데이터 마이닝 중간고사대비 실습

```
# 패키지 설치 /지에스에서 알엑스만남
install.packages("ggplot2")
install.packages("GGally")
install.packages("sqldf")
install.packages("stringr")
install.packages("reshape2")
install.packages("xlsx")

# 패키지 실행
* 여기서는 ""안씀에 주의하기
library(ggplot2)
library(sqldf)
library(stringr)
library(reshape2)
library(GGally)
library(xlsx)

# working directory 설정
setwd("C:/User/user/Desktop/datamining")
```

BostonHousing 실습

CHAS 찰스강에 접해있는지 (접해있으면 1 or 0)

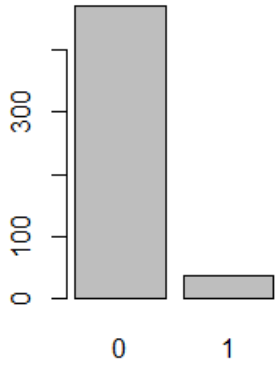
LSTAT 저소득층 비율

MEDV 자기소유 주택가격들의 중앙값

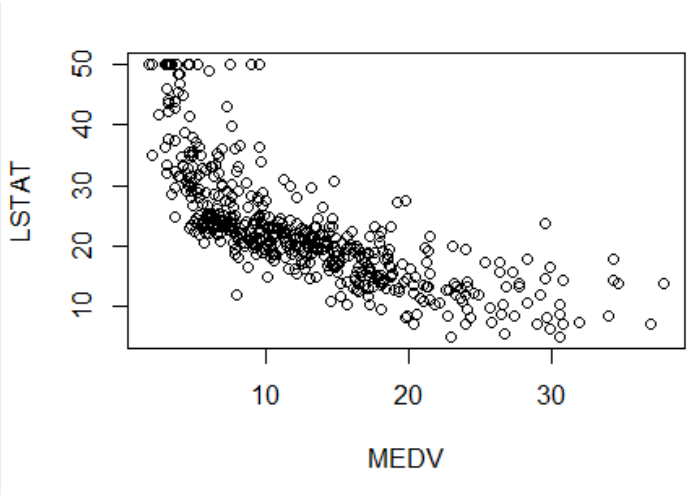
CAT.MEDV medv가 >30이면 1, 아니면 0

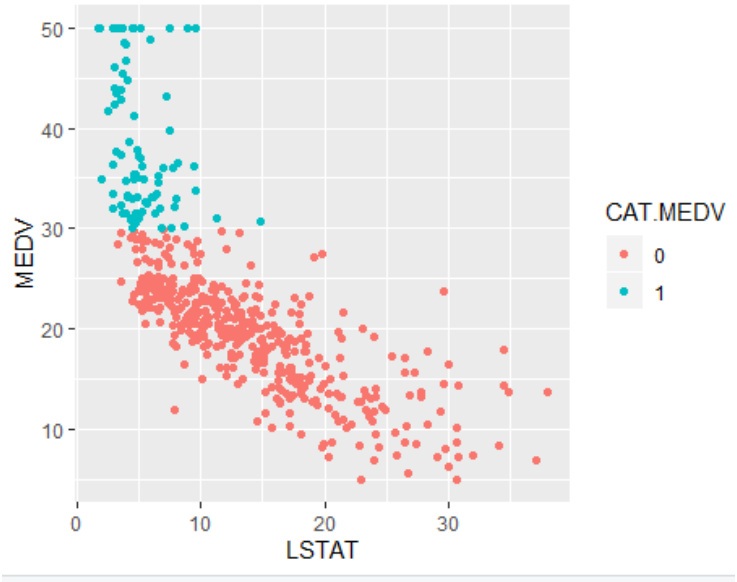
```
# 데이터 불러오기
## 엑셀로 불러오려면
## data.housing <- read.xlsx("Bostonhousing.xlsx", sheetName="Data")
data.housing <- read.csv("Bostonhousing.csv")
summary(data.housing)

# 데이터 칼럼 이름 바꾸기
## names(data.housing)[14] <- "CAT.MEDV" 도 가능
names(data.housing)[14] <- c("CAT.MEDV")
* 1부터 시작함

# CAT.MEDV type바꾸기 int -> factor
data.housing$CAT.MEDV <- as.factor(data.housing$CAT.MEDV)

* barplot그리기
1. table로 그리기
# CHAS barplot 그리기
count <- table(data.housing$CHAS)
barplot(count)
```

	Var1	Freq
1	0	471
2	1	35




```
2. ggplot2로 barplot그리기
# count를 data.frame으로 바꾸기(바꿔도 위에 테이블이랑 형태는 같더라)
count.df <- as.data.frame(count)

# CHAS강에 접해있는 사람 더하기
total <- 0
for(i in 1:length(count.df[,2])) total <- total+count.df[,2]

# column 추가
new_col <- data.frame(ratio=c(count.df[,2]/total))
count.df <- cbind(count.df, new_col)

# CHAS dataframe으로 barplot그리기
p <- ggplot(data=count.df, aes(x=var1, y=ratio))
p #형태만 만든거
p + geom_bar(stat="identity")+xlab('CHAS')+ylab('% of CHAS')

# LSTAT와 MEDV 관계 point찍기
1. plot사용하기
plot(data.housing$LSTAT, data.housing$MEDV, xlab="MEDV", ylab="LSTAT")

2. ggplot사용하기
p <- ggplot(data=data.housing, aes(x=LSTAT, y=MEDV, color=CAT.MEDV))+geom_point()
p
```






```
# 과제 : MEDV히스토그램과 박스플롯을 기본차트와 ggplot2를 이용해서 각각 그려보기
# MEDV히스토그램그리기(histogram에서 세로축은 frequency임)
1. hist이용
h <- hist(data.housing$MEDV, xlim=range(data.housing$MEDV), main="Histogram for MEDV", xlab="MEDV")

2. geom_histogram 이용(ggplot2이용)
p <- ggplot(data=data.housing, aes(x=MEDV))
p + geom_histogram(binwidth=5, color="black", fill="white")
```

```
# boxplot그리기
1. boxplot이용
## 형태는 boxplot(y~x)
b <- boxplot(data.housing$MEDV~data.housing$CHAS, main="CHAS와 MEDV", xlab="CHAS", ylab="MEDV")

2. x축이 되는 변수가 요인변수이면 자동으로 boxplot생성
data.housing$CHAS <- as.factor(data.housing$CHAS)
plot(data.housing$CHAS, data.housing$MEDV, xlab="CHAS", ylab="MEDV")

3. ggplot2이용
data.housing$CHAS <- as.factor(data.housing$CHAS)
ggplot(data=data.housing, aes(x=CHAS, y=MEDV))+geom_boxplot()
```

상관관계 분석

```
#상관관계 테이블
cor_mat <- round(cor(data.housing),2) #소수둘째자리까지 끊기
cor_mat[upper.tri(cor_mat)] <- NA #upper tri란 윗쪽 삼각형에 NA를 넣어라
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE
CRIM	1.00	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
ZN	-0.20	1.00	*NA*	*NA*	*NA*	*NA*	*NA*
INDUS	0.41	-0.53	1.00	*NA*	*NA*	*NA*	*NA*
CHAS	-0.06	-0.04	0.06	1.00	*NA*	*NA*	*NA*
NOX	0.42	-0.52	0.76	0.09	1.00	*NA*	*NA*
RM	-0.22	0.31	-0.39	0.09	-0.30	1.00	*NA*
AGE	0.35	-0.57	0.64	0.09	0.73	-0.24	1.00
DIS	-0.38	0.66	-0.71	-0.10	-0.77	0.21	-0.75
RAD	0.63	-0.31	0.60	-0.01	0.61	-0.21	0.46

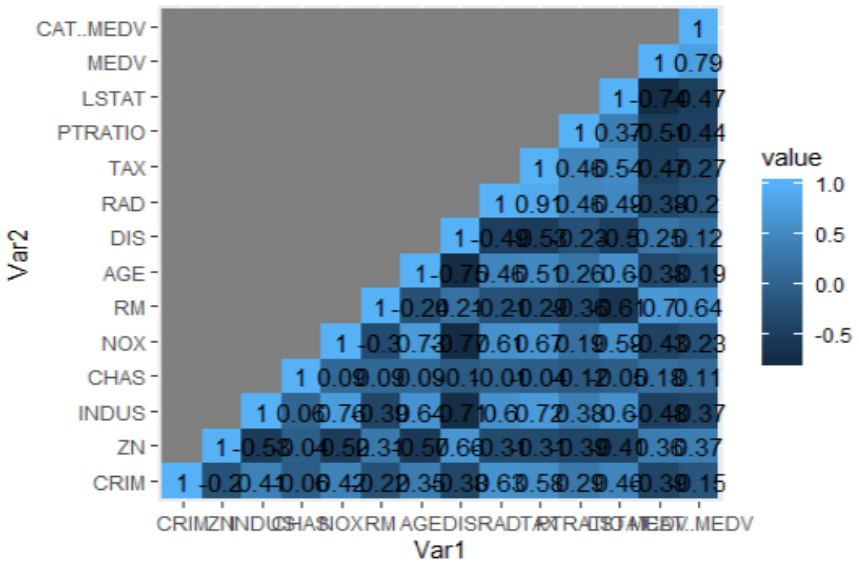
```
cor_mat_melt <- melt(cor_mat) #일자형태로 만들기
```

datamining1.R* × cor_mat_melt ×

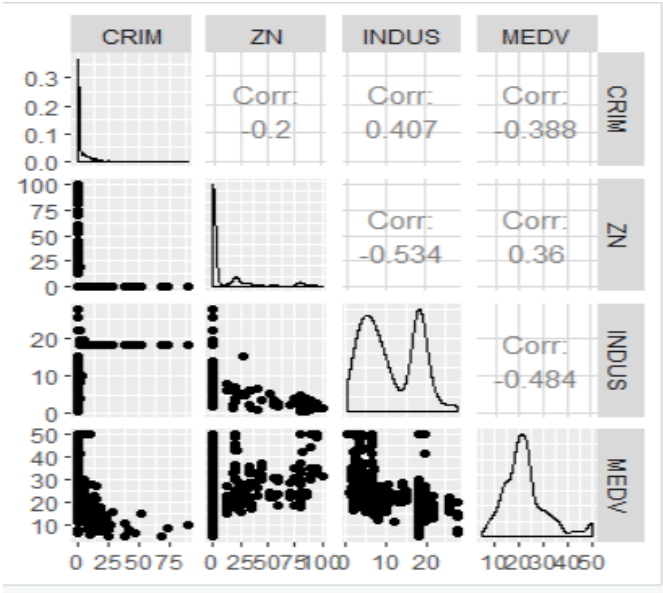
Filter

	Var1	Var2	value
1	CRIM	CRIM	1.00
2	ZN	CRIM	-0.20
3	INDUS	CRIM	0.41
4	CHAS	CRIM	-0.06
5	NOX	CRIM	0.42
6	RM	CRIM	-0.22
7	AGE	CRIM	0.35
8	DIS	CRIM	-0.38
9	RAD	CRIM	0.63
10	TAX	CRIM	0.58
11	PTRATIO	CRIM	0.29

```
g <- ggplot(data=cor_mat_melt, aes(x=Var1, y=Var2, fill=value))+geom_tile()+geom_text(aes(label=value))
g
```



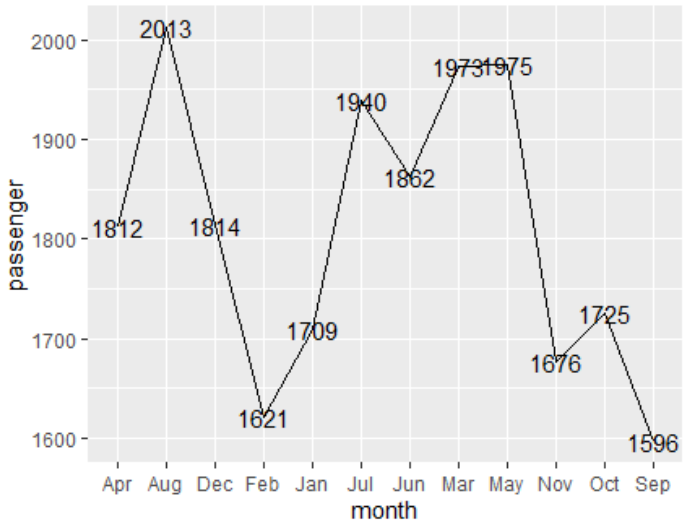

```
# datahousing의 1,2,3,13 총4개에 대한 그래프 상관관계, 분포도(커널밀도함수)
ggpairs(data.housing[, c(1:3, 13)])
```



Army데이터 실습

```
# 파일 불러오기
am_data <- read.csv("AmtrakPassengersMonthly T-competition2.csv")
summary(am_data)

# 이름바꾸기
names(am_data)[1] <- "date"
names(am_data)[2] <- "passenger"

# data형태 바꾸기
X <- NULL
X <- gsub("-", " ", am_data$date) # gsub(pattern, replace, string) string에 있는 pattern을 replace해라ㄹ
```

```
X <- strsplit(X, " ") # 띄어쓰기를 기준으로 나눔
X <- t(matrix(unlist(X), 3, 159)) #3개의 열, 159개 행

# X를 data frame으로 바꾸기
X_frame <- data.frame(X)
names(X_frame)[1] <- "day"
names(X_frame)[2] <- "month"
names(X_frame)[3] <- "year"

# am_data에 X_frame합치기
amtrak_data <- cbind(am_data, X_frame)

# sql문은 테이블 형태의 데이터를 다룰때 사용
# sqldf를 사용하면 데이터 전처리를 빠르게 할 수 있음
# 월별, 연도별로 그룹화해서 승객수 더한 표
am_month_traffic <- sqldf("select month, passenger from amtrak_data group by month")
am_year_traffic <- sqldf("select year, passenger from amtrak_data group by year")

# 월별 승객수
g <- ggplot(data=am_month_traffic, aes(x=am_month_traffic$month, y=am_month_traffic$passenger, group=1))
g+geom_line()+xlab("month")+ylab("passenger")+labs(title="월별 승객수")+geom_text(aes(label=passenger))

# 연도별 승객수
g <- ggplot(data=am_year_traffic, aes(x=am_year_traffic$year, y=am_year_traffic$passenger, group=1))
g+geom_line()+xlab("year")+ylab("passenger")+labs(title="연도별 승객수")+geom_text(aes(label=passenger))
```



Cereals실습

cereals종류가 있고, 평점(rating)이 있음

```
# 파일 불러오기
data.cereals <- read.csv("Cereals.csv")
summary(data.cereals)

# rating의 분산 구하기
var(data.cereals$rating)

# calories와 rating의 상관계수 구하기
cor(data.cereals$calories, data.cereals$rating)

# calories와 rating의 공분산구하기
```

```
cov_cereals <- cov(data.cereals[,c("calories", "rating")])
cov_cereals
```

```
            calories    rating
calories  379.6309 -188.6816
rating   -188.6816  197.3263
```

- 공분산과 상관계수

Cov(X,Y) 에서 X값이 증가할때 Y값도 증가하면 공분산이 양의 상관관계를 가짐

상관계수는 공분산을 각각의 표준편차로 나눈 값, 표준화된 공분산

```
# - 가 없으면 1:3까지 -붙으면 그거빼고
c_data4PCA <- data.cereals[,-c(1:3)] #1,2,3행뺌
View(c_data4PCA)

# missing data 삭제
c_data4PCA <- na.omit(c_data4PCA) #NA로 되있던 데이터가 사라짐

# 주성분분석
c_PCA <- prcomp(c_data4PCA)
c_PCA
summary(c_PCA)
```

```
                PC1           PC2           PC3           PC4          PC5
calories  0.0779841812  0.0093115874 -0.6292057595 -0.6010214629  0.454958508
protein  -0.0007567806 -0.0088010282 -0.0010261160  0.0031999095  0.056175970
fat      -0.0001017834 -0.0026991522 -0.0161957859 -0.0252622140 -0.016098458
sodium    0.9802145422 -0.1408957901  0.1359018583 -0.0009680741  0.013948118
fiber    -0.0054127550 -0.0306807512  0.0181910456  0.0204721894  0.013605026
carbo     0.0172462607  0.0167832981 -0.0173699816  0.0259482087  0.349266966
sugars    0.0029888631  0.0002534853 -0.0977049979 -0.1154809105 -0.299066459
potass   -0.1349000039 -0.9865619808 -0.0367824989 -0.0421757390 -0.047150529
vitamins  0.0942933187 -0.0167288404 -0.6919777623  0.7141179984 -0.037008623
shelf    -0.0015414195 -0.0043603994 -0.0124888415  0.0056471836 -0.007876459
```

summary를 실행하면 전체분산 중 각 주성분의 설명하는 비율(proportion of variance)을 보여줌
이걸 보고 필요한 성분 갯수만큼 선택해서 사용!

PC1은 전체 분산의 53%를 설명, PC2는 38% 설명, PC1과 PC2가 총 91%를 설명함

```
Importance of components:
                          PC1     PC2      PC3      PC4     PC5     PC6     PC7
Standard deviation     83.7641 70.9143 22.64375 19.18148 8.42323 2.09167 1.69942
Proportion of Variance  0.5395  0.3867  0.03943  0.02829 0.00546 0.00034 0.00022
Cumulative Proportion   0.5395  0.9262  0.96560  0.99389 0.99935 0.99968 0.99991
                          PC8     PC9    PC10   PC11    PC12      PC13
Standard deviation     0.77963 0.65783 0.37043 0.1864 0.06302 5.334e-08
Proportion of Variance 0.00005 0.00003 0.00001 0.0000 0.00000 0.000e+00
Cumulative Proportion  0.99995 0.99999 1.00000 1.0000 1.00000 1.000e+00
>
```

```
# 정규화해서 주성분분석
normalize <- function(x){
  return ((x-mean(x))/sd(x))
}
```

```
# lapply(data, function, na.rm=T) na.rm=T은 NA값 제거할거다
# 실행결과로 list를 반환함
c_data4PCA_nor <- as.data.frame(lapply(c_data4PCA,normalize))
c_PCA_nor <- prcomp(c_data4PCA_nor)
c_PCA_nor
```

- **주성분분석 prcomp(principle component analysis)**

독립변수들의 선형조합
목적 ; 차원의 축소, 예측력 향상

PC1 독립변수들의 전체 분산을 가장 많이 설명하는 성분
PC2 첫번째 주성분과 주식인 주성분(서로 독립적인 관계)

선형조합은 나온값에 원래 변수들을 곱해서 새로운 PC1이라는 변수를 생성하는 것
PC1 = 변수1 * 나온 값 + 변수2 * 나온값 + ...

주성분분석을 표준화해서 하는이유는??