 시각화

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

3. 회귀함수인 lm을 이용해서 독립변수들의 회귀모수를 확인한다 (기울기)

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

또는

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

4. 모델 성능 평가

```
# 회귀분석에서 회귀모델 성능평가 : 결정계수
model2 <- lm(expenses~., data=insurance)
summary(model2) # Multiple R-squared: 0.7509
```

```

# 5. 모델 성능 개선
# 하이퍼 파라미터 조절
# 1. knn : k값 / 2. naivebayes : laplace값 / 3. decision tree : trials값
# 회귀분석에서 성능개선 : 파생변수 추가
insurance$age2 <- insurance$age^2
head(insurance)

model3 <- lm(expenses~, data=insurance)
summary(model3) # Multiple R-squared: 0.7537

# 5-1
insurance$bmi2 <- ifelse(insurance$bmi >= 30, 1, 0)
head(insurance)

model4 <- lm(expenses~, data=insurance)
summary(model4) # Multiple R-squared: 0.7582

# 5-2
model5 <- lm(expenses ~ age + age2 + children + bmi + sex + bmi2*smoker +
region, data=insurance)
summary(model5) # Multiple R-squared: 0.8664

```

데이터 소개 : 미국 환자의 가상 의료비가 들어있는 모의 데이터셋이다.
이 데이터는 미국 통계국의 인구 통계를 이용해 생성되었으며 대개 실제 질병을 반영한다.
현재 의료보험에 등록된 1,338명의 수익자 예시가 들어있으며 각 예시는 환자의 특성과 해당
연도에 의료보험에 청구된 전체 의료비를 나타내는 특징으로 구성되어 있다.

age : 나이
sex : 성별
bmi : 체질량 비만 지수
children : 부양가족수
smoker : 흡연여부
region : 사는 지역(북동, 남동, 북서, 남서)
expenses

문제242. 나이가 많을 수록 의료비가 많이 든다는것을 확인했는데 나
이 데이터를 더 크게 만들어서 파생변수로 생성하면 결정계수가 더
올라가는지 확인하시오.

```

insurance$age2 <- insurance$age^2
head(insurance)

model3 <- lm(expenses~, data=insurance)
summary(model3) # Multiple R-squared: 0.7537

```

문제243. 비만인 사람(bmi가 30 이상)이 의료비가 더 많이 들거라 예상하고 insurance 테이블에 비만인 사람과 비만이 아닌 사람에 대한 컬럼을 추가해서 결정계수가 더 올라가는지 확인하시오.

```
insurance$bmi2 <- ifelse(insurance$bmi >= 30, 1, 0)  
head(insurance)
```

```
model4 <- lm(expenses ~ ., data=insurance)  
summary(model4) # 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 확인하시오.

```
model5 <- lm(expenses ~ age + age2 + children + bmi + sex + bmi2*smoker +  
region, data=insurance)  
summary(model5) # Multiple R-squared: 0.8664
```

문제245. 현재 결정계수가 0.8664인데 이 결정계수를 더 올릴 수 있는 파생변수를 생각해서 모델을 생성하시오.

21.02.03

2021년 2월 3일 수요일 오전 10:05

다중 공선성 실험

```
# 1. 다중공선성의 VIF(팽창계수)를 확인할 수 있는 패키지를 설치한다.  
# 다중 공선성을 보이는 변수들은 팽창계수가 10이상인 변수들이다.  
install.packages('car')  
library(car)  
  
# 2. 데이터 로드  
test <- read.csv('test_vif1.csv')  
View(test) # 시험점수(종속변수)/아이큐,공부시간(독립변수)  
  
# 3. 독립변수들끼리의 상관관계를 확인  
cor(test[, c('아이큐','공부시간')])  
  
# 4. 회귀분석 모델을 생성  
test <- test[, -1] # 학생번호 제외  
m1 <- lm(test$시험점수~., data=test)  
summary(m1)  
  
# 5. 다중 공선성을 보이는지 확인한다.  
vif(m1)
```

위의 예제는 팽창계수가 10보다 크지 않으므로 다중공선성을 보이는 변수는 없다.

문제246. test_vif2.csv를 로드하여 다중회귀 분석을 하고 다중공선성을 보이는 독립변수들이 있는지 확인하시오.

```
# 1. 다중공선성의 VIF(팽창계수)를 확인할 수 있는 패키지를 설치한다.  
# 다중 공선성을 보이는 변수들은 팽창계수가 10이상인 변수들이다.  
install.packages('car')  
library(car)  
  
# 2. 데이터 로드  
test2 <- read.csv('test_vif2.csv')  
View(test2) # 시험점수(종속변수)/아이큐,공부시간,등급평균(독립변수)  
  
# 3. 독립변수들끼리의 상관관계를 확인  
cor(test2[, c('아이큐','공부시간','등급평균')])
```

```

# 4.회귀분석 모델을 생성
test2 <- test2[, -1] # 학생번호 제외
m2 <- lm(test2$시험점수~, data=test2)
summary(m2)

# 5.다중 공선성을 보이는지 확인한다.
vif(m2)

# 6.아이큐와 등급평균이 강한 상관관계를 보여서 회귀분석 결과에 좋지않은
# 결과를 보이므로 둘 중에 하나는 제외하고 회귀분석을 해야한다.
# 결정계수가 높은것을 선택한다.

m3 <- lm(test2$시험점수~, data=test2[, c(1,2,3)])
summary(m3) # Multiple R-squared: 0.9053

m4 <- lm(test2$시험점수~, data=test2[, c(1,3,4)])
summary(m4) # Multiple R-squared: 0.9154

```

문제247. 미국 국민의 의료비 데이터로 의료비를 예측하는 회귀모델을 생성하는데 이 회귀모델을 이용해서 의료비를 쉽게 예측할 수 있는 IBM의 웃슨과 같은 질문을 하는 코드를 구현하시오.

```

reg_func <- function(){

  x1=as.integer( readline('나이가 어떻게 되십니까? '))
  x2=as.integer( readline('성별이 어떻게 되십니까? (여자:1,남자:0)'))
  x3=as.integer( readline('비만지수가 어떻게 되십니까? (16~53)' ))
  x4= as.integer( readline('부양가족수가 몇명입니까? (0~5)' ))
  smoke=as.character( readline('흡연을 하십니까?(yes/no)' ))
  region=as.character(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 ( paste( 'ai 가 예측한 의료비는 ', y , ' 입니다 ') )

}

```

reg_func()

회귀 트리 p289

1. 회귀트리

- 수치를 예측하는 트리(tree)

<https://cafe.daum.net/oracleoracle/SeFi/375>

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.

회귀분석의 경우에는 독립변수의 갯수가 많으면 독립변수들간의 상관관계를 유심히 조사해야 한다. 왜냐하면 다중공선성 문제가 생기면 회귀분석결과에 신뢰도가 떨어지기 때문이다.

회귀트리에서 사용하는 수학식 : SDR(표준편차축소)를 사용

문제248. 아래의 책 292페이지의 그림인 원본 데이터를 A, B 중 어느 속성으로 나누는게 더 나은지 SDR을 구해서 알아내시오.

원본 데이터	1 1 1 2 2 3 4 5 5 6 6 7 7 7 7
속성 A로 나눔	1 1 1 2 2 3 4 5 5 6 6 7 7 7 7 ➤7
속성 B로 나눔	1 1 1 2 2 3 4 5 5 6 6 7 7 7 7

1.원본 데이터를 만든다.

```
tee <- c(1,1,1,2,2,3,4,5,5,6,6,7,7,7,7)
```

2.원본 데이터를 A속성으로 나눈 데이터

```
at1 <- c(1,1,1,2,2,3,4,5,5)  
at2 <- c(6,6,7,7,7,7)
```

3.원본 데이터를 B속성으로 나눈 데이터

```
bt1 <- c(1,1,1,2,2,3,4)  
bt2 <- c(5,5,6,6,7,7,7,7)
```

4.A속성으로 나누었을 때 SDR(표준편차 축소값)

```
sdr_a <- sd(tee) - (length(at1) / length(tee) * sd(at1) +  
length(at2) / length(tee) * sd(at2))
```

```
sdr_a # 1.202815
```

```
# 5.B속성으로 나누었을 때 SDR  
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 분류
```

```
# 7.B속성으로 분류한 원본 데이터의 두 영역의 평균값을 각각 구해서 등급예측  
bt1 <- c(1,1,1,2,2,3,4)  
bt2 <- c(5,5,6,6,7,7,7,7)
```

```
mean(bt1) # 2  
mean(bt2) # 6.25
```

회귀트리로 와인품질 측정 분류기 생성

```
# 1. 데이터를 로드한다.
```

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

```
#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
```

```
# 2. 와인 데이터의 종속변수인 quality 데이터가 정규분포에 속하는
```

```
# 안정적인 데이터인지 확인
```

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

```
# 3. wine 데이터를 train 데이터와 test 데이터로 나눈다.
```

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

```
# 4. train 데이터를 가지고 model 을 생성한다.
```

```
install.packages('rpart') # 회귀트리 모델 구현할 수 있는 rpart 설치  
library(rpart)
```

```

model <- rpart( quality ~ . , data=wine_train)
model

...
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 *

```

* 표시가 있는 노드는 잎노드로 노드에서 예측이 이루어진다는 것을 의미한다.

와인 데이터의 예측 등급이다.

...

5. 위에서 나온 모델로 트리를 시각화 하시오 !

```

# digits = 3은 소수점 세번째까지 허용
install.packages('rpart.plot')
library(rpart.plot)
rpart.plot( model, digits=3)
rpart.plot(model, digits=3, fallen.leaves=T, type=3, extra=101)

```

6. 위에서 만든 모델로 테스트 데이터의 라벨을 예측하시오 !

```
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 상관관계 확인한다.

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

9. 두 데이터간의 오차율을 확인

상관관계는 1에 가까울수록 좋은 것이고 오차는 0에 가까울 수록 좋은 모델

```

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

```

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

모델트리

기존 회귀트리 모델 + 다중회귀를 추가한 모델

회귀트리는 무조건 분할한 y(종속변수의 값들) 값들의 평균값으로만 예측을 했는데

모델트리는 분할한 x값과 y값들에 대한 회귀식을 통해서 y값을 예측한다.

모델트리로 와인 품질 측정 분류기 생성

1. 회귀트리 실습의 1번 부터 3번 실습 까지 반복합니다

1.1. 데이터를 로드한다.

```
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)
```

```
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 )
```

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

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

다중 회귀분석에서는 미국 국민 의료비 데이터로 미국 국민의 의료비를 예측하는 회귀 모델을

생성했고 회귀 모델의 성능을 높이기 위해서 파생변수를 생성했다.
회귀트리와 모델트리는 분류를 하는데 회귀의 개념이 섞인 분류이다.

머신러닝의 종류 3가지

1. 지도학습
 - a. 분류 (수치 분류)
 - knn, naivebayes, decision tree, oneR, riper, regression tree, model tree, 신경망
 - b. 회귀 (수치 예측)
 - 단순회귀분석, 다중회귀분석
2. 비지도학습
3. 강화학습

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.

<https://cafe.daum.net/oracleoracle/SZTZ/2160>

1. 데이터를 로드한다.

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

2. 본래 데이터의 최소값, 최대값 비교

```
summary(boston$MEDV)
```

3. 훈련과 테스트 데이터 생성

```
boston_train <- boston[1:495, ] # 495
```

```
boston_test <- boston[496:506, ] # 11
```

```
str(boston_train)
```

4. 회귀트리 모델을 생성한다.

```
model <- rpart( MEDV ~ ., data=boston_train)
```

```
model
```

5. 생성된 모델과 테스트 데이터로 예측한다.

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

6. 결과와 실제 테스트 라벨과의 상관정도를 확인한다.

```
cor(result, boston_test$MEDV)
```

7. 결과와 실제 테스트 라벨과의 평균절대오차를 확인한다.

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

```
MAE( result, boston_test$MEDV)
```

8. 이번에는 보스톤 하우징 데이터를 모델트리로 구현해서 성능을 높여본다.

```
library(RWeka)
m.m5p <- M5P(MEDV ~ ., data=boston_train)
p.m5p <- predict(m.m5p, boston_test)
cor(p.m5p, boston_test$MEDV)
MAE(boston_test$MEDV, p.m5p)
```

오라클에서 바로 SQL로 회귀분석 구하는 방법

<http://cafe.daum.net/oracleoracle/SZTZ/2082>

문제250. SQL로 회귀분석 구현한 스크립트와 CSV파일을 답글로 올리시오.



오라클로
머신러닝...

21.02.04

2021년 2월 4일 목요일 오전 9:42

회귀분석으로 분석한 결과물을 케글에 올리고 자신의 점수 확인하는 방법

1. 훈련 데이터와 테스트를 나누고 훈련 데이터로 회귀모델을 생성
(이미 케글에서 훈련데이터와 테스트와 테스트 데이터를 나눠서 제공)
2. 정답(집값)이 있는 훈련 데이터로 회귀모델을 생성
3. 만든 모델로 정답이 없는 테스트 데이터의 집값을 예측
4. 테스트 데이터의 정답과 내가 만든 예측값과의 오차가 0에 가까울 수록 잘만든 회귀모델이 된다.

오차를 확인하기 위해서는 케글에 결과물을 올려서 확인해야 한다.

<https://www.kaggle.com/c/titanic> (분류)

<https://www.kaggle.com/c/house-prices-advanced-regression-techniques> (회귀)

신경망

머신러닝의 종류 3가지

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

인공신경망을 만들어서 분류작업을 수행

인공신경망을 만들면서부터 체노동을 대신했던 기계(컴퓨터)에게 지능을 부여하기 시작
인공지능 신경망의 시초가 된 것 퍼셉트론이다.

문제250. AND 퍼셉트론의 최종 가중치가 어떻게 되는지 손으로 풀어서 알아내시오.

퍼셉트론 구현(R 코드)

문제251. 아래의 행렬을 만들고 inputs1이라는 변수에 행렬을 입력하시오.

```
[,1] [,2]
[1,] 0 0
[2,] 1 0
[3,] 0 1
[4,] 1 1
```

```
inputs1 <- matrix(c(0,0,1,0,0,1,1,1), nrow=4, ncol=2, byrow=T)
```

```
inputs1
```

문제252. targets1이라는 변수에 아래의 행을 입력하시오.

```
[,1]
[1,] 0
[2,] 0
[3,] 0
[4,] 1
```

```
targets1 <- matrix(c(0,0,0,1), nrow=4, ncol=1, byrow=T)
```

```
targets1
```

문제253. w0, w1, w2에 해당하는 가중치 행렬을 3행 1열로 만들어서 w라는 변수에 저장하시오.

```
w <- matrix(c(0.3,0.4,0.1), nrow=3)
```

```
w
```

문제254. -1의 값을 4행 1열로 만든 행렬을 만들고 x0이라는 변수에 넣으시오.

```
x0 <- matrix(c(-1,-1,-1,-1), nrow=4)
```

```
x0
```

문제255. x0 행렬과 inputs1 행렬을 cbind로 묶어서 아래의 행렬을 new_input에 넣으시오.

```
new_input <- cbind(x0,inputs1)
```

```
new_input
```

문제256. new_input 행렬과 w 행렬과의 곱을 구하시오.

```
k <- new_input %*% w
```

```
k
```

문제257. 아래의 계단함수(활성화 함수)를 생성하시오.

```
step <- function(x) {  
  ifelse(x > 0, 1, 0)  
}
```

```
step(0.3)
```

```
step(-0.2)
```

문제258. 위에서 생성한 k값을 step함수에 넣고 targets1과의 차이를 구하시오.

```
tkf <- targets1 - step(k)
```

```
tkf
```

문제259. for loop 문을 이용해서 1부터 4까지의 숫자가 출력되게 하시오.

```
for (i in 1:4) {  
  cat('\n', i)  
}
```

문제260. 위에서 만든 오차(tkf변수)를 이용해서 가중치(w변수)를 갱신하는 아래의 식을 구현하시오.

```
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)  
}
```

활성화 함수의 종류 (p318)

인공신경망의 기초가 된 알고리즘은 인공신경세포 하나를 컴퓨터로 구현한 퍼셉트론이다. ↓ 셉트론에서 주로 사용되는 활성화함수는 아래와 같다.

인공신경망의 학습 목표는 학습이 잘된 가중치를 생성하는 것이다.

1. 계단함수 : 입력신호의 총합이 임계치 초과여부에 따라 숫자 1과 0으로 리턴
 - $f(0.3) = 1, f(-0.2) = 0$
2. 시그모이드 함수 : 0~1 사이의 연속적인 실수값을 리턴
 - 단층 : 입력층 -> 출력층
 - 다층 : 입력층 -> 은닉층 -> 출력층
 - 신경망을 다층 신경망으로 사용하려면 활성화 함수를 시그모이드(sigmoid) 함수를 사용해야 한다.
 - 시그모이드 단점 : 기울기 소실문제가 발생한다.
3. 렐루 함수(rectified linear unit)
 - 시그모이드 함수의 단점때문에 나온 함수이다.
시그모이드 함수의 단점이 전파가 역전파 될때 기울기 소실로 인해서 전파가 앞층까지 안 된다는 단점이 있어서 나오게 된 함수이다.
4. leaky relu 함수 : 렐루 함수의 음수 부분의 기울기가 0이여서 역전파할때 기울기가 소실되므로 기울기를 0이 아니게 만들어준 함수
5. tanh 함수

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) # -5에서 5까지 0.01 간격으로 숫자 출력
```

x

```
plot(x, step(x), col='blue', type='o')
```

문제262. R로 시그모이드 함수를 생성하시오.

```
sigmoid <- function(x) {  
  1 / (1 + exp(-x))  
}
```

```
sigmoid(3)
```

문제263. 위의 계단함수처럼 시그모이드 함수를 그래프로 그리시오.

```
x <- seq(-5, 5, 0.01)
```

x

```
plot(x, sigmoid(x), col='blue')
```

문제264. 렐로 함수를 생성하시오.

```
relu <- function(x) {  
  ifelse(x>0, x, 0)  
}
```

```
relu(4) # 4  
relu(-3) # 0
```

문제265. 렐루 함수의 그래프를 그리시오.

```
x <- seq(-5, 5, 0.01)
```

x

```
plot(x, relu(x), col='blue')
```

신경망 실습1 (콘크리트 강도를 예하는 인공신경망 만들기)

<https://cafe.daum.net/oracleoracle/SZTZ/2173>

콘크리트의 강도를 예측하는 신경망을 만드는 실습

#자갈, 모래, 시멘트등을 몇대 몇 비율로 섞었을때 어느정도 강도가 나오는지

#예측하는 신경망

```

# 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))

summary(concrete_norm)

# 4. 0~1사이로 데이터가 잘 변경되었는지 확인
summary(concrete_norm$strength)

# 본래 데이터의 최소값, 최대값과 비교
summary(concrete$strength)

# 결측치 확인
colSums(is.na(concrete))

# 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)))
}

```

```

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

for (i in 3:length(colnames(concrete))){

  a = grubbs.flag(concrete[,colnames(concrete)[i]])
  b = a[a$Outlier==TRUE,"Outlier"]
  print ( paste( colnames(concrete)[i] , '--> ', length(b) ) )

}

# 5. 훈련 데이터, 테스트 데이터를 나눈다 (8:2)
concrete_train <- concrete_norm[1:773, ] # 773
concrete_test <- concrete_norm[774:1030, ] # 257

nrow(concrete_train)
nrow(concrete_test)

# 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)
# 0.806285848

# 12. 모델의 성능 개선
concrete_model2 <- neuralnet(formula=strength ~ cement + slag + ash +
                           water +superplastic + coarseagg +
                           fineagg + age, data =concrete_train ,
                           hidden=c(5,2) )

# hidden= c(5, 2)
#      ↑ ↑
# 은닉1층 5개 은닉2층 2개

plot(concrete_model2)

# 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)  
# 0.9335748229
```

이상치 확인 코드

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

문제266. 위에서 만든 신경망보다 더 성능이 좋은 콘크리트 강도를 예측하는 신경망을 생성하시오.

```
concrete_model2 <- neuralnet(formula=strength ~ cement + slag + ash +  
  water +superplastic + coarseagg +  
  fineagg + age, data =concrete_train ,  
  hidden=c(10,5) )
```

문제267. 보스톤 지역 집값을 예측하는 신경망을 만들고 케글에 올려서 자신의 스코어를 캡처해서 올리시오.

<https://www.kaggle.com/c/boston-housing/submissions>

```
# 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      5개의 보스턴 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 특인의 비율을 말함.  
# [13] LSTAT   모집단의 하위계층의 비율(%)  
# [14] MEDV    본인 소유의 주택가격(중앙값) (단위: $1,000)
```

1. 데이터 수집

```
getwd()
```

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

```
head(boston_train)  
head(boston_test)
```

```
str(boston_train)
```

2. 정규화 작업을 수행

```
# min/max 함수 : 0 ~ 1 사이의 데이터로 재구성하는 함수  
# scale 함수 : 평균 0이고 표준편차 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)
```

3. 신경망 모델 생성

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

```
# learningrate =0.02 (학습률)  
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. 모델 평가
model_results <- compute(boston_model, boston_test_norm)

predicted_medv <- model_results$net.result
predicted_medv

# 캐글에 테스트 데이터에 대한 예측값을 올릴때는 정규화 된 값을 올리면
# 안되고 실제 데이터로 올려줘야 한다.

# 5. 데이터를 역정규화 시킨다.

# 5.1 정규화 된 테스트 데이터에 예측한 집값을 붙여서 데이터 프레임으로 구성
boston_test_norm2 <- data.frame(boston_test_norm, "MEDV"=predicted_medv)

# 5.2 역정규화 작업을 수행
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 제출 형식으로 csv 파일 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="c:/data/boston_test9.csv",row.names = F)

# 7. 캐글에 결과 올리기

```

21.02.05

2021년 2월 5일 금요일 오전 9:43

정규화된 데이터를 역정규화하기

<https://cafe.daum.net/oracleoracle/Sg0x/653>

```
# 정규화된 데이터를 역정규화하는 실습
```

```
# 1. 테스트 데이터를 만든다.
```

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

```
# 2. 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. 정규화된 데이터를 다시 역정규화 한다.
```

```
# 3.1 정규화 하기전 데이터에서 최소 벡터와 최대벡터를 찾는다.
```

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

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

```
# Map(역정규화시키는 함수명, 정규화된 데이터 프레임명,
```

```
# 정규화 시키기 전 최소벡터, 정규화 시키기 전 최대벡터)
```

```
as.data.frame(Map(denormalize,ddnorm,minvec,maxvec))
```

문제268. 하이퍼 파라미터를 조정해서 오차가 더 떨어지도록 하시오.

```
# 1. 데이터 수집
```

```
getwd()
```

```
# 2. 정규화 작업을 수행
```

```
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)

# 3. 신경망 모델 생성
#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 출력
model_results <- compute(boston_model, boston_test_norm)

pred_medv <- model_results$net.result
pred_medv

# 5. 테스트 데이터의 집값만 역정규화 시킨다.
denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data$medv)
pred_medv_un <- denormalize(pred_medv)

# 6. 제출 포맷으로 변경
sample <- cbind(boston_test$ID, pred_medv_un)
head(sample)

colnames(sample) <- c("id", "medv")

write.csv(sample, "Submission_sample.csv", row.names=FALSE)

```

정규화된 테스트 데이터의 예측 집값을 역정규화 시킬 때 훈련 데이터의 집값중 최대값인 50 과 최소값인 5를 이용해서 집값만 역정규화 시킨다.

문제269. 보스톤 지역 집값을 예측하는 신경망을 만들고 케글에 올려서 자신의 스코어를 캡처해서 올리시오.

파생변수를 추가해서 머신러닝 모델의 성능 높이는 방법

(보스톤 하우징)

- 집값에 영향을 주는 독립변수가 무인지 확인 (상관관계로 확인)
 1. pairs.panel() 함수 이용 (p275)

```
library(psych)
pairs.panels(boston_train[, -1])
```

1.2 corrplot 패키지를 이용해서 상관관계를 확인

```
install.packages('corrplot')
library(corrplot)

boston_cor <- cor(boston_train[, -1])
boston_cor

corrplot(boston_cor, method = 'circle')
corrplot(boston_cor, method = 'number')

# 남색과 빨간색으로 시각화 되는데 남색에 가까우면 양의 상관관계가 높고
# 빨간색으로 진해지면 음의 상관관계가 높다.
# 원의 크기가 클수록 상관관계가 크다.
```

2. 히스토그램 그래프를 그려서 데이터 분포를 확인

2.1 종속변수가 정규성을 띠는지 확인

```
hist(boston_train$medv)
```

2.2 방의 개수(rm)이 정규성을 띠는지 확인

```
hist(boston_train$rm)
```

2.3 하위계층의 비율(lstat)이 정규성을 띠는지 확인

```
hist(boston_train$lstat)
```

2.4 결측치가 있는지 확인

```
colSums(is.na(boston_train))
colSums(is.na(boston_test))
```

2.5 이상치가 있는지 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)))
}
```

```
train <- 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(train[, colnames(train)[i]])
```

```
b = a[a$Outlier==TRUE,"Outlier"]
print ( paste( colnames(train)[i] , '-->' , length(b) ) )
}
```

2.6 결측치가 있다면 다른 값으로 치환해야 한다.

(이상치가 터무니없이 큰 값이라면 그 값을 포함한 행을 지운다.)

- 결측치가 있는 컬럼의 다른 데이터 평균값
 - 결측치가 있는 컬럼의 다른 데이터 최빈값
 - 결측치가 있는 컬럼의 다른 데이터 회귀식의 y 값

3. 파생변수 생성

3.1 age와 medv와의 plot 그래프를 확인

```
attach(boston_train)  
plot(age, medv, col='blue')  
cor(age, medv)
```

3.2 dis와 medv와의 plot 그래프를 확인

```
plot(dis, medv, col='blue')  
cor(dis, medv)
```

3.3 아래의 조건을 해당하는 건수 확인

```
range(age)  
hist(age)  
range(dis)  
hist(dis)
```

```
boston_train[age < 20 & dis >= 5, ]  
mean(boston_train[age < 30 & dis >= 5, 'medv' ])
```

3.4 age < 30 이면서 dis >= 5 인 데이터는 1로 하고 아니면 0으로 하는 파생변수를 생성

```
boston_train$age_dis <- ifelse(age < 30 & dis >= 5, 1, 0)  
table(boston_train$age_dis)
```

```
boston_test$age_dis <- ifelse(boston_test$age < 30 &  
                                boston_test$dis >= 5, 1, 0)  
table(boston_test$age_dis)
```

4. 파생변수를 추가한 데이터를 학습시켜 모델 재생성

1. 데이터 수집

`getwd()`

2. 정규화 작업을 수행

```
normalize <- function(x) {
```

```
return ( (x-min(x)) / (max(x) - min(x) ) )
```

1

```

table(boston_train$age_dis)

boston_test$age_dis <- ifelse(boston_test$age < 30 &
                                boston_test$dis >= 5, 1, 0)
table(boston_test$age_dis)

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)

# 3. 신경망 모델 생성
#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(2,5),
                           learningrate =0.02)

# 4. 테스트 데이터에 대한 예측값을 출력
model_results <- compute(boston_model, boston_test_norm)

pred_medv <- model_results$net.result
pred_medv

# 5. 테스트 데이터의 집값만 역정규화 시킨다.
denormalize <- function(x) { return ( x*45+5 ) } #max(train_data$medv) #min(train_data
$medv)
pred_medv_un <- denormalize(pred_medv)

# 6. 제출 포맷으로 변경
sample <- cbind(boston_test$ID, pred_medv_un)
head(sample)

colnames(sample) <- c("id", "medv")

write.csv(sample, "Submission_sample.csv", row.names=FALSE)

```

5. 테스트 데이터를 예측하고 캐글에 결과물 제출

다중 회귀 분석으로 보스톤 집값을 예측하는 모델을 만들어서 캐글 상위권 도전

```

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 + age_dis,
  data=boston_train_norm)

summary(boston_reg_model)

model_results <- predict(boston_reg_model, boston_test_norm)

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_sample2.csv", row.names=FALSE)

```

문제269. 더미변수가 생성되게 하고 결정계수를 높이시오.

문제270. 다중회귀 분석으로 보스톤 우중 데이터의 집값 예측 모델을 만들어서 테스트 데이터를 예측하고 케글에서 점수를 확인하시오.

머신러닝 종류 3가지

1. 지도학습

- 분류 : knn(3장), naivebayes(4장), decision tree(5장), 신경망(7장)
- 회귀 : 회귀분석(6장)

2. 비지도학습

- apriori 알고리즘(연관규칙 8장)
- k-means(9장)

3. 강화학습

연관규칙

연관규칙이란 데이터 내부에 존재하는 항복간의 상호관계 혹은 종속관계를 찾아내는 분석기법이다. 데이터 간의 관계에서 조건과 반응을 연결하는 분석으로 장바구니 분석 또는 서열 분석이라고 한다.

Apriori 알고리즘

간단한 성능 측정치를 이용해서 거대한 데이터에서 데이터간의 연관성을 찾는 알고리즘

Apriori 알고리즘이 유용한 데이터 패턴

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 적용할 수 있다.

연관규칙에 관련한 중요 용어 3가지

1. 지지도 : 전체 거래중 항목 A 와 B를 동시에 포함하는 거래의 비율

A 와 B가 동시에 포함된 건수

$$P(A \cap B) = \frac{\text{A 와 B가 동시에 포함된 건수}}{\text{전체 거래 건수}}$$

1.1 X 아이템의 지지도

$n(X)$ <- 아이템 X

$$\text{표기식} : \text{support}(X) = \frac{n(X)}{N}$$

N <- 전체 거래 건수

1.2 두 아이템 X와 Y의 지지도

$$n(X \cap Y) \quad <- \text{아이템 } X \text{와 } Y \text{를 포함하는 건수}$$

표기식 : $\text{support}(X, Y) = \frac{n(X \cap Y)}{N}$

$N \quad <- \text{전체 거래 건수}$

2. 신뢰도 : A 상품을 샀을 때 B 상품을 살 조건부 확률에 대한 척도
(두 아이템의 연관규칙이 유용한 규칙일 가능성의 척도)

$$\text{confidence}(X \rightarrow Y) = \frac{P(A \cap B)}{P(A)}$$

----- = -----
A를 포함하는 거래수

$$n(X \cap Y) \quad <- \text{아이템 } X \text{와 } Y \text{가 동시에 발생하는 건수}$$

표기식 : $\text{confidence}(X \rightarrow Y) = \frac{n(X \cap Y)}{n(X)}$

$n(X) \quad <- \text{전체 건수에서 아이템 } X \text{의 건수}$

아이템 X를 포함하는 거래중에서 아이템 Y도 포함하는 거래비율(조건부 확률)을 말한다.
신뢰도가 높을 수록 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)

$$\text{신뢰도} = \frac{P(A \cap B)}{P(A) \times P(B)}$$

----- = -----

$$c(X \rightarrow Y)$$

표기식 : $\text{lift}(X \rightarrow Y) = \frac{c(X \rightarrow Y)}{s(Y)}$

$$\text{향상도} = \frac{P(B|A)}{P(B)} = \frac{P(A \cap B)}{P(A) P(B)}$$

----- = -----

$A \text{와 } B \text{를 포함하는 거래건수}$

 $A \text{를 포함하는 거래건수} \times B \text{를 포함하는 거래건수}$

아이템 X가 주어지지 않았을 때의 아이템 Y의 확률대비
아이템 X가 주어졌을 때의 아이템 Y의 확률증가 비율을 말한다.

향상도	설명	예시
향상도 = 1	서로 독립적 관계	과자와 후추
향상도 > 1	양(+)의 상관관계	빵과 버터
향상도 < 1	음(-)의 상관관계	설사약과 변비약

문제271. 다음의 데이터의 지지도를 구하시오.

거래번호	거래 아이템
1	우유, 버터, 시리얼
2	우유, 시리얼
3	우유, 빵
4	버터, 맥주, 오징어

$$\text{지지도}(\text{우유} \rightarrow \text{시리얼}) = \frac{n(\text{우유} \cap \text{시리얼})}{\text{전체 건수}} = \frac{2}{4} = 50\%$$

문제272. 우유를 구매할 때 시리얼도 구매할 신뢰도를 구하시오.
(조건부 확률)

$$\text{표기식} : \text{confidence}(X \rightarrow Y) = \frac{n(X \cap Y)}{n(X)} = \frac{2}{3} = 66\%$$

문제273. 우유를 구매할 때 시리얼도 구매할 향상도를 구하시오.

$$\text{표기식} : \text{lift}(X \rightarrow Y) = \frac{\text{confidence}(X \rightarrow Y)}{\text{support}(Y)} = \frac{(2/3)}{(2/4)} = \frac{4}{3} = 1.33\%$$

$$\text{향상도} : \text{lift}(A \rightarrow B) = \frac{P(B|A)}{P(B)} = \frac{P(A \cap B)}{P(A) P(B)} = \frac{\text{A와 } b\text{를 포함하는 거래 건수}}{\text{A를 포함하는 거래건수} \times \text{B를 포함하는 거래건수}}$$

문제274. 우유와 시리얼을 샀을때의 지지도와 신뢰도를 각각 구하시오.

	지지도	신뢰도
우유 -> 시리얼	50%	66%
시리얼 -> 우유	50%	100%

문제275. 다음은 남학생과 여학생이 선호하는 책에 대한 교차표이다. 전체에서 1명을 뽑았을 때 그 학생이 남학생인 경우 소설책을 좋아할 확률을 구하시오.

	소설책	여행책
남학생	50	30
여학생	10	20

$$P(\text{소설책} \cap \text{남학생}) = 50/110$$

$$P(\text{소설책} | \text{남학생}) = \frac{50}{110} = \frac{5}{11} = 5/8$$

문제276. 다음은 쇼핑몰의 거래내역이다. 연관규칙 : 우유 -> 빵에 대한 신뢰도를 구하시오.

항목	거래수
우유	10
빵	20
우유, 빵	50
빵, 초콜릿	40
전체거래건수	100

$$P(\text{빵} \cap \text{우유}) = 50/100$$

$$P(\text{빵} | \text{우유}) = \frac{50}{100} = \frac{5}{10} = 5/6$$

문제277. 다음 중 연관규칙 분석에서 품목 A의 거래수가 5이고 품목 B의

거래수가 3, A와 B가 동시에 포함된 거래수가 4이다. 전체 거래수가 10일 때 품목 A, B의 지지도를 구하시오.

$$\text{support}(A \rightarrow B) = \frac{n(A \cap B)}{N} = \frac{4}{10} = 0.4$$

문제278. 아래는 A 쇼핑몰의 거래내역이다. 연관규칙 : 과자 -> 기저귀에 대한 지지도를 구하시오.

품목	판매수량
과자	10
기저귀	20
과자, 기저귀	40
기저귀, 맥주	20
전체판매수량	100

$$\text{P}(\text{과자} \cap \text{기저귀}) = \frac{n(\text{과자} \cap \text{기저귀})}{N} = \frac{40}{100} = 0.4$$

문제279. 아래는 A 쇼핑몰의 거래내역이다. 연관규칙 : 과자 -> 기저귀에 대한 신뢰도와 향상도를 구하시오.

$$\text{신뢰도} = \frac{\text{P}(\text{기저귀} \cap \text{과자})}{\text{P}(\text{과자})} = \frac{40}{50} = 0.8$$

$$\text{향상도} = \frac{\text{confidence}(\text{과자} \rightarrow \text{기저귀})}{\text{support}(\text{기저귀})} = \frac{0.8}{0.8} = 1$$

컴퓨터가 연관규칙을 찾는 샘플

거래번호	아이템 목록
1	A, B, D
2	B, C
3	A, B, C, E
4	B, C, E

위의 거래들을 보고 가장 적절한 아이템 목록의 조합을 알아내야 한다.

1. 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 큰것만 추출해서 다시 정리한다.

아이템	지지도
A	2
B	4
C	3
E	2

2. 아이템들간의 연관규칙을 알기 위해서 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

위의 결과에서 지지도가 1보다 큰것만 추출해서 다시 정리한다.

아이템	지지도
A, B	2
B, C	3
B, E	2
C, E	2

위의 A B C E로 만들 수 있는 3개의 조합으로 지지도를 구성한다.

아이템	지지도
A, B, C	1
A, B, E	1
A, C, E	1
B, C, E	2

지지도가 1보다 큰것만 추출해서 정리

아이템	지지도
B, C, E	2

아프리오 알고리즘 예제1

(맥주와 기저귀)

<https://cafe.daum.net/oracleoracle/SZTZ/2203>

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

```

# 2. arules 패키지를 설치한다.
# 아프리오의 알고리즘을 사용할 수 있는 패키지
install.packages("arules")
library(arules)

# x 데이터 프레임을 행렬로 변환
trans <- as.matrix(x, "Transaction")
trans

# 3. apriori 함수를 이용해서 연관관계를 분석한다.
# 지지도가 0.2 이상이고 신뢰도가 0.6 이상인 rule을 발견
rules1 <- apriori(trans, parameter=list(supp=0.2,
                                         conf=0.6, target="rules"))
rules1

# 규칙을 확인
inspect(sort(rules1))

# 4. 위의 맥주와 기저귀 연관 관계를 시각화 하기
install.packages("sna")
install.packages("rgl")
library(sna)
library(rgl)
# 희소행렬을 만든다.
b2 <- t(as.matrix(trans)) %*% as.matrix(trans)

# 위에서 만든 희소행렬에서 대각선에 해당하는 데이터를 빼서
# 대각선 데이터를 0이 되게 한다.
b2.w <- b2 - diag(diag(b2))
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)

```

아프리오 알고리즘 예제2

(보습학원과 연관된 업종)

<https://cafe.daum.net/oracleoracle/SZTZ/2204>

```

# 1. 데이터를 로드합니다.
build <- read.csv("building.csv", header = T)

# 2. na 를 0 으로 변경합니다.
build[is.na(build)] <- 0

# 3. 필요한 변수만 선별합니다.
build <- build[-1]
View(build)

# 4. 연관규칙 패키지를 다운로드 받습니다.

```

```

# install.packages("arules")
library(arules)

# 5. 연관규칙 모델을 생성합니다.
# 행렬로 변환
trans <- as.matrix(build , "Transaction")

# 지지도 0.2이상 신뢰도 0.6이상인 룰을 찾는다.
rules1 <- apriori(trans , parameter = list(supp=0.2 , conf = 0.6,
                                             target = "rules"))
rules1

# 6. 연관규칙을 확인합니다.
inspect(sort(rules1))
View(inspect(sort(rules1)))

# 7. 시각화를 합니다.

# 여러 규칙들 중에서 보습학원 부분만 따로 검색
rules2 <- subset(rules1 , subset =
                  lhs %pin% '보습학원' & confidence > 0.7)
inspect(sort(rules2))

# 여러 규칙들 중에서 편의점 부분만 따로 검색
rules3 <- subset(rules1 , subset =
                  rhs %pin% '편의점' & confidence > 0.7)
rules3
inspect(sort(rules3))

#visualization
# 희소행렬로 변경
b2 <- t(as.matrix(build)) %*% as.matrix(build)
# install.packages("sna")
# install.packages("rgl")
library(sna)
library(rgl)

# 희소행렬의 대각선을 0으로 변경
b2.w <- b2 - diag(diag(b2))
#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)

```

연관규칙과 연관된 데이터 분석(12장)

사회적 연결망 데이터 분석

- 개인과 집단간의 관계를 노드와 링크로서 모델링해 그것의 구조나 확산 및 진화과정을 계량적으로 분석하는 방법론이다.

이때 노드란 분석하고자하는 객체로써, 사람이나 사물 등을 말하는 것이다.

이 노드간의 관계를 나타내는 연결을 링크라고 한다.

노드에 연결된 링크의 수를 Degree라고 한다.

<https://cafe.daum.net/oracleoracle/Sg0x/715>

1. 요번주에 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)
```

```
paper2
```

2. 요번주에 개별적으로 책을 읽은 시간 데이터를 로드한다.

```
book <- read.csv("book_hour.csv", header = T)
```

```
paper2
```

```
book
```

```
library(sna)
```

```
x11()
```

```
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)
```

문제280. 위의 시각화 결과에서 광희에 해당하는 부분만 따로 분리해서 시각화 하시오. (변경)

머신러닝 종류 3가지

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

k-means 알고리즘

k-means 군집화 이론

- k-means 알고리즘은 주어진 데이터를 k개의 클러스터로 묶는 알고리즘으로, 각 클러스터와 거리차이의 분산을 최소화하는 방식으로 동작한다.

이 알고리즘은 자율학습의 일종으로 레이블(정답)이 달려있지 않은 입력 데이터에 레이블(정답)을 달아주는 역할로도 활용되고 있다.

정답이 없는 데이터 속에서 어떤 패턴을 찾고 싶을 때 비지도 학습의 k-means 군집화 머신러닝 알고리즘을 활용한다.

오라클의 머신러닝 패키지를 이용해서 만든 사례

1. 모 통신사에서 기지국을 세울 때 사용
2. 병원에서 암 판별 머신러닝 모델을 만드는데 라벨링있는 지도학습데이터를 훈련 시킬 때 비지도학습을 같이 사용해서 모형의 정확도를 올리는데 참조
3. 마케팅쪽에서 세그멘테이션(segmentation)에 활용
 - 고객들을 특성에 맞게 군집화 한다.
4. 보험회사에서 세그멘테이션(segmentation)하여 고객별 맞춤형 보험 상품 개발 및 광고를 진행
5. 미국의 순찰지역을 정하는데 활용하여 범죄를 미리 예방하는데 사용

k-means 실습1

<https://cafe.daum.net/oracleoracle/Sg0x/731>

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

# 2. 위에서 만든 데이터 셋으로 plot 그래프를 그린다.
plot(data)

# x와 y 좌표로 되어있는 dta를 2개로 군집화한다.
km <- kmeans( data, 2 )

km$cluster

cbind(data, km$cluster)

km$centers

# 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)

plot( data, col=km$cluster + 1, xlim=range(0:10), ylim=range(0:10) )

```

군집간의 거리 측정 방법

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 간 국집에서 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
 - a. 수학적 거리
 - i. 유클리드 거리 계산 (p 410)
 - ii. 맨하튼 거리 계산
 - iii. 민코프스키 거리 계산
 - b. 통계적 거리
 - i. 표준화 거리 계산

- ii. 마할라노비스 거리 계산
- 2. 명목형 변수 거리 계산

k-means 군집의 절차

유클리드 거리로 군집화 하는 방법
<https://cafe.daum.net/oracleoracle/SeFi/458>

단계	알고리즘	설명
1	k개 객체 선택	초기 군집중심으로 k개의 객체를 임의로 선택
2	할당	자료를 가장 가까운 군집 중심에 할당
3	중심을 갱신	각 군집 내의 자료들의 평균을 계산하여 군집 중심을 갱신
4	반복	군집 중심의 변화가 거의 없을 때까지 단계2와 3을 반복

문제281. 아래는 학생들의 키와 몸무게를 정규화한 데이터이다. 유 클리디안 거리를 사용해서 최단 연결법을 통해 학생들을 3개의 군 집으로 나누고자 할 때 가장 적정한 것을 고르시오.

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)

4번

k-means의 단점

1. k값을 지정하기가 어렵다.
 - 적당한 k 값을 계산하는 공식 : $k = \sqrt{n/2}$, n(전체건수)
2. 이상치에 민감하다.





k-means

- 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 비지도학습은 정답이 없기 때문에 타당성 검증이 어렵다.
 - 클러스터링한 결과를 가지고 기술 통계 기법을 사용하여 잘 군집화 했는지 분석한다.(p 430)

소셜 미디어의 글들을 희소 행렬로 만들고 k-means로 군집화 해서 사람들의 특성을 5 가지로 분류

<http://cafe.daum.net/oracleoracle/SeFi/460>

문제282. 아래의 데이터 세트 A와 B간의 맨하튼 거리로 계산하면 얼마인지 확인하시오.

	김xx	서xx
키	165	180
몸무게	50	65

k-means 실습예제1

<https://cafe.daum.net/oracleoracle/SZTZ/2221>

1. 기본 데이터 셋을 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
```

2. 위에서 만든 데이터셋으로 plot 그래프를 그린다

```
plot(data)
```

3. k-means 패키지를 설치한다

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

4. kmeans 함수로 데이터를 분류한다.

```
# k 개 구하는 공식 : k=sqrt(n/2)
km <- kmeans(data,2)
km
```

```

km$center # 중앙점이 어딘지 확인한다.
cbind(data, km$cluster)

# 6. 원래 데이터를 그린 plot 그래프와 위의 그래프를 합쳐서 출력한다.
plot(round(km$center), col=km$center, pch=22, bg=km$center,
      xlim=range(0:10), ylim=range(0:10))
par(new=T)
plot( data, col=km$cluster+1, xlim=range(0:10), ylim=range(0:10) )

# 7. 위의 data 를 factoextra 패키지를 이용해서 시각화 한다.
install.packages("factoextra")
library(factoextra)

km <- kmeans(data,2)

fviz_cluster( km, data = data, stand=F)

```

문제283. factoextra 패키지를 이용해서 동물 데이터를 k-means로 군집화하고 시각화 하시오.

<https://cafe.daum.net/oracleoracle/Sg0x/751>

```

zoo <- read.csv("zoo.csv")
zoo_n <- zoo[ , 2:17] # 동물이름과 라벨을 제외한 컬럼으로 구성
ncol(zoo_n)

zoo_model <- kmeans(zoo_n, 7) # k값을 7로 주고 군집화하는 모형 생성
x <- cbind( zoo[ , 18], zoo_model$cluster)
x2 <- data.frame(x)
x2
library(doBy)
orderBy(~X1, x2)

library(factoextra)
km <- kmeans(zoo_n,7)
fviz_cluster( km, data = zoo_n, stand=F)

```

k-means 시각화 할 때 fviz_cluster함수가 유용하다.

k-means 실습예제2

<https://cafe.daum.net/oracleoracle/SeFi/459>

```

# 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

plot(data)

# 2. 야채,과일,단백질 3가지를 k-means 가 잘 분류 했는지 시각화
# install.packages("stats")
# library(stats)
km <- kmeans(data, 3)

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
# install.packages("factoextra")
library(factoextra)

km <- kmeans(data,3)

fviz_cluster( km, data = data, stand=F)

```

문제284. 유방암 데이터(wisc_bc_data.csv)의 악성종양과 양성종양을 k-means로 2개로 군집화 해서 정답과 비교하시오.

```

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)

```

k-means 실습예제3

<https://cafe.daum.net/oracleoracle/SZTZ/2222>

```

# 1. 데이터를 로드한다.
academy <- read.csv("academy.csv")
academy <- academy[ , c(3,4) ]
View(academy)

# 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)

```

문제285. 위의 결과에서 수학점수는 높으나 영어점수가 보통인 학생들의 번호만 추출하시오.

```

# 1. 데이터를 로드한다.
academy <- read.csv("academy.csv")
academy <- academy[ , c(3,4) ]
View(academy)

# 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")
academy_seg <- cbind( academy[ ,c(1,3,4)], km$cluster)
academy_seg[km$cluster == 2, 1]

```

책의 예제를 실습하기 전에 미리 알아야하는 내용

1. imputation (결측치를 다른 값으로 치환)
2. ave함수

결측치를 다른 값으로 대체하는 방법

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.

ex)

1:10

ave(1:10) # 1~10 까지의 숫자들의 평균값이 입력된 숫자의 갯수만큼 출력

문제286. 부서번호, 부서번호별 평균월급을 출력하시오.

aggregate(sal~deptno, emp, mean)

문제287. 사원 테이블에서 자기가 속하는 부서번호의 평균 월급이 출력되게 하시오.

ave(emp\$sal) # emp 테이블의 월급 평균값이 14개가 출력

ave(emp\$sal, emp\$deptno)

문제288. 이름, 월급, 부서번호, 자기가 속한 부서번호의 평균월급을 출력하시오.

```
select ename, sal, deptno,
       avg(sal) over(partition by deptno) as avgSal
  from emp;
```

emp\$avgSal <- ave(emp\$sal, emp\$deptno, FUN=function(x) mean(x))

emp[, c('ename', 'sal', 'deptno', 'avgSal')]

FUN=function(x) mean(x)는 생략 가능

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)

emp

k-means 실습예제4

<https://cafe.daum.net/oracleoracle/Sefi/460>

1. 데이터를 로드한다.

corpus 패키지를 가지고 만들어낸데이터

teens <- read.csv("snsdata.csv")

View(teens)

2. 성별이 남자가 몇명이고 여자가 몇명인지 확인한다.

table(teens\$gender)

prop.table(table(teens\$gender))

3. 성별에 NA 가 몇개인지도 출력되게하시오

table(teens\$gender, useNA="ifany")

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)

colSums(is.na(teens))

5. 유클리드 거리 계산을 위해 성별에 관련한 더미변수 2개를 생성한다.

성별이 여자이고 결측치가 아닌 데이터는 1로 출력하고 아니면 0으로 출력

teens\$female <- ifelse(teens\$gender=="F" & !is.na(teens\$gender), 1, 0)

성별이 없는 사람들을 no_gender 이라는 파생변수를 통해서 처리될 수 있도록

성별이 NA이면 1 아니면 0을 출력하는 파생변수를 만든다.

teens\$no_gender <- ifelse(is.na(teens\$gender),1,0)

기존 성별 데이터에 NA값이 몇개가 있는지 확인

table(teens\$gender, useNA="ifany")

여성이면 1 아니면 0인 건수가 어떻게 되는지 확인

table(teens\$female, useNA="ifany")

성별이 없으면 1 아니면 0인 건수가 어떻게 되는지 확인

table(teens\$no_gender, useNA="ifany")

결측치가 있으면 유클리드 거리계산 공식을 사용할 수 없으므로 결측치를

다른 값으로 치환한다.

6. 나이가 결측치로 나온 데이터를 졸업년도로 나이를 추정해서 결측치를 채워넣는 작업

나이에 대한 결측치도 성별처럼 파생변수를 생성하던지 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)

colSums(is.na(teens))

```

# 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))

summary(interests_z)

# 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

```

문제291. 남녀 성별의 비율을 출력하시오.

```

# 1. 데이터를 로드한다.
# corpus 패키지를 가지고 만들어낸데이터
teens <- read.csv("snsdata.csv")

# 2. 성별이 남자가 몇명이고 여자가 몇명인지 확인한다.
table(teens$gender)

prop.table(table(teens$gender))

```

문제292. teens 데이터 프레임에 결측치가 얼마나 있는지 조회하시오.

```
colSums(is.na(teens))
```

문제293. 졸업연도, 졸업연도별 평균나이를 출력하시오.

```
aggregate(age~gradyear, data=teens, mean, na.rm=T)
```

문제294. teens 데이터 프레임의 결측치를 확인하시오.

```
colSums(is.na(teens))
```

문제295. 저자의 해석을 알아내기 위해 군집별로 female(여성)이 몇 명인지 출력되게 하시오.

```
teens <- cbind(teens, teen_clusters$cluster)
```

```
attach(teens)
```

```
tapply(female, teen_clusters$cluster, sum)
```

21.02.10

2021년 2월 10일 수요일 오전 9:43

SOM(Self Organizing Map) 분석

<https://cafe.daum.net/oracleoracle/Sg0x/768>

비지도 학습 신경망으로 고차원의 데이터를 이해하기 위해 저차원의 뉴런으로 정렬하여 지도의 형태로 형상화 하는 기법

k-means 군집화 : 유clidean 거리공식을 이용해서 가장 가까운 이웃끼리 군집화하는 알고리즘

SOM : 비지도학습 + 신경망

iris 데이터를 som 비지도학습 신경망을 이용해서 군집화하는 실습

```
# 1. 아이리스 데이터의 컬럼 이름 확인 및 건수 확인  
colnames(iris)
```

```
# 2. 아이리스의 정답컬럼의 level 확인  
levels(iris$Species)
```

```
# 3. som 패키지 설치  
install.packages('kohonen')  
library(kohonen)
```

```
# 4. 훈련데이터와 테스트데이터를 3대 1로 분류  
train <- sample(1:150, 100) #무작위로 100개 추출 (학습데이터)
```

```
# som 함수가 list형태의 데이터를 원하므로 list형태로 변환한다.  
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
```

```
# 5. somgrid 함수를 이용해서 경쟁층을 구현한다.  
gr <- somgrid(xdim = 3, ydim = 5, topo = "hexagonal")  
#grid 갯수 및 모양 설정
```

```
#토플로지(영어: topology, 문화어: 망구성방식)는 컴퓨터 네트워크의 요소들  
#(링크, 노드 등)을 물리적으로 연결해 놓은 것
```

```
# 6. 훈련 데이터 100개로 som학습 시키기
```

```

# rlen = 200 : 150개의 데이터를 200번 학습 시킨다는 뜻이다.
# alpha = c(0.05, 0.01) : 러닝 레이트를 0.05에서 시작해서 0.01까지 학습
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 확인
# 학습이 진행될수록 경쟁층의 뉴런과 입력 데이터와의 거리가 짧아지는지 확인
plot(ss, type="changes")

# 8. som모델의 각 뉴런(경쟁층의 뉴런)이 몇개의 입력 데이터와 매핑 되는지 확인
plot(ss, type="count", main="Node Counts")

# 9. 경쟁층의 각 뉴런끼리의 거리를 확인
plot(ss, type="dist.neighbours", main = "SOM neighbour distances")

# 10. 경쟁층의 뉴런에 속한 입력층의 종류가 무엇인지 확인
plot(ss, type="codes")

# 11. 테스트 데이터를 잘 맞추는지 확인
test_Set$Species # 실제정답

test.pred <- predict(ss, newdata = test_Set)

test.pred$predictions$Species

# 12. 예측값과 실제값을 비교
table(test.pred$predictions$Species,test_Set$Species)

```

문제296. 다음중 SOM에 대한 설명으로 가장 알맞지 않은 것을 고르시오.

1. SOM은 경쟁학습으로 각각의 뉴런이 입력 백터와 얼마나 가까운가를 계산하여 연결강도를 반복적으로 재조정하는 학습과정을 거치면서 연결 강도는 입력 패턴과 가장 유사한 경쟁층의 뉴런이 승자가 된다.
2. SOM은 고차원의 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 보존하여 입력변수의 정보와 그들의 관계가 지도상에 그대로 나타난다.
4. SOM을 이용한 군집분석은 역전파 알고리즘을 사용하므로써 군집의 성능이 우수하고 수행속도가 빠르다.

모델 성능 평가

머신러닝이 수행한 결과(분류, 예측)에 대한 공정한 평가를 통해 머신러닝이 앞으로도 미래의 데이터에 대해 잘 분류하고 예측할 수 있도록 해주고 분류결과가 요행수로 맞힌게 아니라는 것을 확신하게 해주며 분류 결과를 좀 더 일반화 할 수 있기 때문에 중요하다.

다른 성능 척도

1. 카파통계량
2. 민감도와 특이도
3. 정밀도와 재현율
4. Roc 곡선
5. F1 척도

정확도 계산하는 방법

$$\text{정확도} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

실제 분류 범주를 정확하게 예측한 비율
전체 예측에서 긍정(TP)와 참 부정(TN)이 차지하는 비율

<https://cafe.daum.net/oracleoracle/Sg0x/791>

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 negative이나 예측값은 positive인 경우
- FN(False Negative) : 실제값은 positive이나 예측값은 negative인 경우

Positive : 관심범주

- 암을 분류하는 분류기를 만들 때 관심범주가 암이기 때문에 암(Postive), 정상(Negative)이 된다.
- 스팸메일을 분류하는 분류기를 만들 때 스팸메일이 Positive이고 햄메일이 Negative가 된다.

카파통계량 (p 449)

두 관찰자간의 측정 범주값에 대한 일치도를 측정하는 방법

$$\Pr(a) - \Pr(e)$$

$$k = \frac{40 + 30}{100} = 0.7$$

$Pr(a)$: 데이터에서 관찰된 2명의 평가자들의 일치 확률

$Pr(e)$: 2명의 평가자들이 데이터로 부터 계산된 확률적 일치 확률(우연히 일치할 확률)

카파통계량이 0이면 완전 불일치이고 1이면 완전 일치이다.

k	일치정도
0.8	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

ex)

시험을 응시한 학생이 100명이라고 할 때, 2명의 평가자가 합격, 불합격을 각각 판정하고 두 평가자의 일치도를 아래와 같이 보여주고 있다.

		평가자 A	
		합격	불합격
평가자 B	합격	40	10
	불합격	20	30

$$Pr(a) = \frac{40 + 30}{100} = 0.7$$

평가자 A : 합격 60, 불합격 40

합격 확률 : 0.6

불합격 확률 : 0.4

평가자 B : 합격 50, 불합격 50

합격 확률 : 0.5

불합격 확률 : 0.5

평가자 A와 B 둘 모두 확률적으로 합격을 줄 확률 : $0.6 \times 0.5 = 0.3$

평가자 A와 B 둘 모두 확률적으로 불합격을 줄 확률 : $0.4 \times 0.5 = 0.2$

$$\Pr(e) = 0.3 + 0.2 = 0.5$$

$$k = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)} = \frac{0.7 - 0.5}{1 - 0.5} = \frac{0.2}{0.5} = 0.4$$

문제297. 책 446페이지의 이원 교차표의 카파 통계량을 계산하시오.

<https://cafe.daum.net/oracleoracle/Sg0x/792>

$$\Pr(a) = 0.9748$$

$$\Pr(e) = \text{ham}(A) \times \text{ham}(B) + \text{spam}(A) \times \text{spam}(B) = 0.7857$$

$$\text{ham}(A) : 0.8878$$

$$\text{spam}(A) : 0.1122$$

$$\text{ham}(B) : 0.8683$$

$$\text{spam}(B) : 0.1317$$

$$k = \frac{0.9748 - 0.7857}{1 - 0.7857} = \frac{0.1891}{0.2143} = 0.8824$$

문제298. 위의 카파통계량을 R로 구하시오.

```
sms_result <- read.csv('sms_results.csv')
head(sms_result)
```

```
install.packages('vcd')
library(vcd)
```

```
table(sms_result$actual_type, sms_result$predict_type)
```

```
Kappa(table(sms_result$actual_type, sms_result$predict_type))
```

p453 참조 kappa의 k를 소문자로 사용하지 않도록 주의해야 한다.

문제299. 아래의 혼동행렬에서 카파통계량을 구하시오.

실제값/예측치	True	False	합계
True	30	70	100
False	60	40	100
합계	90	110	200

```
m <- as.table(matrix( c(30, 70, 60, 40), nrow=2, byrow=T))
Kappa(m)
```

민감도와 특이도(p454)

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 긍정인 범주 중에서 긍정으로 올바르게 예측한 비율

$$\text{sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

특이도 : 실제로 부정인 범주 중에서 부정으로 올바르게 예측한 비율

$$\text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

민감도와 특이도도 카파 통계량 처럼 0~1 까지의 범위에 있으며, 값이 1에 가까울 수록 바람직하나 제로는 한쪽이 높으면 한쪽이 낮아져서 둘 다 높게 맞출수가 없다.

만약 유방암을 예측하는 머신러닝 모델을 아래와 같이 만들었다.

	knn모델	decision tree	신경망
정확도	99%	99%	99%
민감도	0.6	0.7	0.7
특이도	0.5	0.4	0.5

위의 3가지 모델 중 하나를 선택해야 한다면 민감도를 최고로 높게 맞춰놓고 그중에 특이도가 높은것을 선택한다.

민감도가 높으면 특이도가 다소 낮더라도 가치는 충분하다.

문제300. 스팸분류기의 이원교차표를 보고 민감도와 특이도를 R로 구현하시오.

```
sms_result <- read.csv('sms_results.csv', stringsAsFactors=T)
```

```

head(sms_result)

install.packages('caret')
library(caret)

# 민감도
sensitivity(sms_result$predict_type, sms_result$actual_type, positive='spam')

# 특이도
specificity(sms_result$predict_type, sms_result$actual_type, negative='ham')

```

문제301. 아래의 혼동행렬에서 특이도를 구하시오.

실제값/예측치	True	False	합계
True	30	70	100
False	60	40	100
합계	90	110	200

$$\frac{\text{TN}}{\text{TN} + \text{FP}} = 0.4$$

정밀도와 재현율(p456)

정밀도 : 참으로 예측한 것 중에 실제 참인것의 비율

$$- \text{precision} = \text{TP}/(\text{TP}+\text{FP})$$

재현율 : 실제 참인것중에서 참이라고 예측한 비율 (민감도와 같다)

$$- \text{TP}/(\text{TP}+\text{FN})$$

1. 소극적 예측 : 암이라고 판단하는것 자체를 소극적으로 봐서 확실한 경우가 아니면 암으로 판단하지 않는 것이다.
- 정밀도 ↑ 재현율 ↓
2. 공격적 예측 : 조금만 의심이 가도 암이라고 판단
- 정밀도 ↓ 재현율 ↑

문제302. 아래의 스팸 분류기의 정밀도를 구하시오.

```

sms_result <- read.csv('sms_results.csv', stringsAsFactors=T)
head(sms_result)

install.packages('caret')
library(caret)

# 민감도(재현율)

```

```
sensitivity(sms_result$predict_type, sms_result$actual_type, positive='spam')
```

정밀도

```
posPredValue(sms_result$predict_type, sms_result$actual_type, positive='spam')
```

문제303. 아래의 혼동행렬에서 민감도를 구하시오.

실제값/예측치	True	False	합계
True	30	70	100
False	60	40	100
합계	90	110	200

$$\frac{\text{TP}}{\text{TP} + \text{FN}} = 0.3$$

머신러닝의 종류 3가지

1. 지도학습

- a. 분류 : knn, naivebayes, decision tree, rule, riper, 신경망, 서포트 벡터 머신, 로지스틱 회귀, 랜덤포레스트, xgboost
- b. 예측 : 다중회귀분석

2. 비지도학습 : 연관분석, k-means, som

3. 강화학습

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					
$\frac{TP}{TP + FP}$					

카파통계량, 정확도, 민감도, 특이도, 정밀도, 재현율 출력 실습

<https://cafe.daum.net/oracleoracle/Sg0x/829>

#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_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] )

#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
predict_type <- credit_result # 머신러닝 모델이 예측한 정답
positive_value <- 'yes' # 관심범주
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)
```

문제304. 위의 머신러닝 모델의 성능을 올리는 테스트를 진행하시오.
(하이퍼 파라미터 seed값과 trials를 조정해서 정확도와 정확도 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(69)  
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] # 실제 테스트 데이터의 라벨  
predict_type <- credit_result # 머신러닝 모델이 예측한 정답
```

```

positive_value <- 'yes' # 관심범주
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. 위의 내용을 엑셀에 표로 정리하시오.

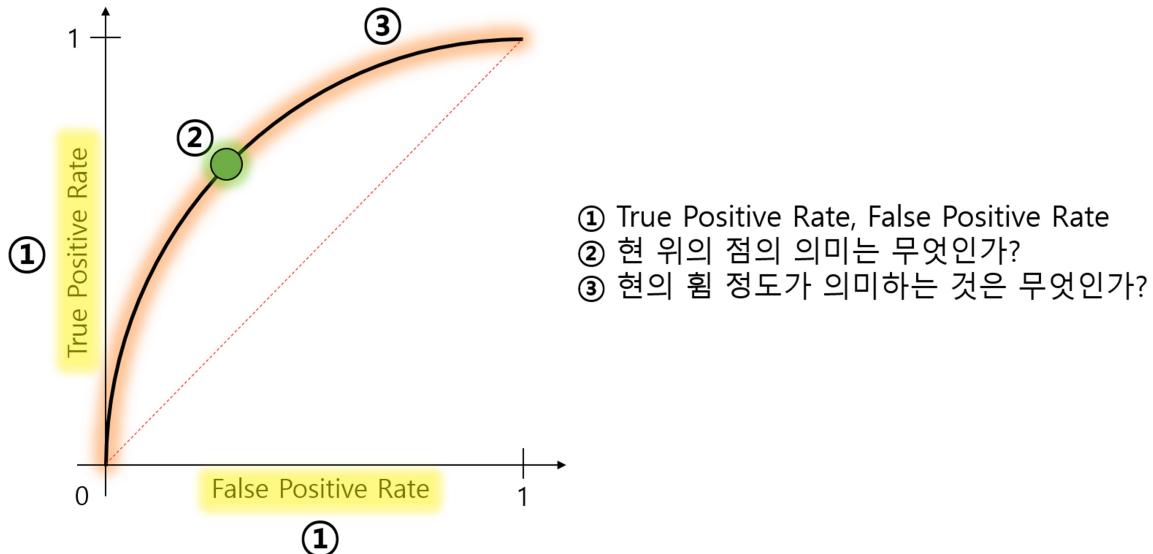
ROC 곡선

<https://cafe.daum.net/oracleoracle/Sg0x/827>

Receiver Operating Charateristic

이진 분류기의 성능 평가 기법중에 하나이다.

2차원 위에 그려지는 그래프이고 x축이 False Positive Rate이고 y축이 True Positive Rate이다.



다른 성능척도인 민감도와 특이도를 숫자로만 보는게 아니라 시각화해서 여러 모델중에 가장 좋은 모델을 쉽게 찾게해주는 그래프가 ROC 그래프이다.

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")
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)

```

문제306. ROC 그래프를 그리는 코드를 돌려서 시각화하는데 그래프 색깔을 변경하시오.

#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

```
# type='prob' 옵션을 주게 되면 확률이 출력된다.
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], # 테스트 데이터의 실제 정답
                             predict_type = credit_result, # 테스트 데이터에 대한 예측값
                             prob_yes = round(credit_test_prob[ , 2], 5), # yes일 확률
                             prob_no = round(credit_test_prob[ , 1], 5)) # no일 확률
```

```
credit_results
```

#3. 예측 데이터 프레임을 csv 로 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' # 관심번주
negative_value <- 'no'

# ROC 곡선 그리기
#install.packages("ROCR")
library(ROCR)
head(credit_results) # 3번째 컬럼과 4번째컬럼의 확률을 확인한다.
# prediction(predictions=관심범주의 확률, labels=실제정답)
# ROC 커브를 그리기 위한 100개의 데이터 포인트가 pred에 담긴다.
pred <- prediction(predictions = credit_results$prob_yes,
                     labels = credit_results$actual_type)
pred

# ROC curves
# performance(100개의 데이터 포인트, y축 값, x축 값)
perf <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(perf, main = "ROC curve for SMS spam filter", col = "red", lwd = 2)

# add a reference line to the graph
# 대각선 출력
abline(a = 0, b = 1, lwd = 2, lty = 2, col='gray')
# calculate AUC
perf.auc <- performance(pred, measure = "auc")
str(perf.auc)
unlist(perf.auc@y.values)

```

문제307. 위의 ROC 그래프를 다시 그리는데 seed값 : 659, trails : 100 으로 주고 다시 그리시오.

#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 )
```

실제값과 예측값 대입

type='prob' 옵션을 주게 되면 확률이 출력된다.

```
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], # 테스트 데이터의 실제 정답
                                predict_type = credit_result, # 테스트 데이터에 대한 예측값
                                prob_yes = round(credit_test_prob[ , 2], 5), # yes일 확률
                                prob_no = round(credit_test_prob[ , 1], 5)) # no일 확률
```

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] # 테스트 데이터의 라벨
```

```
predict_type <- credit_result # 테스트 데이터의 예측값
```

```
positive_value <- 'yes' # 관심변수
```

```
negative_value <- 'no'
```

ROC 곡선 그리기

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

```
head(credit_results) # 3번째 컬럼과 4번째 컬럼의 확률을 확인한다.
```

```
# prediction(predictions=관심변수의 확률, labels=실제정답)
```

```
# ROC 커브를 그리기 위한 100개의 데이터 포인트가 pred에 담긴다.
```

```
pred <- prediction(predictions = credit_results$prob_yes,
                     labels = credit_results$actual_type)
```

```
pred
```

```

# ROC curves
# performance(100개의 데이터 포인트, y축 값, x축 값)
perf <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(perf, main = "ROC curve for SMS spam filter", col = "red", lwd = 2)

# add a reference line to the graph
# 대각선 출력
abline(a = 0, b = 1, lwd = 2, lty = 2, col='gray')
# calculate AUC
perf.auc <- performance(pred, measure = "auc")
str(perf.auc)
unlist(perf.auc@y.values)

```

ROC 그래프에서 cut off 지정

<https://cafe.daum.net/oracleoracle/Sg0x/827>

로지스틱 회귀로 데이터를 분류하고 ROC 커브를 그린 다음 cut off 지점을 알아내는 코드

로지스틱 회귀 : 종속변수가 이진 데이터인 명목형 데이터인 회귀분석

```

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")
binary <- read.csv('c://data//binary.csv')
str(binary)

#Logistic Regression Model
install.packages("nnet")
library(nnet)

# multinom은 로지스틱 회귀 함수이다.
mymodel <- multinom(admit~,data = binary)

#mis classification rate
# 예측하고 테이블로 결과 확인
p <- predict(mymodel,binary)
tab <- table(p,binary$admit)
tab

# 오차율 확인
1-sum(253,29)/400

# Model Performance Evaluation
install.packages("ROCR")
library(ROCR)
pred <- predict(mymodel,binary,type = "prob") # 예측값에 대한 확률 출력
hist(pred) # 확률의 분포 확인

```

```

pred <- prediction(pred,binary$admit)
pred # ROC 커브를 그리기 위한 400개의 데이터 포인트를 뽑는다.

eval <- performance(pred,"acc")
eval # 392개의 데이터 포인트 추출

plot(eval)

abline(h=0.71,v=.45)

#Identifying the best cutoff and Accuracy
# x축이 cut off, y축이 정확도를 나타내는 그래프를 시각화하기 위한 데이터 포인트
eval

# 392개의 데이터 포인트 중 max값에 해당하는 인덱스 번호 출력
max <- which.max(slot(eval,"y.values")[[1]])

# y축 정확도 중 61번째에 해당하는 정확도 값 출력
acc <- slot(eval,"y.values")[[1]][[max]]

# x축 cut off 값들 중 61번째에 해당하는 값을 출력
cut <- slot(eval,"x.values")[[1]][[max]]

print(c(Accuracy=acc,Cutoff = cut))

#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)

```

문제308. 위의 정확도와 cutoff를 출력하는 코드를 이용해서 독일 은행 데이터의 정확도와 cut off를 구하시오.

```

#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
# type='prob' 옵션을 주게 되면 확률이 출력된다.
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], # 테스트 데이터의 실제 정답
                           predict_type = credit_result, # 테스트 데이터에 대한 예측값
                           prob_yes = round(credit_test_prob[ , 2], 5), # yes일 확률
                           prob_no = round(credit_test_prob[ , 1], 5)) # no일 확률

credit_results

#3. 예측 데이터 프레임을 csv 로 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' # 관심변수
negative_value <- 'no'

```

```

# ROC 곡선 그리기
#install.packages("ROCR")
library(ROCR)
head(credit_results) # 3번째 컬럼과 4번째컬럼의 확률을 확인한다.
# prediction(predictions=관심범주의 확률, labels=실제정답)
# ROC 커브를 그리기 위한 100개의 데이터 포인트가 pred에 담긴다.
pred <- prediction(predictions = credit_results$prob_yes,
                     labels = credit_results$actual_type)
pred

# ROC curves
# performance(100개의 데이터 포인트, y축 값, x축 값)
perf <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(perf, main = "ROC curve for SMS spam filter", col = "red", lwd = 2)

# add a reference line to the graph
# 대각선 출력
abline(a = 0, b = 1, lwd = 2, lty = 2, col='gray')
# calculate AUC
perf.auc <- performance(pred, measure = "auc")
str(perf.auc)
unlist(perf.auc@y.values)

# 정확도와 cut off 출력
eval <- performance(pred,"acc")
eval # 392개의 데이터 포인트 추출

plot(eval)

#Identifying the best cutoff and Accuracy
# x축이 cut off, y축이 정확도를 나타내는 그래프를 시각화하기 위한 데이터 포인트
eval

# 392개의 데이터 포인트 중 max값에 해당하는 인덱스 번호 출력
max <- which.max(slot(eval,"y.values")[[1]])

# y축 정확도 중 61번째에 해당하는 정확도 값 출력
acc <- slot(eval,"y.values")[[1]][[max]]

# x축 cut off 값들 중 61번째에 해당하는 값을 출력
cut <- slot(eval,"x.values")[[1]][[max]]

print(c(Accuracy=acc,Cutoff = cut))

# x축을 fpr로 하고 y축을 tpr로 하는 2차원 그래프에 cut off 지점으로 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 확인해야할 다른 성능 척도

카파통계량

민감도

특이도정밀도

재현율

AUC

cut off

$$\text{Precision(정밀도)} * \text{recall(재현율)}$$

$$\text{F1 SCORE : } 2 \times \frac{\text{Precision(정밀도)} * \text{recall(재현율)}}{\text{Precision(정밀도)} + \text{recall(재현율)}}$$

F1 값이 높으면 Precision와 recall 모두 좋은 결과를 보인다.

F1 값이 낮으면 잘못 판단한 False값들에 문제가 있다는 것이다.

FP에 문제가 있는것인지 FN에 문제가 있는것인지 확인하는 작업을 수행해야 한다.

1. 직접 계산하는 방법

```
Fmeasure <- 2 * precision * recall / (precision + recall)
```

2. 패키지를 이용하는 방법

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

```
F1_Score(actual_type, predict_type, positive = positive_value)
```

0~1 사이의 범위를 가지고 정밀도와 민감도(재현율)을 하나로 합친 성능평가 지표

정밀도와 민감도 양쪽이 모두 클 때 F1값도 큰 값을 가진다.

문제309. 독일은행 데이터에서 하이퍼 파라미터 trials=1, seed=31 과 trials=100, seed=659 의 F1 스코어가 어떻게 되는지 각각 출력하시오.

문제310. 다음 혼동 행렬에서 F1 스코어는 얼마인지 구하시오.

	TRUE	FALSE	합계
TRUE	60	40	100
FALSE	20	80	100
합계	80	120	200

$$\text{Precision(정밀도)} * \text{recall(재현율)}$$

$$\text{F1 SCORE} : 2 \times \frac{\text{Precision(정밀도)} * \text{recall(재현율)}}{\text{Precision(정밀도)} + \text{recall(재현율)}}$$

$$= 0.6$$

문제311. 유방암 데이터를 악성과 양성으로 분류하는 머신러닝 모델의 정확도와 다른 성능 척도를 출력하시오.

모델 성능개선을 위해 11장에서 소개하는 5가지

1. k-foldout
2. caret 을 이용한 모델 자동튜닝
3. 앙상블 - 배깅
4. 앙상블 - 부스팅
5. 앙상블 - 랜덤포레스트

k-foldout

데이터 집합을 무작위로 동령한 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 기법이다.

k-foldout 실습

k-foldout을 통해서 의사결정트리 C50 패키지의 최적의 하이퍼파라미터를 테스트 하기 전에 알아내는 목적

```
setwd("c:\\data")
#install.packages("caret")
library(caret)
credit <- read.csv("credit.csv") # 독일 은행의 채무 불이행자를 예측
nrow(credit)

# 10-fold CV
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$Fold01, ] # Fold1을 테스트 데이터로 사용
credit01_train <- credit[-folds$Fold01, ] # Fold1을 제외한 나머지 데이터를 훈련 데이터로 사용
nrow(credit01_test ) # 100
nrow(credit01_train) # 900

#전체 10폴드 교차검증을 수행하려면 이 단계는 10회 반복되어야한다.

## Automating 10-fold CV for a C5.0 Decision Tree using lapply() ---- |
```

```
library(caret)
library(C50) # 의사결정트리 모델 사용
#install.packages('irr')
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)

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)

```

lapply

함수 lapply(데이터, 함수) 함수에 데이터를 하나씩 자동으로 입력하는 함수

```

result <- lapply(1:3, function(x)x*2)

result

```

문제1. 위의 결과가 카파지수가 아닌 정확도가 출력되게 하시오.

```

library(caret)
library(C50)
library(irr)
credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
str(credit)
set.seed(123)

folds <- createFolds(credit$default, k = 10)

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.

```

library(caret)
library(C50)
library(irr)
credit <- read.csv("credit.csv", stringsAsFactors = TRUE)
str(credit)
set.seed(123)

folds <- createFolds(credit$default, k = 10)

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)

mean(unlist(cv_results))

```

문제3. 위의 데이터를 훈련 데이터75% 테스트 데이터25%로 나누고 훈련 데이터75%에 대해서만 k-foldout을 진행하고 훈련 데이터의 정확도의 평균을 확인하시오.

```

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 테스트 할 때 한번 사용
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

# credit_train으로 k-foldout을 진행
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))

```

위와 같이 훈련 데이터를 일부 분리해서 validation 데이터로 사용하는 것은 훈련 데이터를 가지고 최적의 하이퍼 파라미터를 찾아내기 위해서이다.

문제4. 하이퍼 파라미터인 seed 값은 659로하고 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))

```

문제5. 위에서 알아낸 seed 값과 trials를 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

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
```

훈련 데이터의 정확도가 0.75이고 테스트 데이터의 정확도가 0.74 이면 아주 작게 오버피팅이 일어났다.

문제6. 다음 중 k-fold cross validation에 대한 설명으로 가장 부적절한 것을 고르시오.

1. 데이터 집합을 무위로 k개의 부분집합으로 나누어 검증하는 방법이다.
2. 전체 데이터를 k개의 동일 크기로 나눈다.
3. 모든 데이터를 학습과 평가에 사용할 수 있다.
4. k값이 증가하면 수행시간과 계산량이 감소한다.

4번

성능개선_caret을 이용한 자동튜닝

<https://cafe.daum.net/oracleoracle/Sg0x/872>

caret 패키지의 역할

이전에는 머신러닝 모델의 성능을 높이기 위해서 우리가 직접 모델의 성능을 높이는 하이퍼파라

미터를 알아내야 했다. 그런데 caret을 이용하면 패키지가 알아서 최적의 파라미터를 찾아준다.

caret패키지 사용 실습1

```
# 의사결정트리 C5.0 의 하이퍼파라미터인 trials, winnow, model 의 27개의 조합에 대한  
# 각각의 정확도를 구하는 작업
```

```
library(caret)  
library(C50)  
library(irr)  
  
credit <- read.csv("credit.csv", stringsAsFactors=TRUE)  
set.seed(300)  
  
m <- train( default~ . , data=credit, method="C5.0")  
m # 튜닝한 결과를 확인할 수 있다.  
  
p <- predict( m , credit )  
table(p, credit$default)
```

문제. 위의 코드는 credit 전체코드를 한번에 사용해서 최적의 하이퍼파라미터를 찾고 credit전체코드를 한번에 사용해서 평가를 하는 코드이다. 이번에는 훈련 데이터를 75%로 하고 테스트 데이터를 25%로 나눠서 75%의 훈련 데이터에 대해서 caret을 이용한 자동 튜닝을 진행하고 만든 모델로 25%의 테스트 데이터를 평가하시오.

```
# 의사결정트리 C5.0 의 하이퍼파라미터인 trials, winnow, model 의 27개의 조합에 대한  
# 각각의 정확도를 구하는 작업
```

```
library(caret)  
library(C50)  
library(irr)  
  
credit <- read.csv("credit.csv", stringsAsFactors=TRUE)  
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)
```

```
# 정확도 확인 코드  
library(gmodels)  
y <- CrossTable(credit_test$default, p)  
sum(y$prop.tbl * diag(2))
```

winnow

<https://cafe.daum.net/oracleoracle/Sg0x/888>

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.

커스터마이징

<https://cafe.daum.net/oracleoracle/Sg0x/872>

문제. 위의 코드를 다시 수행하는데 k-fold 의 k 값을 디폴트인 25개가 아니라 30개로 변경해서 수행하면 정확도가 올라가는지 확인하시오.

```
# 의사결정트리 C5.0 의 하이퍼파라미터인 trials, winnow, model 의 27개의 조합에 대한  
# 각각의 정확도를 구하는 작업
```

```
library(caret)  
library(C50)  
library(irr)  
  
credit <- read.csv("credit.csv", stringsAsFactors=TRUE)  
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
```

```
# 커스터마이징  
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)
```

```
# 정확도 확인 코드  
library(gmodels)  
y <- CrossTable(credit_test$default, p)  
sum(y$prop.tbl * diag(2))
```

문제. k값을 40으로 하면 정확도가 더 올라가는지 확인하시오.

```
library(caret)
library(C50)
library(irr)

credit <- read.csv("credit.csv", stringsAsFactors=TRUE)
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

# 커스터마이징
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)

# 정확도 확인 코드
library(gmodels)
y <- CrossTable(credit_test$default, p)
sum(y$prop.tbl * diag(2))
```

문제7. 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 분류하였는데 이번에는 양상을 기법의 랜덤포레스트를 사용하여 분류하시오.

(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

# 랜덤포레스트 : 의사결정트리 + 양상을 기법법
m <- train( Species~ . , data=iris_train, method="rf")

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)
nrow(test_Set)

## 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.

# 아래의 코드에서 sample을 사용해서 랜덤으로 생성한 숫자값을 seeds
# 변수에 담기 때문에 지정한 seed
set.seed(123)
```

```

# 리스트 자료구조를 생성하는데 리스트의 요소를 51개로 해서 seeds라는 이름으로 생성
seeds <- vector(mode = "list", length = 51)
seeds

# for문을 이용해서 seeds 리스트의 1번부터 50번 까지의 요소안에 숫자를 22개씩
# 랜덤으로 채워넣는다.
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 = "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

```

<https://cafe.daum.net/oracleoracle/Sg0x/891>

문제2. method='adaptive_cv'로 해서 수행하고 정확도를 확인하시오.

```

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)
nrow(test_Set)

## 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.

# 아래의 코드에서 sample을 사용해서 랜덤으로 생성한 숫자값을 seeds
# 변수에 담기 때문에 지정한 seed
set.seed(123)

```

```

# 리스트 자료구조를 생성하는데 리스트의 요소를 51개로 해서 seeds라는 이름으로 생성
seeds <- vector(mode = "list", length = 51)
seeds

# for문을 이용해서 seeds 리스트의 1번부터 50번 까지의 요소안에 숫자를 22개씩
# 랜덤으로 채워넣는다.
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

```

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
nrow(train_Set)
nrow(test_Set)

## 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.

# 아래의 코드에서 sample을 사용해서 랜덤으로 생성한 숫자값을 seeds
# 변수에 담기 때문에 지정한 seed
set.seed(123)

# 리스트 자료구조를 생성하는데 리스트의 요소를 51개로 해서 seeds라는 이름으로 생성
seeds <- vector(mode = "list", length = 51)
seeds

```

```

# for문을 이용해서 seeds 리스트의 1번부터 50번 까지의 요소안에 숫자를 22개씩
# 랜덤으로 채워넣는다.
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, verbose=TRUE,
                      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 knn모델에서 k=11과 함께 최적의 seed 값이 무엇인지 구하시오.

양상블

<https://cafe.daum.net/oracleoracle/Sg0x/873>

다양한 전문가 팀을 만드는 것과 유사한 원리를 활용하는 메타 학습방법
 모든 양상블 방법은 약한 학습자 여러개를 결합하면 강한 학습자가 만들어진다는 아이디어를 기반으로 한다.

양상블 모형은 여러개의 분류 모형을 같이 사용하여 한꺼번에 평가하는 모형을 말한다.

양상블 실습1

(정확도가 60%밖에 되지 않는 분류기 모형들이 즐비한데 이 모형들을 최소한 몇개를 써야 정확도가 90% 가 되는지 확인하시오.)

```

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
}

```

앙상블의 원리

여러 모델을 이용하여 데이터를 학습하고 모든 모델의 예측 결과를 평균하여 예측한다.

앙상블의 장점

1. 다양한 모델의 결과를 종합하여 전반적으로 오류를 줄여주어 정확도는 높여준다.
2. 모델별로 다양한 bias를 종합하여 결과를 생성하게 되어 오버피팅을 줄여준다.

앙상블을 이용해서 나온 좋은 모형의 모습

1. 학습 모형의 예측 오류는 크게 bias와 variance 이다.
2. bias는 예측값과 정답과의 거리가 얼마나 떨어져 있는 거리 이다.
3. variance는 학습한 모델별로 예측한 값들의 차이 이다.
4. 앙상블을 이용한 좋은 모형이란 bias를 낮추고 variance를 최소화하는 모형이다.

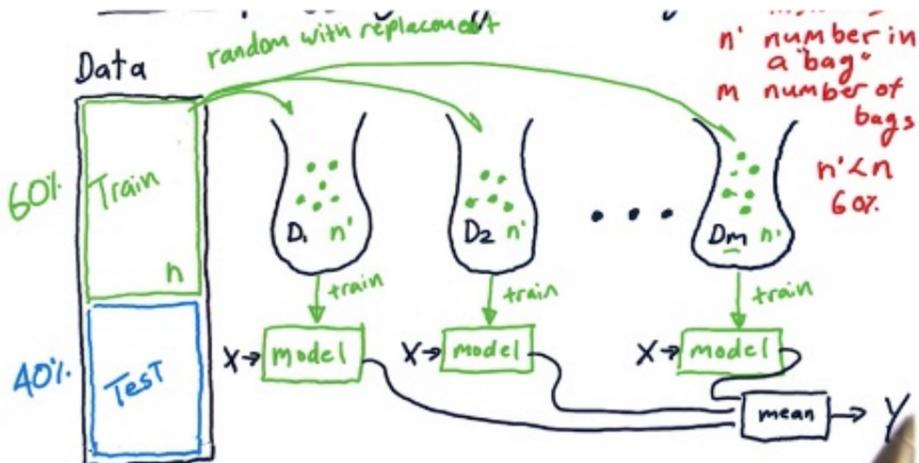
앙상블을 이용해서 오류를 줄이고 좋은 모형이 나오도록한 앙상블의 종류

1. **Bagging:** generate weak models (decision trees) from random samples and aggregate them simply

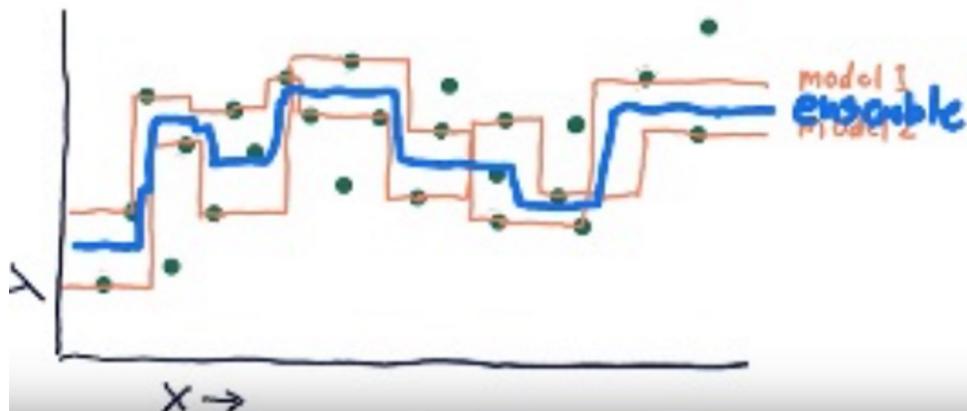
Bagging의 개념

학습데이터를 랜덤으로 샘플링하여 여러개의 bag으로 분할하고, 각 bag별로 모델을 학습한 후, 각 결과를 합하여 최종 결과를 추출

- n : 전체 학습 데이터 수
- n' : bag에 포함된 데이터 수, 전체 데이터 중 샘플링된 데이터
- m : bag의 갯수, 학습할 모델별로 샘플링된 데이터 셋



- 동일한 모델을 사용하고 데이터만 분할하여 여러개의 모델을 학습하는 양상을 기법을 말한다.
- 위의 그림에서 데이터를 샘플링할 때는 복원 추출한다.



- 위의 그림에서는 model1 하나로만 보면 하나의 bag으로만 학습된 모델이다. 각 모델별로 보면 어느 데이터에 오버피팅되어 테스트 데이터로 검증하면 예측성능이 낮다.
- 배깅은 이렇게 여러 weak model을 여러 개 결합하여 전체적으로 높은 variance를 낮은 variance로 만들면서 예측 성능을 향상시킨다.
- 위의 그림을 보면 모델끼리 노 보안하고 양보하면서 예측한다.

2. Boosting: sequentially generate weak models from previous residuals & combine them

with different weights

3. **Random Forest:** inject more randomness compared with Bagging

배깅 실습

(독일 은행 데이터로 채무 불이행자를 예측)

#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), ]
```

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
```

문제1. 위의 코드에서 bag의 갯수를 50개로 늘리고 확인하시오.

#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), ]
```

배깅으로 성능 높이기

```
#install.packages("ipred")  
library(ipred)  
set.seed(300)
```

```
mybag <- bagging( default ~ . , data=credit_train, nbagg=50)  
# 설명: nbagg=50 은 양상블에 사용되는 bag의 갯수를 50개
```

```
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
```

21.02.17

2021년 2월 17일 수요일 오전 9:46

모델 성능개선을 위해 11장에서 소개하는 5가지

1. k-foldout
2. caret 을 이용한 모델 자동튜닝
3. 양상블 - 배깅
4. 양상블 - 부스팅
5. 양상블 - 랜덤포레스트

양상블 부스팅 이론과 실습

<https://cafe.daum.net/oracleoracle/Sg0x/907>

부스팅은 배깅을 약간 개선시킨 알고리즘인데 샘플링하는 과정에서 복원 추출할 때 동일한 확률로 하는게 아니라 추출할때마다 확률을 서로 다르게 개선시키는 방법을 사용한다.

처음에는 모두 동일한 확률로 복원추출하지만 다음 추출과정에서는 오분류된 데이터를 추축확률이 더 높도록 조정하고 올바르게 분류된 데이터는 추출확률을 낮게 조정하여 복원추출한다. 이런 작업을 정해진 단계의 수만큼 반복적으로 사용한다.

배깅과 부스팅의 차이점

배깅과 부스팅은 모두 의사결정나무의 안정성을 높인다는 공통점이 있다. 둘 다 표본 추출에 있어서 데이터셋에 복원랜덤추출하지만 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 부스트는 오차가 큰 객체에 대해 더 높은 가중치를 부여한다.

부스팅실습 (독일 데이터)

#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. 부스팅으로 성능 높이기
#install.packages("adabag")
library(adabag)
set.seed(300) # 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

```

문제1. iris 데이터셋을 boosting을 이용해서 분류하시오.

```

#1. 데이터를 로드한다.
View(iris)
str(iris)

#2. 데이터에 각 컬럼들을 이해한다.
prop.table( table(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), ]

#6. 부스팅으로 성능 높이기

```

```
#install.packages("adabag")
library(adabag)
set.seed(300) # 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
```

문제2. 위의 boosting 모델의 성능 높이시오.

#1. 데이터를 로드한다.

```
View(iris)
str(iris)
```

#2. 데이터에 각 컬럼들을 이해한다.

```
prop.table( table(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), ]
```

#6. 부스팅으로 성능 높이기

```
#install.packages("adabag")
library(adabag)
set.seed(300) # 데이터를 복원추출하기 때문에 필요
m_adaboost <- boosting( Species ~ ., data=iris_train, boos=TRUE, mfinal=3)
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
```

문제3. 위의 옵션인 boos=TRUE 와 mfinal=3이 무엇인지 확인하시오.

?boosting

양상블 랜덤포레스트

<https://cafe.daum.net/oracleoracle/Sg0x/908>

의사결정트리는 오버피팅될 가능성이 높다는 약점을 가지고 있다. 가지치기를 통해 최대 높이를 조정해서 오버피팅 될 가능성을 낮출수는 있지만 이것만드로는 오버피팅 문제를 충분히 해결할 수 없다. 그래서 의사결정트리에 양상블 기법을 추가해서 훈련 데이터를 여러 의사결정트리 모델들이 샘플링하여 학습하고 예측결과를 투표하여 가장 많이 득표한 결과를 최종 분류의 결과로 선택한다.

양상블 랜덤포레스트 수행 과정

랜덤포레스트는 제일 먼저 bagging 이라는 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.

랜덤포레스트의 파이퍼 파라미터

1. ntree : 몇개의 나무를 생설할지를 설정하는 인자
2. mtry : 각 노드에서 랜덤하게 고려될 변수의 갯수를 지정
3. nodesize : 나무의 깊이를 설정하는 인자로 최소한의 노드의 갯수를 뜻한다. 이 값이 크면 깊이가 얕은 나무가 생성되고 이 값이 작으면 얕은 나무가 생성된다.

양상블 랜덤포레스트 실습1

<https://cafe.daum.net/oracleoracle/Sg0x/903>

random forest 는 decesion tree 와 bagging 을 결합한 알고리즘이다.

실습 :

kyphosis(후만증)은 척추의 비정상적으로 과도한 볼록 곡률입니다.

후만증 데이터 프레임에는 81 개의 행과 4 개의 열이 있습니다.

```

# 척추 교정 수술을 받은 어린이에 대한 데이터를 나타냅니다.
# 데이터 세트는 3 개의 입력과 1 개의 출력을 포함합니다.

# 데이터로서 독립변수가 나이(개월수)와 관련된 척추의 수 그리고 수술 된
# 첫 번째 (최상위) 척추의 수이다.

# 데이터가 81개 밖에 안되므로 데이터를 전부 의사 결정트리에 넣어서 예측과
# 실제 라벨을 비교해서 정확도를 봅니다.

# 0. 필요한 패키지를 다운로드 합니다.
install.packages("rpart")
install.packages("rattle")
install.packages("randomForest")

# 1. 데이터를 로드한다.
kyphosis <- read.csv("kyphosis.csv", stringsAsFactors = TRUE)

# 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)
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)

# [1] 0.8395062
# 그럼 83% 로 나온다.

# 이 수치를 높이기 위해서 random forest 를 사용한다.

# 1. 랜덤 포레스트 모델을 만든다.
library(randomForest)

# 랜덤포레스트 패키지 설명 :
# https://www.rdocumentation.org/packages/randomForest/versions/4.6-14/topics/randomForest

fit <- randomForest(kyphosis ~ age + number + start, data=kyphosis)
res2 <- predict(fit , newdata = kyphosis)
sum(res2 == kyphosis$kyphosis)/NROW(kyphosis)

# [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.
```

```
str(fit)
```

문제1. 척추 데이터의 랜덤포레스트 모델을 다시 생성하는데 mtry의 갯수를 늘려서 테스트 하시오.

```
# 0. 필요한 패키지를 다운로드 합니다.
```

```
#install.packages("rpart")  
#install.packages("rattle")  
#install.packages("randomForest")
```

```
# 1. 데이터를 로드한다.
```

```
kyphosis <- read.csv("kyphosis.csv", stringsAsFactors = TRUE)
```

```
# 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)  
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)
```

```
# [1] 0.8395062
```

```
# 이 수치를 높이기 위해서 random forest 를 사용한다.
```

```
# 1. 랜덤 포레스트 모델을 만든다.
```

```
library(randomForest)
```

```
fit <- randomForest(kyphosis ~ age + number + start, data=kyphosis, mtry=3)  
res2 <- predict(fit , newdata = kyphosis)  
sum(res2 == kyphosis$kyphosis)/NROW(kyphosis)
```

```
# [1] 0.9876543 <--- 98% 로 정확도가 올라간다.
```

```
str(fit)
```

양상블 랜덤포레스트 실습2

<https://cafe.daum.net/oracleoracle/Sg0x/909>

```
# 필요한 패키지 다운로드
```

```

#install.packages('randomForest')
library(randomForest)

# 0. shuffle
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)
nrow(iris_test)

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=0.5,fill=1:4)

```

```

# 최적의 파라미터를 찾는 loop문
ntree <- c(600, 700, 800)
mtry <- c(2:4)
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)
  }
}

```

양상블과 자동 튜닝을 결합하여 최적의 모델 찾기

<https://cafe.daum.net/oracleoracle/Sg0x/928>

```

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. 훈련 데이터 75%, 테스트 데이터 25%로 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

```

2. 랜덤포레스트를 이용해서 훈련을 시키는데 자동파라미터 튜닝도 같이 진행

```
m <- train( Species~ ., data=iris_train, method="rf" )
```

랜덤포레스트: 의사결정트리 + 양상블 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))
```

서포트 벡터 머신

<https://cafe.daum.net/oracleoracle/Sg0x/921>

서포트 벡터 머신(support vector machine)은 기계학습의 분야중 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.

서포트 벡터 머신의 결정경계

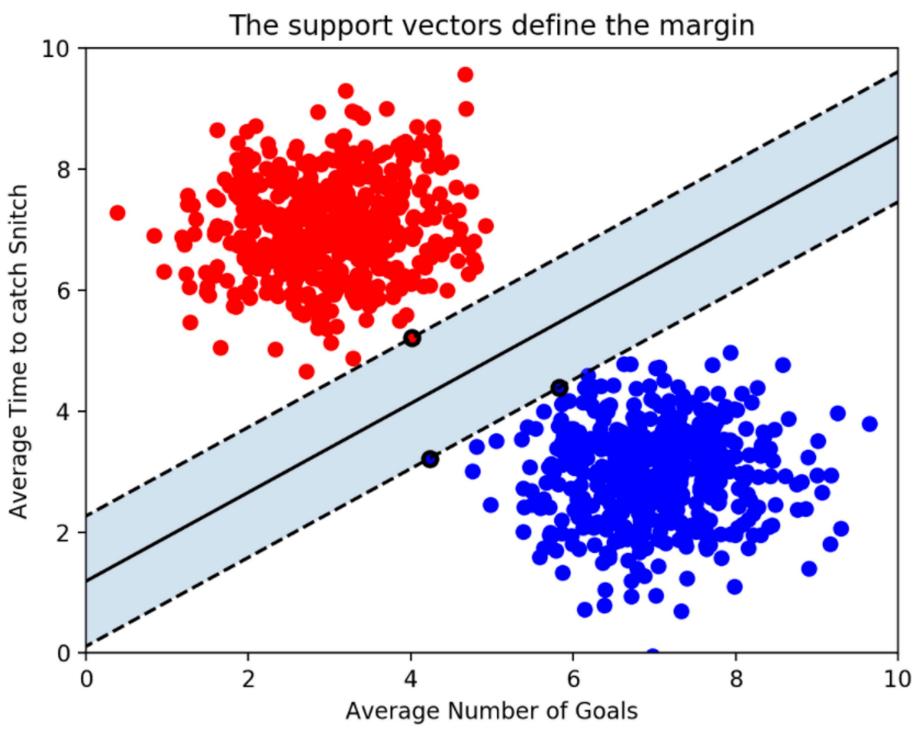
결정경계란 분류를 위한 기준선이다. 2차원일때는 선이고 3차원일때는 평면이 된다. 우리가 생각할 수 있는 부분은 3차원까지이고 차원이 더 많아지게 되면 평면이 아니라 초평면이 된다.

두클래스(분류) 사이의 거리가 가장 먼 선이 좋은 결정경계이다.

서포트 벡터

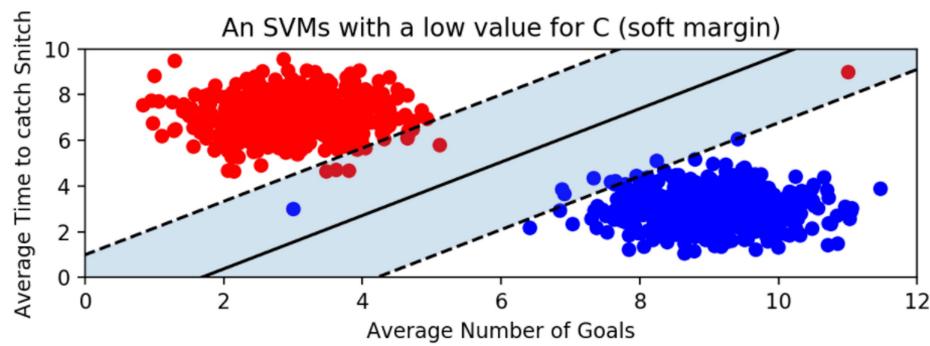
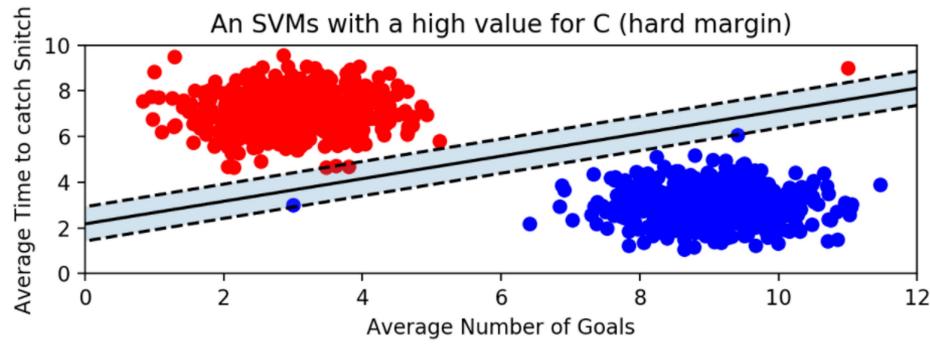
결정경계와 가까이 있는 데이터 포인트들을 의미한다. 이 데이터들이 경계를 정의하는 결정적인 역할을 한다.

마진



점선으로 부터 결정경계까지의 거리
최적의 결정경계는 마진을 최대화 한다.

소프트마진 과 하드마진



위의 그림중에 위쪽 그림은 아웃라이어를 허용하지 않고 기준을 까다롭게 세운 모델이며 이걸 하드 마진이라고 한다. 서포트 벡터와 결정경계사이의 거리가 매우 좁다. 즉 마진이 작아진다. 그러면 오버피팅 문제가 발생할 수 있다.

아래쪽 그림은 아웃라이어들이 마진안에 어느정도 포함되도록 기준을 잡았으며 이걸 소프트 마진이라고 한다. 그림과 같이 서포트 벡터와 결정경계 사이의 거리가 멀어진다. 대신 너무 대충

학습하는 꼴이라 언더피팅 문제가 발생할 수 있다.

하이퍼 파라미터

c가 클수록 하드마진(오류허용 x)이 일어나고 c가 작을수록 소프트마진이 일어난다.

gamma는 결정경계를 얼마나 유연하게 그을 것인지 정해주는 것이다. gamma를 높이면 학습 데이터에 많이 의존을 해서 오버피팅을 초래할 수 있고 gamma를 낮추면 학습 데이터에 별로 의존하지 않아 언더피팅이 발생할 수 있다.

서포트 벡터 머신 분류 확인

<https://cafe.daum.net/oracleoracle/SZTZ/2169>

서포트 벡터 머신 원리 설명

<https://cafe.daum.net/oracleoracle/Sg0x/923>

법선벡터란 한 평면이나 직선에 대해서 수직인 벡터를 말한다.

서포트 벡터 머신 실습

<https://cafe.daum.net/oracleoracle/Sg0x/925>

```
#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)
```

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 , kenel="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")
library(e1071)
mymodel <- svm(Species~. , data=iris , kenel="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)

```

문제. 미국 대합원 입학 데이터(binary.csv)를 서포트 벡터 머신으로 분류하고 정확도를 확인하시오.

21.02.18

2021년 2월 18일 목요일 오전 9:41

서포트 벡터 머신 원리

<https://cafe.daum.net/oracleoracle/Sg0x/923>

서포트 벡터 머신과 로지스틱 회귀와의 차이

- 로지스틱 회귀

<https://cafe.daum.net/oracleoracle/Sg0x/931>

```
# Logistic Regression
```

```
# Read data file
mydata <- read.csv("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) # 0.29

# Goodness-of-fit test
with(mymodel, pchisq(null.deviance - deviance, df.null-df.residual, lower.tail = F))

```

- 서포트 벡터 머신 (비선형)

```

####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-fold 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 & Labedata
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) )
```

문제. 위에서 실습한 서포트 벡터 머신의 커널을 다른 커널로 변경해서 실험하시오.

1. svmPoly
2. sigmoid
3. svmLinear
4. svmRadialCost

서포트 벡터 머신으로 필기체 데이터 분류

필기체 데이터는 딥러닝때 배울 mnist 데이터인데 모두 숫자로 되어있는 데이터이다.

<https://cafe.daum.net/oracleoracle/SeFi/596>

<https://cafe.daum.net/oracleoracle/Sedp/74>

mnist 데이터는 필기체 숫자 0~9번 까지 총 10개의 클래스로 되어있고 1클래스당 6000개의 필기체 데이터를 포함하고 있다.

하나의 필기체는 784개의 픽셀로 되어있다.

전부 숫자로만 되어있는 데이터여서 신경망이나 서포트 벡터 머신을 이용해서 분류할 때 유리한 데이터이다.

```
install.packages("caret")
install.packages("doParallel")
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 # 병렬처리를 6개의 프로세서가 동시에 수행
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 함수는 mnist 데이터를 R studio로 로드하기 위한 함수
# show_digit 함수는 필기체 데이터를 하나 시각화하기 위한 함수
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, ...)
}

# 훈련 데이터와 테스트 데이터를 구성하기 위한 데이터 프레임 생성
train <- data.frame()
test <- data.frame()

# Load data.
load_mnist()

length(train$x)
length(train$y) # 훈련 데이터 60000개
length(test$y) # 테스트 데이터 10000개

# 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)

inTrain$y <- as.factor(inTrain$y)
str(inTrain)

# 훈련 데이터 60000개를 6:4로 나눠서 60%는 훈련시킬때 쓰고 40%는 테스트
# 할 때 쓰기 위해 나눈다.
trainIndex = createDataPartition(inTrain$y, p = 0.60,list=FALSE)
training = inTrain[trainIndex,]
cv = inTrain[-trainIndex,]
nrow(training)
nrow(cv)

# 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 예측
results <- predict(fit, newdata = head(cv, 1000))
results

confusionMatrix(results, head(cv$y, 1000))

# 훈련 데이터 중 5번째 숫자를 시각화 해서 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 102번째 훈련 데이터를 넣고 무엇으로 예측하는지 확인하시오.

위의 코드

```

:
:
```

```

# 훈련 데이터 중 5번째 숫자를 시각화 해서 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]

```

kaggle_타이타닉 데이터

<https://cafe.daum.net/oracleoracle/Sg0x/946>

<https://cafe.daum.net/oracleoracle/Sg0x/948>

1. 파이썬에서 아래와 같이 seaborn에 내장되어 있는 타이타닉 데이터를 내려받습니다.

```
import seaborn as sns
import pandas as pd

tat = sns.load_dataset('titanic')

print(tat)

tat2 = pd.DataFrame(tat)

tat2.to_csv('d:\\\\data\\\\tatanic.csv', sep=',', na_rep='NaN') # missing data representation (결측값 표기)
```

2. 내려받은 타이타닉 데이터를 R로 로드합니다.

```
# 타이타닉 생존자 분류
# 1. 데이터 로드
tat <- read.csv("tatanic.csv", stringsAsFactors = TRUE)
View(tat)
head(tat)

#survived : 생존=1, 죽음=0
#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)

#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)

```

#0.shuffle 을 먼저 합니다.

```

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 )
round(p)

```

```
table(round(p), tat_test$survived)
```

```
library(gmodels)
y <- CrossTable(tat_test$survived ,round(p) )
sum(y$prop.tbl * diag(2))
```

문제. 나이의 결측치를 나이의 평균값으로 하지 말고 승객나이중에 최대값으로 결측치를 채우고 다시 테스트 데이터의 정확도를 확인하시오.

```
# 나이의 결측치를 나이의 최대값으로 치환
tat$age[is.na(tat$age)] <- max(tat$age,na.rm=TRUE)
```

kaggle_보스톤 하우징 데이터

1. 파이썬의 보스톤 하우징 파일 내리기

```
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)
df['price'] = boston.target

df.to_csv('c:\\\\data\\\\boston.csv', sep=',', na_rep='NaN')
# missing data representation (결측값 표기)
```

2. Rstudio로 데이터 불러오기

```
##### 1. 데이터 불러오기 #####
getwd()
library(data.table)

boston <- read.csv("c:\\\\data\\\\boston.csv" )
nrow(boston)
```

```

#0.shuffle 을 먼저 합니다.
set.seed(123)
boston_shuffle <- sample(1:506, 506)
boston_shuffle
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)
str(boston_train)
str(boston_test)

# boston_train와 boston_test의 컬럼명을 소문자로 변경
colnames(boston_train) <- tolower(colnames(boston_train))
colnames(boston_test) <- tolower(colnames(boston_test))

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) # NA중앙값 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) > 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(price~., 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)
```

```
summary(model1)
```

```
#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)
```

```
lm12<-step(lm1, direction="both")
```

```
# price ~ 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)

```
# max(train_data$medv) # min(train_data$medv)  
denormalize <- function(x) { return ( x*45+5 ) }  
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)
```

```
head(sample)
```

```
boston_test$price
```

```
str(boston_test$price)  
str(sample)
```

```
cor (boston_test$price, sample$medv)
```

문제. 상관계가 아니라 오차가 나오게 하시오.

21.02.19

2021년 2월 19일 금요일 오전 9:45

복습

1. 서포트 벡터머신의 오차함수가 어떻게 만들어졌는지 원리를 설명
2. 로지스틱 회귀와 서포트 벡터 머신의 차이
3. 서포트 벡터머신의 성능을 개선하기 위해 caret패키지를 이용하는 방법
4. 서포트 벡터머신을 이용해서 필기체 데이터 분류
5. 타이타닉과 보스톤 하우징 데이터를 파이썬에 내장되어 있는 데이터를 R로 가져옴
6. 타이타닉 데이터를 랜덤포레스트로 분류
7. 보스톤 하우징 데이터 회귀모델 생성

보스톤 하우징 데이터 회귀분석과 랜덤포레스트 예측결과 비교

<https://cafe.daum.net/oracleoracle/Sg0x/963>

- 회귀분석

```
##### 1. 데이터 불러오기 #####
```

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

```
boston <- read.csv("c:\\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)  
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) # NA중앙값 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

# strict 하게 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
# [중요] method에 따른 머신러닝 알고리즘을 적용할 때 참고
# https://cafe.daum.net/oracleoracle/Sg0x/965

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)

```

- 양상불 랜덤포레스트

```

##### 1. 데이터 불러오기 #####
getwd()
library(data.table)

boston <- read.csv("c:\\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)
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) # NA중앙값 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
# [중요] method에 따른 머신러닝 알고리즘을 적용할 때 참고
# https://cafe.daum.net/oracleoracle/Sg0x/965

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)
```

보스톤 하우징 데이터에 파생변수를 추가하고 양상을 학습

```

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 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

# strict 하게 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
# [중요] method에 따른 머신러닝 알고리즘을 적용할 때 참고
# https://cafe.daum.net/oracleoracle/Sg0x/965

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)

```

파생변수 추가 전

모델	회귀	랜덤포레스트
상관관계	0.8089203	0.8986854
오차	5.537822	3.245476

파생변수 추가 후

모델	회귀	랜덤포레스트
상관관계	0.8272225	0.8997801
오차	5.441164	3.199271

문제312. 위의 코드에서 파생변수를 새로 추가한 후의 상관관계와 오차를 확인하시오.

파생변수의 영향력 확인

<https://cafe.daum.net/oracleoracle/Sg0x/964>

종속변수를 설명할 독립변수를 선택하는 방법에 있어서 가장 많이 쓰는 방법이 단계적 변수 선

택 방법이다. 이걸 지원하는 R 함수가 step 함수이다.

```
##### 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)
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 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

# strict 하게 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. 상관관계를 감
안하여 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(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)

```

문제313. 파생변수를 추가해서 step 함수 사용 결과를 출력하시오.

kaggle last competition (보스톤)

- R

kaggle ongoing competition (타이타닉)

- SQL
- R

kaggle ongoing competition (보스톤)

- R

kaggle last competition (보스톤)



boston_2...

kaggle ongoing competition (타이타닉)

- SQL

<https://cafe.daum.net/oracleoracle/Sg0x/969>



부록_예제
_1

정확도를 올리기 위한 3가지 tip

1. 결측치를 다른값으로 치환
 - a. 호칭으로 나이를 예상해서 나이의 결측치를 채워 넣는다.
 - b. 나이에 대한 회귀 예측값으로 결측치를 채워 넣는다.
2. 이상치를 제거 또는 치환
3. 파생변수를 생성
 - a. 여자와 아이를 구분하기 위한 파생변수를 생성
 - b. 나이의 결측치를 이름의 호칭의 평균값으로 채워 넣는다.



부록_예제
_2

문제314. 위의 결과에서 파생변수를 변경하거나 결측치를 다른 값으로 치환해서 순위를 올리시오.