기말고사 대체과제

R 활용한 데이터 분석

mpg 데이터, star 데이터

2023011742 박민준

목차

1. mpg 데이터 분석

2. star 데이터 분석

3. 보완점

1. mpg 데이터 분석

1-1 데이터 의미와 구조

ggplot2 패키지 안에 있는 mpg 데이터셋 불러오기

- library(ggplot2)

#help()함수 또는 ? 활용

- ?mpg

>1999년부터 2008년까지의 38개의 모델 자동차의 연비 데이터이고 234개의 행과 11개의 변수를 가지고 있다.

manufacturer: 제조사 명 (캐릭터형)

model: 모델명 (캐릭터형)

displ: 엔진 배기량 (더블형-실수형 중 정수,소수 모두 가짐)

year: 제조 연도 (정수형)

cyl: 실린더 수 (정수형)

trans: 변속기 유형 (캐릭터형)

drv: 구동 방식 (f = 전륜구동, r = 후륜구동, 4 = 사륜구동) (캐릭터형)

cty: 도시 연비 (마일/갤런) (정수형)

hwy: 고속도로 연비 (마일/갤런) (정수형)

fl: 연료 유형 (캐릭터형)

class: 자동차 유형 (캐릭터형)

구조확인

```
> str(mpg)
tibble [234 × 11] (S3: tbl_df/tbl/data.frame)
 $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
 $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
 $ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
 $ year        : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999
2008 ...
 $ cyl         : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...
```

```
 $ trans      : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)"
"auto(av)" ...
 $ drv        : chr [1:234] "f" "f" "f" "f" ...
 $ cty        : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
 $ hwy        : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...
 $ fl         : chr [1:234] "p" "p" "p" "p" ...
 $ class      : chr [1:234] "compact" "compact" "compact" "compact" ...
```

문자형: manufacturer,model,trans,drv,fl,class

수치형: displ,year,cyl,cty,hwy

```
> summary(mpg[,c(3,4,5,8,9)]) #범주형 자료 제외
     displ             year            cyl             cty             hwy
 Min.   :1.600   Min.   :1999   Min.   :4.000   Min.   : 9.00   Min.   :12.00
 1st Qu.:2.400   1st Qu.:1999   1st Qu.:4.000   1st Qu.:14.00   1st Qu.:18.00
 Median :3.300   Median :2004   Median :6.000   Median :17.00   Median :24.00
 Mean   :3.472   Mean   :2004   Mean   :5.889   Mean   :16.86   Mean   :23.44
 3rd Qu.:4.600   3rd Qu.:2008   3rd Qu.:8.000   3rd Qu.:19.00   3rd Qu.:27.00
 Max.   :7.000   Max.   :2008   Max.   :8.000   Max.   :35.00   Max.   :44.00
```

1-2 범주형, 양적 자료 분포와 시각화

범주형 자료의 빈도수 파악 table()함수 사용

- table(mpg$manufacturer)
- table(mpg$model)
- table(mpg$trans)
- table(mpg$drv)
- table(mpg$fl)

```
> table(mpg$fl)

  c   d   e   p   r
  1   5   8  52 168
```

- table(mpg$class)

변수들 간 교차 빈도 파악

- table(mpg$manufacturer,mpg$drv)

```
> table(mpg$manufacturer,mpg$drv)#제조사별 구동 방식 빈도

           4  f  r
audi      11  7  0
chevrolet  4  5 10
dodge     26 11  0
ford      13  0 12
```

```
honda       0  9  0
hyundai     0 14  0
jeep        8  0  0
land rover  4  0  0
lincoln     0  0  3
mercury     4  0  0
nissan      4  9  0
pontiac     0  5  0
subaru     14  0  0
toyota     15 19  0
volkswagen  0 27  0
```

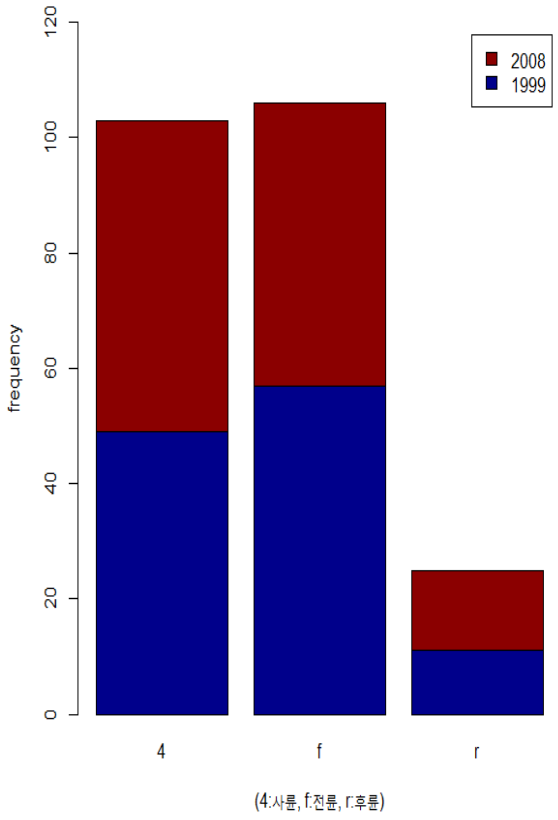
➢ 후륜 구동은 chevrolet,ford,lincoln만 사용하는 걸 확인할 수 있다.

#교차빈도 시각화 연도별 구동 방식 빈도

- barplot(table(mpg$year,mpg$drv),legend.text = T,ylim = c(0,120),col = c("darkblue","darkred"),

  ylab = "frequency",

  xlab = "(4:사륜, f:전륜, r:후륜)") #ylim(y축의 범위 설정),col(색상) legend.text=T(범례)



그래프를 보면 확실히 후륜구동 방식의 차는 적은 것을 알 수 있다.

1999년과 2008년의 구동방식별 제조차의 수는 비슷한 것으로 판단된다.

미세하지만 1999년에는 전류구동이 4륜구동보다 많았고 2008년에는 4륜구동의 차가 더 많이 제조된 것을 확인해 볼 수 있다.

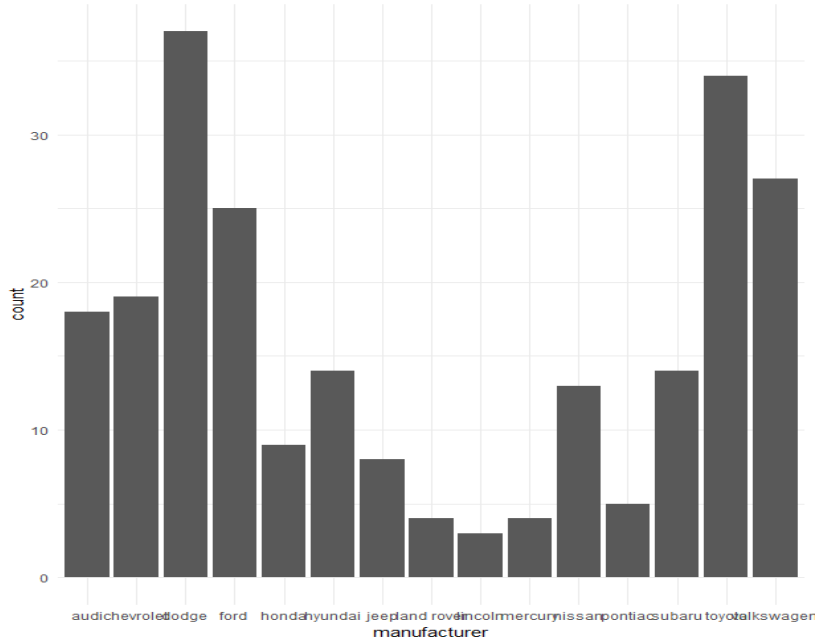
#범주형 빈도 시각화-ggplot2의 geom_bar()함수 이용,R에 내장된 barplot()도 사용 가능

제조사별 빈도수 막대 그래프

- ggplot(mpg, aes(x = manufacturer)) +

  geom_bar()+

  theme_minimal() #theme(배경설정)



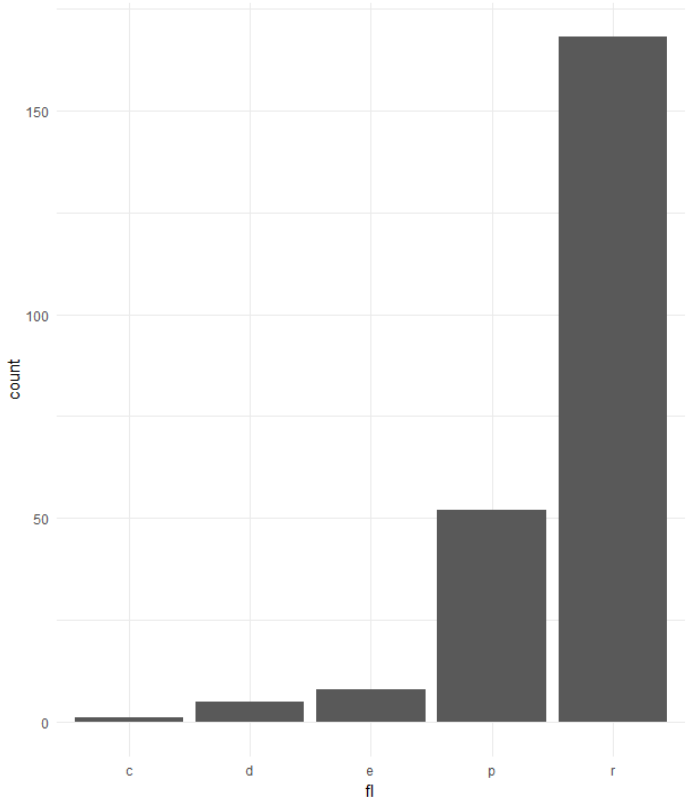
그래프를 보면 dodge차가 가장 많고 lincoln차가 가장 적은 것으로 판단.

연료별 빈도수 막대 그래프

- ggplot(mpg, aes(x = fl)) +

  geom_bar()+

  theme_minimal()



r(regular): 일반 휘발유가 가장 많이 분포된 것을 확인가능하고 p(premium): 고급 휘발유가 분포 ->휘발유가 가장 일반적임을 알 수 있다.

c: 압축 천연가스
d: 디젤
e: 에탄올 E85

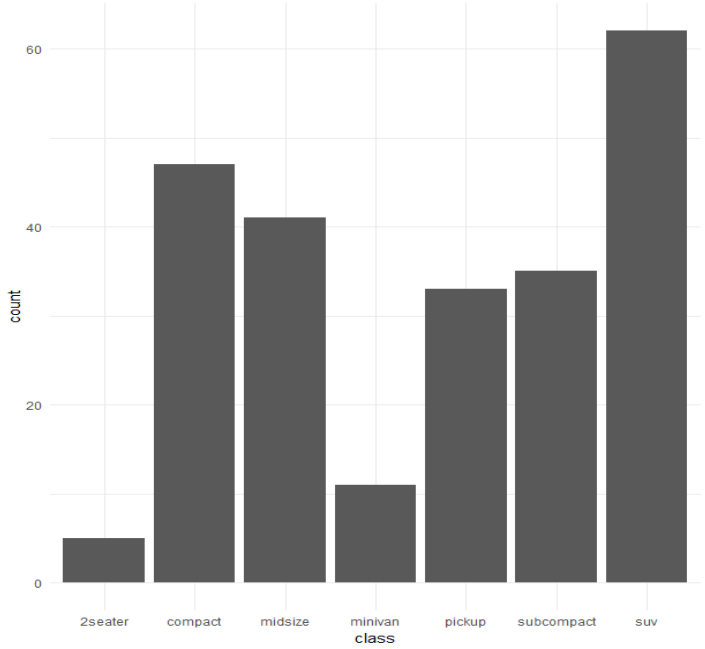
나머지 연료는 적은 분포를 가지고 있다.

자동차 유형별 빈도수 막대 그래프

- ggplot(mpg, aes(x = class)) +

  geom_bar()+

theme_minimal()



suv가 가장 많고 2seater이 가장 적다.

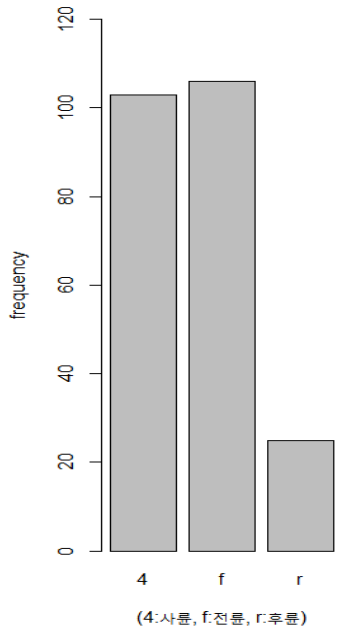
모델별 변속기별 그래프는 생략 (유의미 하지 못하다고 판단)

구동방식별 그래프 시각화는 기본함수 barplot(),boxplot()사용

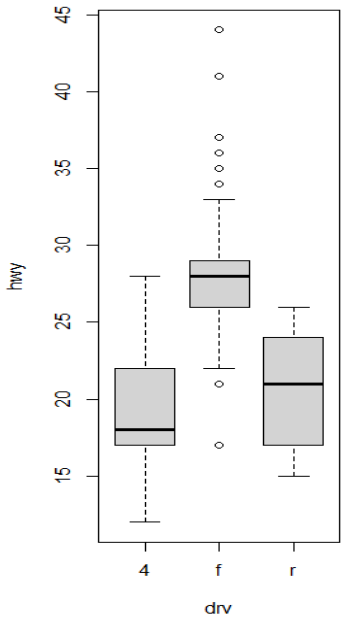
- barplot(table(mpg$drv),ylim = c(0,120),

  xlab = "(4:사륜, f:전륜, r:후륜)",

  ylab = "frequency")



- boxplot(hwy~drv,data=mpg)



앞서 교차 빈도로 확인 한 결과들 boxplot으로 분포도 확인 가능 그리고 y축은 고속도로 연비를 넣어 연비 비교 가능하다.

#양적 자료 분포 시각화-geom_histogram()함수 이용

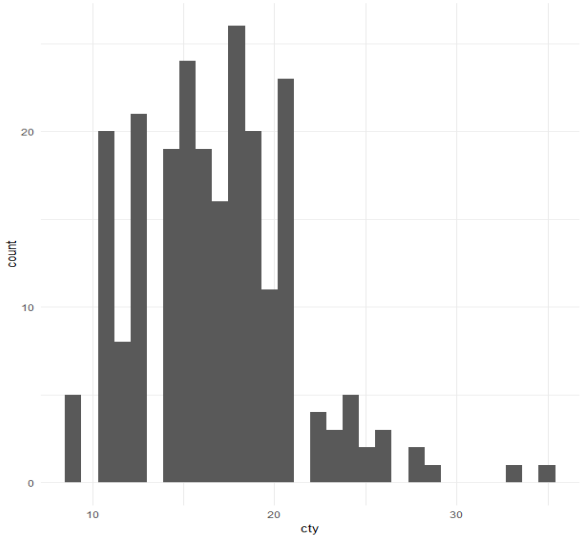
대표적으로 도시연비 변수만 시각화 함.

#도시연비 히스토그램

- ggplot(mpg, aes(x = cty)) +

  geom_histogram()+

  theme_minimal()



연비 분포를 보면 10~20사이에 가장 많이 분포됨을 확인할 수 있다.

1-3 상관계수 분석과 시각화

mpg 데이터 중 수치형 변수 상관계수 확인

```
with(mpg,cor(displ,year))
[1] 0.1478428
    > with(mpg,cor(displ,cyl)) #양의 상관관계(엔진배기량,실린더수)
      [1] 0.9302271
    > with(mpg,cor(displ,cty)) #음의 상관관계(엔진배기량,도시연비)
      [1] -0.798524
    > with(mpg,cor(displ,hwy)) #음의 상관관계(엔진배기량,고속도로 연비)
      [1] -0.76602
> with(mpg,cor(year,cyl))
[1] 0.1222453
> with(mpg,cor(year,cty))
[1] -0.03723229
> with(mpg,cor(year,hwy))
[1] 0.002157643
    > with(mpg,cor(cyl,cty)) #음의 상관관계(도시연비,실린더수)
      [1] -0.8057714
    > with(mpg,cor(cyl,hwy)) #음의 상관관계(고속도로 연비,실린더수)
      [1] -0.7619124
    > with(mpg,cor(cty,hwy)) #양의 상관관계(도시연비,고속도로 연비)
```

```
[1] 0.9559159
```

상관계수는 절대값이 1에 가까우면 상관관계가 크다. 절대값 0.7이상인 값들을 상관관계 있다고 판단.

연관성이 큰 변수들 중 양의 상관계수를 가지는 도시연비와 고속도로 연비, 엔진 배기량과 실린더 수는 정비례 관계이다.

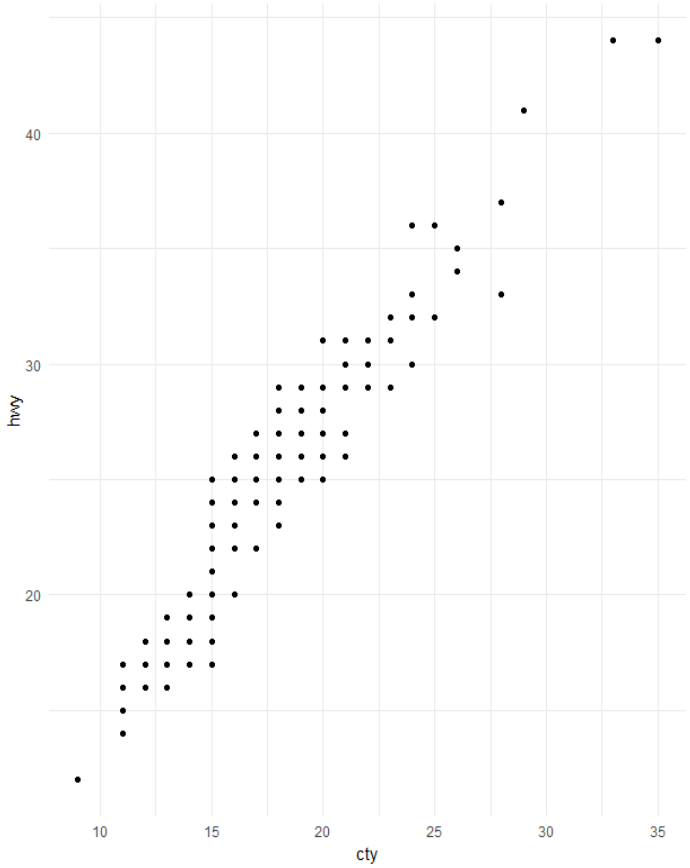
반비례관계로 음의 값의 상관계수를 가지는 변수들은 엔진 배기량과 도시연비, 고속도로 연비이고 실린더 수와 도시연비, 고속도로 연비이다.

#그림을 통해 양의 상관관계 한 개 음의 상관관계 하나씩 시각화-산점도 geom_point()

양의 상관관계

도시연비와 고속도로연비

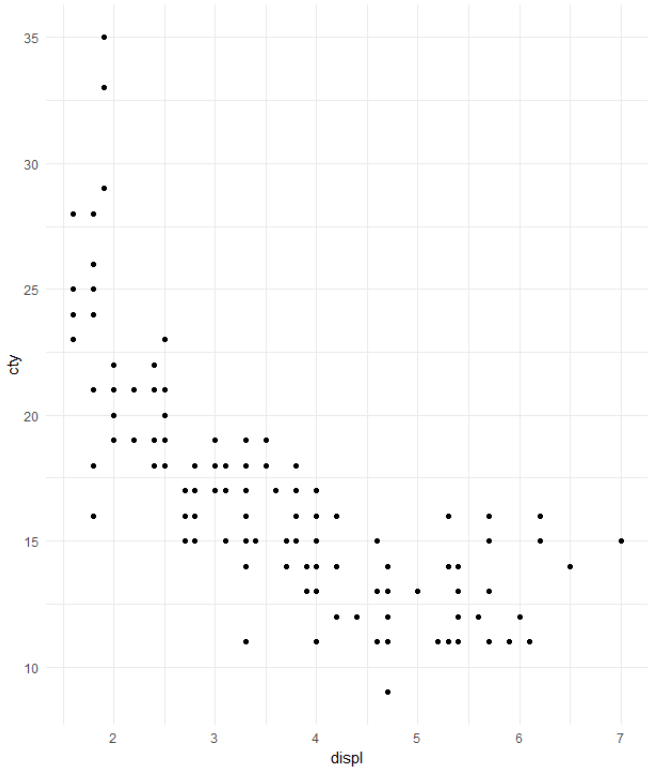
- ggplot(mpg,aes(x=cty,y=hwy))+geom_point()+theme_minimal()



산점도 분포를 보면 도시연비가 높으면 고속도로 연비도 높은 걸 확인할 수 있다. 그래프 모양도 서로 상관관계가 높다는 걸 알 수 있다.

음의 상관관계-엔진배기량과 도시연비

- ggplot(mpg,aes(x=displ,y=cty))+geom_point()+theme_minimal()



앞의 도표처럼 이상적인 비례관계는 아니지만 전체적으로 엔진 배기량수가 낮으면 연비가 높고 엔진 배기량이 많으면 연비가 낮다.

따라서 엔진 배기량의 수가 많은 차를 구매하면 낮은 연비의 차를 구매할 수 있을 것이다.

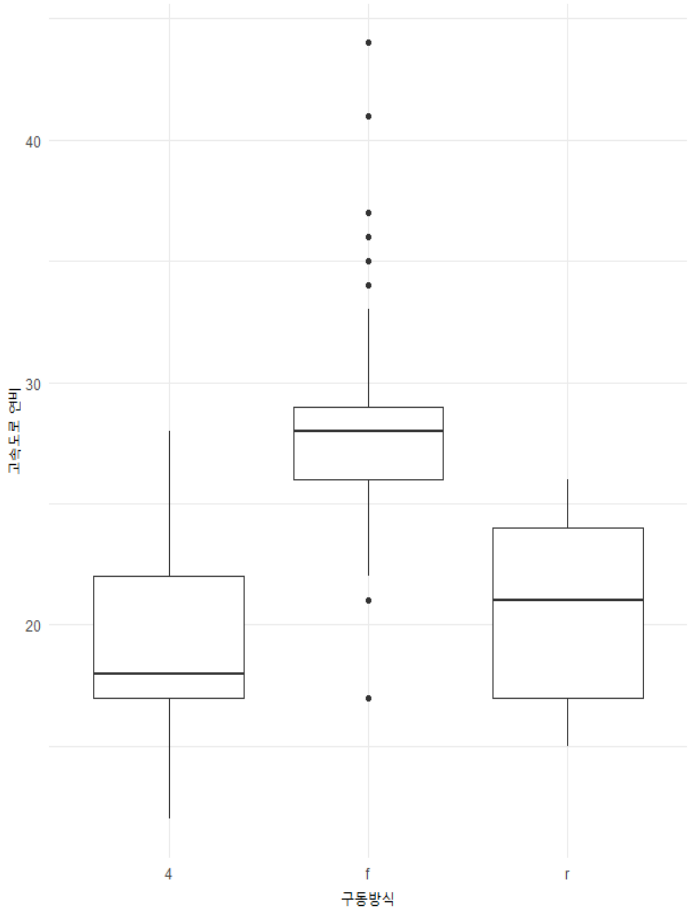
1-4 교차 분석

변수 선별

구동방식별(캐릭터형) 고속도로 연비(수치형) 비교

빠른 속도로 달리는 고속도로 특성상 자동차 구동방식에 따른 연비 차이가 나타나는지 확인해 보자 했음.

- ggplot(mpg,aes(x=drv,y=hwy))+geom_boxplot() +theme_minimal()+labs(x="구동방식",y="고속도로 연비") #ggplot함수는 제목을 labs()에 할당



사륜구동 방식이 확실히 연비가 적은 걸 확인할 수 있다. 평균이 가장 낮고 전체적인 분포 또한 가장 낮다.

후륜구동 방식은 전체적인 분포가 작은 걸 보아 연비 폭이 작을 걸 확인할 수 있고 사륜과 비교해 보아도 연비가 낮은 편이다.

평균도 높고 이상치도 많은 전류구동 방식이 고속도로 연비가 가장 높다.

이 자료를 토대로 판단하면 연비가 가장 낮은 4륜구동 방식의 차를 구매하는 게 좋을 것 같다.

1-5 회귀분석

도로 연비에 미치는 변수를 확인하고자 함.

도로연비를 종속 값으로 두고 전체 회귀분석 시행

- mpg.reg=lm(cty~.,data = mpg)
- mpg.reg
- summary(mpg.reg)

```
modela4                  -0.317006   1.005571  -0.315 0.752943
modela4 quattro          -0.187573   0.939831  -0.200 0.842037
modela6 quattro                NA         NA      NA       NA
```

요약 확인 결과 Na가 반환된 것을 확인

NA가 한번이라도 반환된 독립변수 제거

다시 회귀분석 시행

- mpg.reg2=lm(cty~manufacturer+displ+year+cyl+trans+hwy+fl,data=mpg)

- mpg.reg2

- summary(mpg.reg2)

```
Call:
lm(formula = cty ~ manufacturer + displ + year + cyl + trans +
    hwy + fl, data = mpg)

Residuals:
    Min      1Q  Median      3Q     Max
-2.29107 -0.49651  0.01627  0.52843  3.04101

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)          -72.13776   36.49161  -1.977 0.049421 *
manufacturerchevrolet   0.58066    0.45934   1.264 0.207645
manufacturerdodge       0.81441    0.39573   2.058 0.040874 *
manufacturerford        0.91490    0.40586   2.254 0.025256 *
manufacturerhonda       2.98048    0.41465   7.188 1.25e-11 ***
manufacturerhyundai     0.17003    0.38542   0.441 0.659573
manufacturerjeep        1.26699    0.47954   2.642 0.008885 **
manufacturerland rover  0.14416    0.53397   0.270 0.787451
manufacturerlincoln     0.22023    0.65258   0.337 0.736104
manufacturermercury     1.10883    0.57552   1.927 0.055425 .
manufacturernissan      1.36269    0.37471   3.637 0.000351 ***
manufacturerpontiac    -0.34362    0.53395  -0.644 0.520603
manufacturersubaru      1.28803    0.40371   3.190 0.001647 **
manufacturertoyota      1.43868    0.35261   4.080 6.48e-05 ***
manufacturervolkswagen  1.01212    0.29865   3.389 0.000843 ***
displ                  -0.31745    0.17874  -1.776 0.077234 .
year                    0.03800    0.01823   2.084 0.038396 *
cyl                    -0.19828    0.11849  -1.673 0.095786 .
transauto(l3)           0.68725    0.79846   0.861 0.390413
transauto(l4)          -0.57249    0.48464  -1.181 0.238879
transauto(l5)          -1.11070    0.46693  -2.379 0.018304 *
transauto(l6)          -1.43459    0.58734  -2.443 0.015445 *
transauto(s4)           0.02914    0.70611   0.041 0.967120
transauto(s5)          -1.67186    0.65663  -2.546 0.011638 *
transauto(s6)          -0.82152    0.48931  -1.679 0.094713 .
transmanual(m5)        -0.40792    0.47581  -0.857 0.392282
transmanual(m6)        -0.81596    0.46751  -1.745 0.082443 .
hwy                     0.53710    0.02169  24.768  < 2e-16 ***
fld                     5.30148    1.03094   5.142 6.39e-07 ***
fle                     1.55454    1.01478   1.532 0.127112
flp                     1.98057    0.95125   2.082 0.038595 *
flr                     2.32710    0.94286   2.468 0.014414 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8695 on 202 degrees of freedom
Multiple R-squared:  0.9638,   Adjusted R-squared:  0.9583
F-statistic: 173.6 on 31 and 202 DF,  p-value: < 2.2e-16
```

요약을 토대로 간단 해석

Estimate: 회귀계수의 추정값이 양수면 정비례, 음수면 반비례(독립변수와 종속변수)

Std. Error: 회귀계수 추정값의 표준 오차, 이 값이 작을수록 추정값 신뢰성이 높다.

t value: 회귀계수 추정값을 표준 오차로 나눈 값, 이 값이 클수록 해당 계수는 통계적으로 유의미하다.

Pr(>|t|): 독립 변수가 종속 변수에 미치는 효과가 우연히 발생한 것인지를 나타내는 p-value, 일반적으로 0.05보다 작으면 유의미한 영향을 미친다고 판단한다.

Multiple R-squared 값이 0.9638-이 모델이 96%의 설명력을 가지고 있음을 나타냄. 즉 모델이 적합하다.

#결과 분석

제조별-(혼다,닛산,토요타,폭스바겐), 고속도로 연비, 연료유형-d(diesel)의 변수들이 연비에 큰 영향을 미치는 것으로 판단됨.

여기서 고속도로 연비는 당연히 상관계수가 높기에 연비에 가장 큰 변수임을 확인 가능

그래서 다른 변수를 확인해서 제조별로 연비가 높다고 판단한 혼다,닛산,토요타,폭스바겐은 연비가 높은 차이므로 연비를 생각해서 구매를 한다면 4개의 제조사는 제외할 필요가 있음.

마지막으로 연료유형 변수 중 디젤 연료가 가장 높은 연비를 가져 디젤차 구매를 피하는 것이 좋음.

프리미엄 휘발유, 일반휘발유도 연비와 관련성이 있어 구매 시 고려 대상이다.

추가 분석

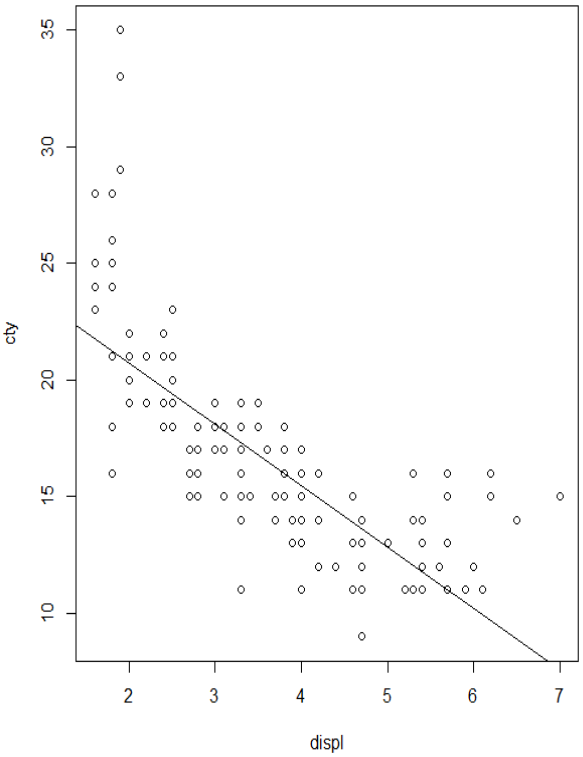
단순 선형 회귀분석- 종속-도시연비, 독립-엔진 배기량 수

- mpg.reg3=lm(cty~displ,data=mpg)
- summary(mpg.reg3)

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  25.9915     0.4821   53.91   <2e-16 ***
displ        -2.6305     0.1302  -20.20   <2e-16 ***
---
```

Cty=-2.63.5*displ+25.9915

- plot(cty~displ,data=mpg) #시각화
- abline(mpg.reg3) #회귀직선



앞서 상관계수를 살펴봤듯이 엔진 배기량과 도시연비는 상관관계가 있었고 회귀직선과 회귀식을 통해 두 변수를 예측할 수 있다.

점진적 회귀분석: 독립변수를 하나씩 넣어보며 최적의 R계수값 구하기->가장 좋은 변수 선택 후 최적의 모델 구축->독립변수 하나 고정후 다시 하나씩 넣어보며 모델 구축

최적의 회귀식 구하기=AIC가 최소가 되는 모델 찾기 MASS패키지 안 stepAIC()함수 이용

caret 패키지 안 dummyVars()함수 범주형에서->수치형으로 변경

- dmy=dummyVars(~.,data = mpg)
- mpg_new=data.frame(predict(dmy,newdata = mpg))

- full_model=lm(cty~.,data = mpg_new) # 가장 큰모델 생성

- stepAIC(full_model,direction = "backward") #변수 제거하면서 진행

```
lm(formula = cty ~ manufacturerchevrolet + manufacturerdodge +
    manufacturerford + manufacturerhonda + manufacturerhyundai +
    manufacturerjeep + manufacturerland.rover + manufacturerlincoln +
    manufacturermercury + manufacturernissan + manufacturersubaru +
    manufacturertoyota + modelc1500.suburban.2wd + modelcamry +
    modelcamry.solara + modelcaravan.2wd + modelexplorer.4wd +
    modelf150.pickup.4wd + modelgti + modeljetta + modelk1500.tahoe.4wd +
    modelland.cruiser.wagon.4wd + modelnew.beetle + displ + transauto.av. +
    transauto.l3. + transauto.l5. + hwy + flc + fld + fle, data = mpg_new)
```

- model.1=lm(cty~1,data = mpg_new) #가장 작은모델 생성성

```
#문자형에서 수치형으로 변경되 변수 다수 존재해 처음 수치형으로만 진행
```

- stepAIC(model.1,scope = cty~displ+year+cyl+hwy,direction = "forward") #변수 추가하며 진행

```
lm(formula = cty ~ hwy + cyl, data = mpg_new)
```

- stepAIC(full_model,direction = "both") #변수 추가 제거 교차하며 진행

```
lm(formula = cty ~ manufacturerchevrolet + manufacturerdodge +
    manufacturerford + manufacturerhonda + manufacturerhyundai +
    manufacturerjeep + manufacturerland.rover + manufacturerlincoln +
    manufacturermercury + manufacturernissan + manufacturersubaru +
    manufacturertoyota + modelc1500.suburban.2wd + modelcamry +
    modelcamry.solara + modelcaravan.2wd + modelexplorer.4wd +
    modelf150.pickup.4wd + modelgti + modeljetta + modelk1500.tahoe.4wd +
    modelland.cruiser.wagon.4wd + modelnew.beetle + displ + transauto.av. +
    transauto.l3. + transauto.l5. + transauto.l6. + hwy + flc +
    fld + fle + modelpathfinder.4wd + transauto.s5., data = mpg_new)
```

2. star 데이터 분석

2-1 데이터 선정과 의미

평소에 관심있고 분석해 보고 싶은 주제 중 우주에 관해 데이터를 살펴보았다. 그 중 별의 특성을 통해 분류해 보면 어떨까 생각해 데이터를 찾았고 Kaggle에서 쉽게 찾을 수 있었다. 데이터 자체도 군집분석에 특화되었다고 생각이 들었는데 이유는 변수도 한가지가 아닌 다양하게 있어 활용하기 좋다는 생각을 했고 데이터의 수도 적당했다. 그리고 이미 분류를 가지고 있는 데이터라 비교해 보기도 쉽다고 생각이 들었다.

Kaggle에서 데이터 다운로드후

#데이터셋 불러오기

- star=read.csv("C:/Users/user/Desktop/6 class csv.csv")

데이터 구조 확인

```
str(star)
'data.frame':  240 obs. of  7 variables:
 $ Temperature..K.      : int  3068 3042 2600 2800 1939 2840 2637 2600 2650 2700 ...
 $ Luminosity.L.Lo.     : num  0.0024 0.0005 0.0003 0.0002 0.000138 0.00065 0.00073 0.0004 0.00069 0.00018 ...
 $ Radius.R.Ro.         : num  0.17 0.154 0.102 0.16 0.103 ...
 $ Absolute.magnitude.Mv.: num  16.1 16.6 18.7 16.6 20.1 ...
 $ Star.type            : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Star.color           : chr  "Red" "Red" "Red" "Red" ...
 $ Spectral.Class       : chr  "M" "M" "M" "M" ...
```

240개의 관측치와 7개의 변수를 가지고 있다.

각각의 변수의 의미는

절대온도(in K)

상대 광도(L/Lo)

상대 반경(R/Ro)

절대 등급(Mv)-별 자체의 밝기를 나타내는 기준

별 유형**(Brown Dwarf(0),Red Dwarf(1),White Dwarf(2), Main Sequence(3) , SuperGiants(4), HyperGiants(5))**

별색(white,Red,Blue,Yellow,yellow-orange etc)

스펙트럼 클래스(O,B,A,F,G,K,M)

2-2 군집분석 구체화

군집분석 중 계층적, 비계층적 선택 별의 특성으로 군집화 시행-여기서 어떤 변수를 활용할지 고민 군집분석후 별의 유형변수와 비교가 가능한지 적용해보기

변수 중 온도와 별 색상 그래프로 시각화해서 색깔에 따른 온도 보여주기

온도와 절대 등급의 상관계수 구해서 연관성 파악

변수 중 별유형에 따른 절대 등급 시각화

2-3 데이터 전처리와 분석

결측치 확인(변수별)

- colSums(is.na(star)) #is.na-결측치 확인, colSums-열의 합

```
colSums(is.na(star))
      Temperature..K.      Luminosity.L.Lo.          Radius.R.Ro.
Absolute.magnitude.Mv.            Star.type
                    0                     0                     0
0                     0
          Star.color       Spectral.Class
                    0                     0
```

결측치 없음.

boxplot()함수로 이상치 확인-이상치 다수 존재하지만 제거 필요성 못 느낌-각각의 별이 가지는 특성 다 활용할 필요가 있다고 판단 추후 이상 있을 시 제거

열이름 단순화(분석하기 편하게 변경)

- colnames(star)=c("Temper","Lo","Ro","Mv","type","color","class")

먼저 온도와 색깔 시각화

star$color 확인해 보면 대소문자 통일 안된 것을 볼 수 있음

#소문자 통일

- star$color=tolower(star$color)

#특수문자 제거

- star$color=gsub("[[:punct:]]", "",star$color)

#띄워쓰기 제거

- star$color=gsub(" ", "",star$color)

색깔별 평균 온도 계산

- library(dplyr)
- temp_col <- star %>%

  group_by(color) %>%

  summarise(mean_tem=mean(Temper))

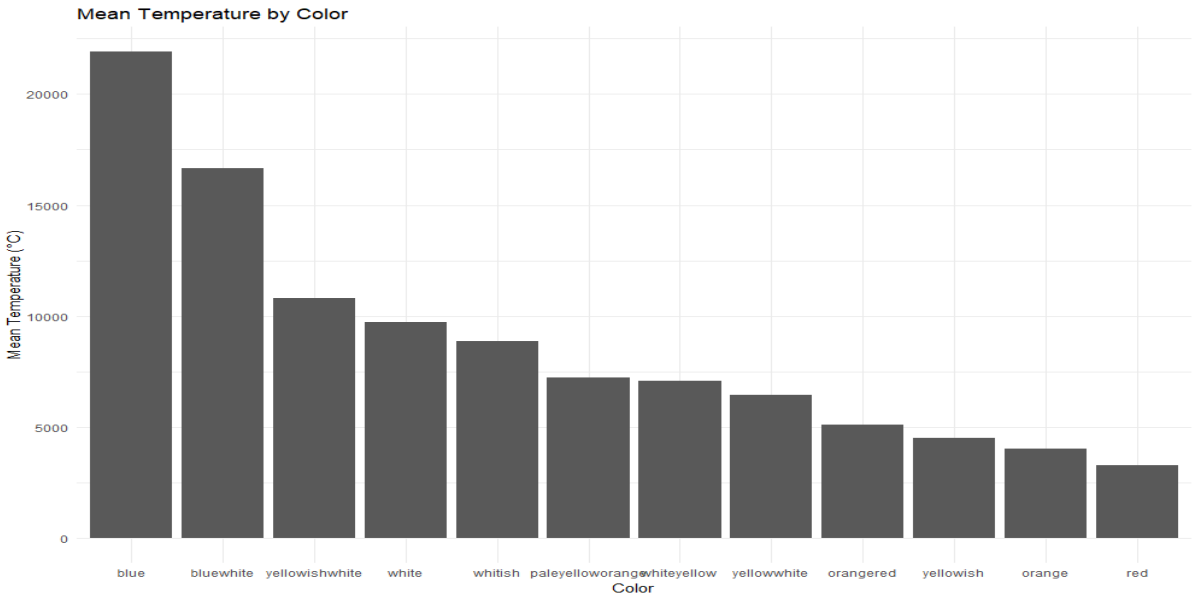
막대그래프 시각화

- library(ggplot2)
- ggplot(temp_col, aes(x=reorder(color,-mean_tem),y=mean_tem)) + #내림차순 적용

  geom_bar(stat = "identity") + # 막대그래프 생성(데이터 값 그대로 막대높이 적용)

  labs(title = "Mean Temperature by Color",x="Color",y="Mean Temperature (°C)") +

  theme_minimal()



그래프를 보면 붉은 별이 평균 온도가 가장 낮은 것으로 확인되고 푸른 별이 평균 온도가 가장 높다. 내림차순 정렬로 푸른빛일수록 온도가 높고 붉은색으로 갈수록 온도가 낮아진다.

별의 밝기는 파장에 따라 달라지므로 온도가 높으면 짧은 파장의 복사에너지가 방출되어 푸른색으로 보이고 표면온도가 낮으면 긴 파장의 복사에너지가 나와 붉은 색으로 보이기 된다.

온도, 절대 등급(밝기) 상관계수 파악

```
cor(star$Temper,star$Mv)
[1] -0.4202605
```

내 생각은 온도와 밝기가 상관이 있다고 판단해 상관계수를 구해보았는데 내 예상외로 0.5보다 낮은 값이 나왔다. 즉 큰 연관성이 없다고 판단이 된다.

상관계수 파악

```
cor(star[,c(1:4)])
            Temper         Lo         Ro         Mv
Temper  1.00000000  0.3934041  0.06421597 -0.4202605
Lo      0.39340408  1.0000000  0.52651572 -0.6926192
Ro      0.06421597  0.5265157  1.00000000 -0.6087282
Mv     -0.42026054 -0.6926192 -0.60872823  1.0000000
```

값들을 살펴보면 크게 유의미한 관계는 찾아 보기는 힘들다. 그나마 가장 큰 상관계수는 음의 상관계수를 가지는 상대광도(Lo)와 절대 등급(Mv)이다.

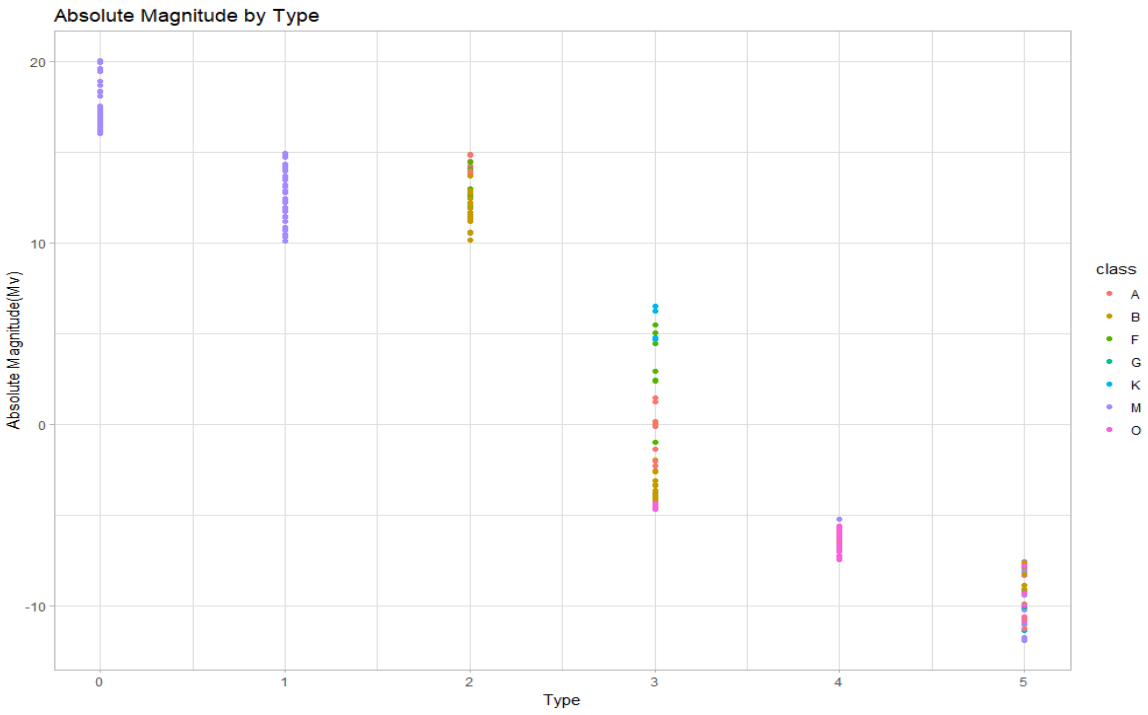
별유형에 따른 절대 등급 시각화

- ggplot(star,aes(x=type, y=Mv, color=class))+ #개별 클래스 색깔로 표시해 분포 확인

  geom_point(stat = "identity")+

  labs(title="Magnitude by Type", x="Type", y="Magnitude(Mv)")+

  theme_light()



Brown Dwarf(0):15~20, Red Dwarf(1): 10~15, White Dwarf(2):10~15, Main Sequence(3):-5~8 , SuperGiants(4):-8~-5, HyperGiants(5):-12~-8의 범위를 가지고 있다.

여기서 주의할 점은 별의 등급은 낮을수록 별이 밝다는 뜻이고 음수 값을 가지면 매우 밝은 별이란 뜻이다. 이로 인해 갈색 왜성이 가장 어두운 별의 분포를 가지고 있고 극대거성은 매우 밝은 별의 분포이다.

2-4 군집분석

군집분석을 수행할 때 Data Scaling이 필요한 이유

변수 간 크기 차이의 영향 완화시켜준다. 데이터의 각 변수들이 서로 다른 단위를 가질 수 있다. 예를 들어, 변수 A는 미터 단위로 측정된 거리일 수 있고, 변수 B는 킬로그램 단위로 측정된 무게일 수 있다. 이러한 경우, 스케일링을 하지 않으면 값의 범위가 큰 변수가 거리 계산에 더 큰 영향을 미치게 된다. 그리고 변수 간의 크기 차이 때문에 특정 변수가 다른 변수들보다 군집 형성에 더 큰 영향을 미칠 수 있기에 동등한 가중치로 변경시켜줘야 한다.

대부분의 군집분석 알고리즘(예: K-평균, 계층적 군집분석)은 변수 간의 거리를 계산하여 유사성을 측정한다. 스케일링을 하지 않으면, 큰 값을 가진 변수들이 거리 계산을 지배하게 되어 결과적으

로 왜곡된 군집을 형성하게 된다.

수렴 속도 개선: K-평균 같은 군집 분석 알고리즘은 반복적인 최적화 과정을 통해 수렴한다. 스케일링을 통해 수렴 속도를 개선하여 더 빠르게 최적의 군집을 찾을 수 있도록 도와준다.

스케일링을 통해 데이터의 범위를 제한하면, 수치적으로 더 안정적인 계산이 가능하다. 이는 알고리즘의 결과가 데이터의 초기 값에 덜 민감하게 만들고, 보다 안정적인 군집을 형성이 가능하다.

스케일링 진행

```
head(star)
  Temper      Lo     Ro    Mv type color class
1   3068 0.002400 0.1700 16.12    0   red     M
2   3042 0.000500 0.1542 16.60    0   red     M
3   2600 0.000300 0.1020 18.70    0   red     M
4   2800 0.000200 0.1600 16.65    0   red     M
5   1939 0.000138 0.1030 20.06    0   red     M
6   2840 0.000650 0.1100 16.98    0   red     M
star2=star[,1:4] #스케일링 진행할 변수만 추출
head(star2)
  Temper      Lo     Ro    Mv
1   3068 0.002400 0.1700 16.12
2   3042 0.000500 0.1542 16.60
3   2600 0.000300 0.1020 18.70
4   2800 0.000200 0.1600 16.65
5   1939 0.000138 0.1030 20.06
6   2840 0.000650 0.1100 16.98


star_scale=scale(star2) #scale() 함수 이용
summary(star_scale)
     Temper              Lo                Ro                Mv
 Min.   :-0.8959   Min.   :-0.5974   Min.   :-0.4586   Min.   :-1.5478
 1st Qu.:-0.7488   1st Qu.:-0.5974   1st Qu.:-0.4584   1st Qu.:-1.0078
 Median :-0.4943   Median :-0.5974   Median :-0.4571   Median : 0.3732
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 0.4772   3rd Qu.: 0.5064   3rd Qu.:-0.3759   3rd Qu.: 0.8844
 Max.   : 3.0885   Max.   : 4.1366   Max.   : 3.3091   Max.   : 1.4885
```

#군집수 구하기

- nc=NbClust(star_scale,min.nc=2, max.nc=15, method="kmeans")

```
According to the majority rule, the best number of clusters is  3
```

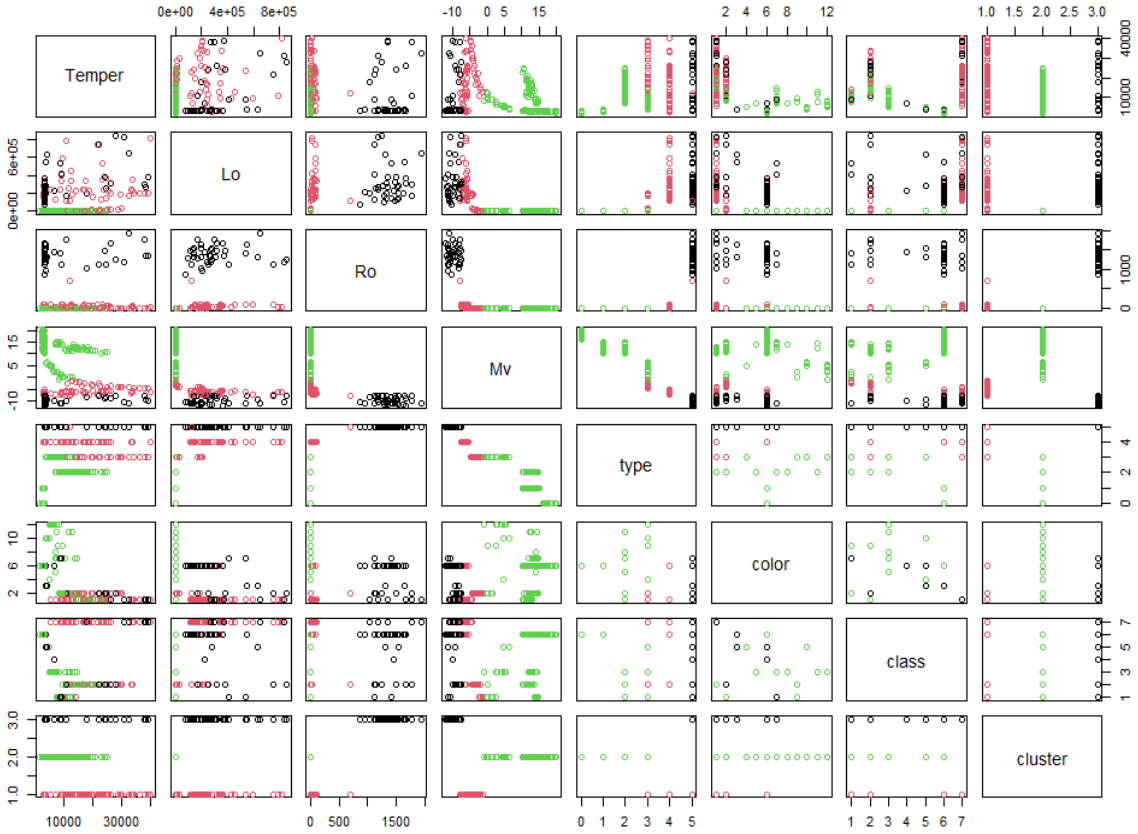
- star_km=kmeans(star_scale,centers = 3,nstart = 50) #50번 반복시행후 최소값 적용

```
table(star_km$cluster) # 클러스터링 빈도수

  1   2   3
 39  61 140
```

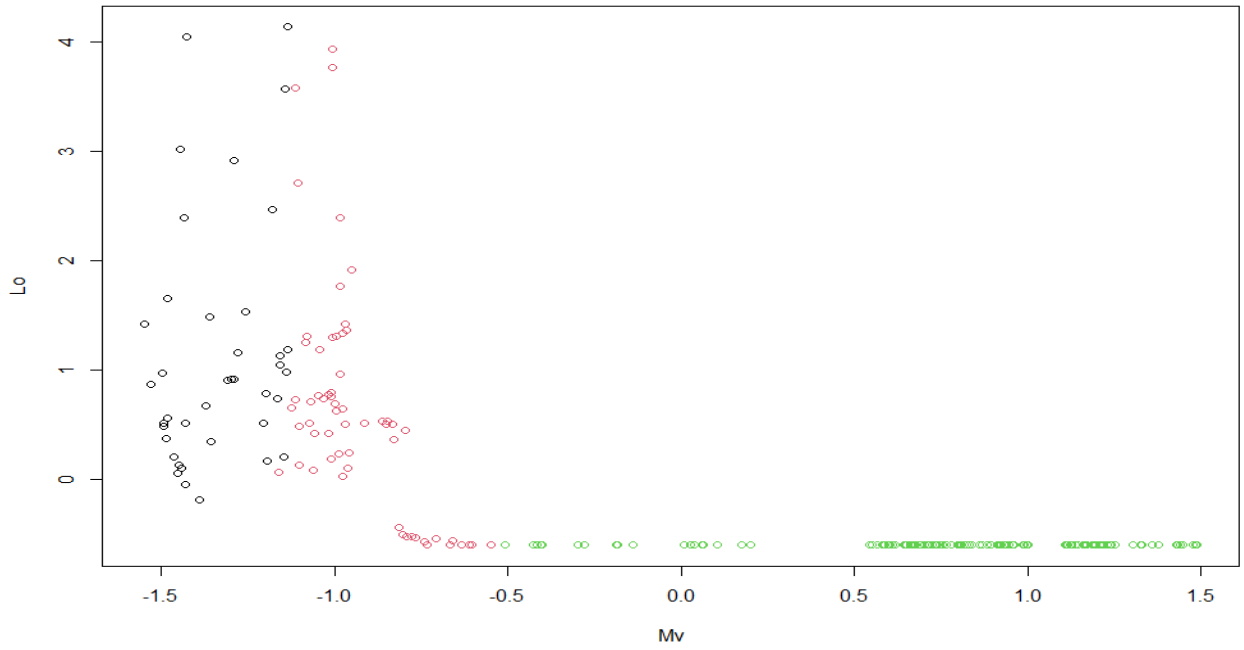
- star$cluster=star_km$cluster #클러스터 변수 추가

- plot(star,col=star_km$cluster) # 전체 시각화로 변수간 관계 확인



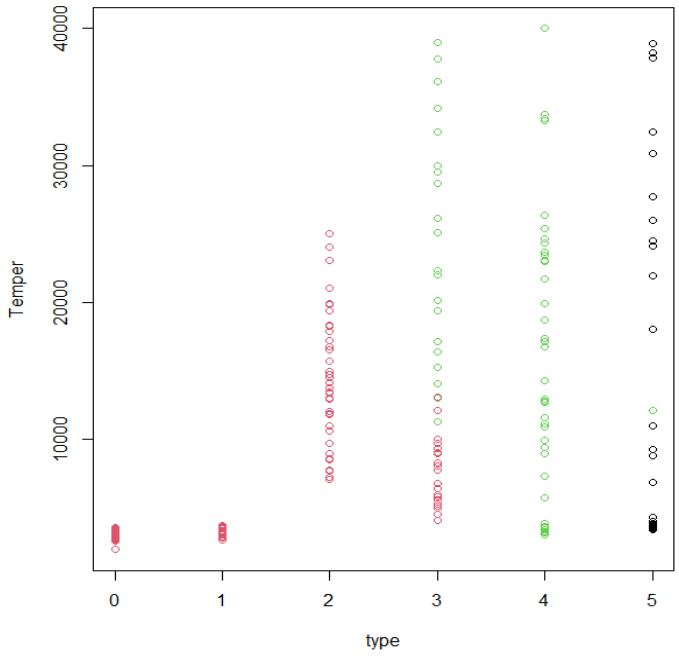
대략 절대 등급별로 클러스터링 구분 잘된 것을 확인 가능하다. 또 유형별로도 나뉘는 것을 알 수 있음. 상관계수가 커질수록 클러스터가 명확히 구분

- plot(star_scale[,c(4,2)],col=star_km$cluster) # 절대등급과 상대 광도 클러스터링 시각화



절대 등급이 (-2 ~ -1.3), (-1.3 ~ 0.5), (0.5 ~ 1.5) 3부분으로 클러스터링 된 것을 확인.

- plot(star[,c(5,1)],col=star_km$cluster) # 유형별 온도에 따른 클러스터링



대략 0~2까지 군집 되었고 주계열성(3)이 3개의 군집 다 포함된 것을 확인 할 수 있다.

극대거성(5)도 3개의 군집이 다 있는 것으로 확인.

```
table(star$cluster,star$type)
```

```
    0  1  2  3  4  5
1   0  0  0  0  0 39
2   0  0  0 20 40  1
3  40 40 40 20  0  0
```

처음 궁금했던 별유형별 군집 빈도는 주계열성(3)이 두개의 군집에 포함이 되었고 다른 별유형은 대체적으로 비슷한 군집으로 묶인 것을 알아볼 수 있다.

- star_km=kmeans(star_scale,centers = 6,nstart = 50) #클러스터링 수 6개 지정 후 군집화 시행

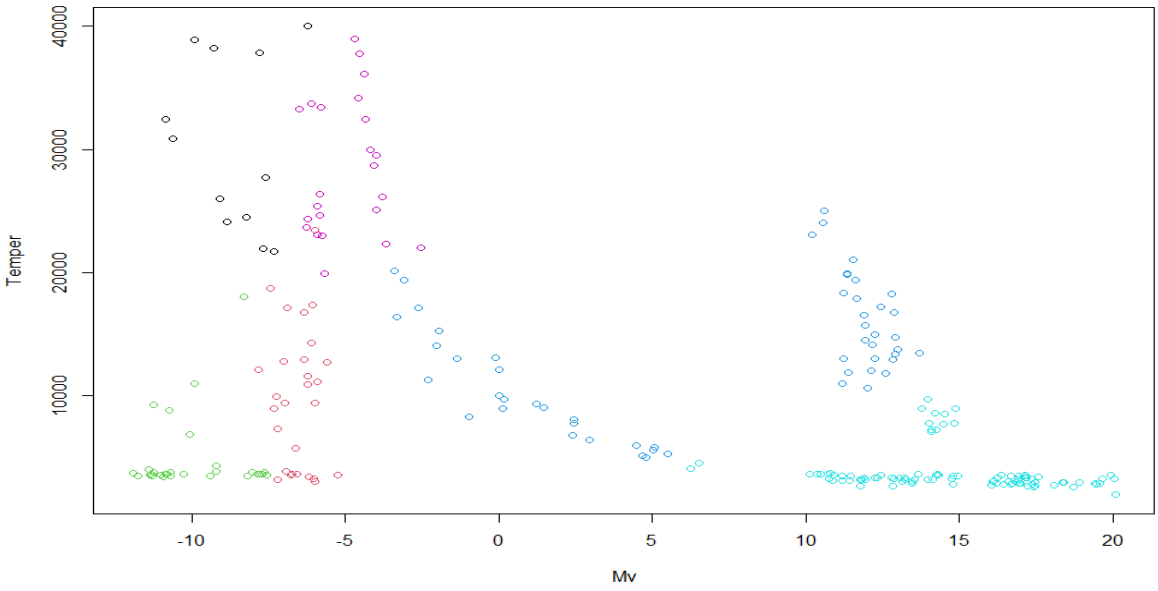
6개 지정 이유는 별유형이 6개이므로 정해 봄.

```
table(star$cluster,star$type)
```

```
   0  1  2  3  4  5
1  0  0  0 12 11  0
2  0  0 29 26  0  0
3 40 40 11  2  0  0
4  0  0  0  0 27  1
5  0  0  0  0  0 29
6  0  0  0  0  2 10
```

내가 원했던 별 유형별로 딱 맞아 떨어지지 않지만 대충 가까운 별유형끼리 군집화 된 걸 볼 수 있다.

- plot(star[,c(4,1)],col=star_km$cluster) #6개 군집 절대등급 온도 시각화



절대 등급과 온도별은 6개의 군집이 잘 보여 3개보다 더 정밀한 분석이 가능할 것으로 판단됨.

3.보완점

회귀분석시 인자형 상관이 있을까? (문자, 수치)에 대한 고민을 해보면 좋을 것 같다.

문자형도 분석에 포함이 되는데 이건 어떤 방식일까? 굳이 수치형으로 변경하는 이유는?

회귀분석시 독립변수 전체 설정 후 분석과 하나씩 설정 후 분석의 값이 달라 판단이 어려운데 어떤 방식이 효율적일까? 예)전체분석 후 휘발유가 연료에 연관성이 가장 크다. 하나씩 분석할 경우 디젤연료가 연료에 연관성이 가장 크다.

군집 분석에 있어 x축과 y축에 대한 고민이 필요해 보인다.

Kmeans 군집 분석에 더 좋은 시각화 방법은 없을까? 아쉬움이 남는다.

데이터 선정 후 처음 계획하고 수립한 가상의 분석과 실제 분석의 차이가 클 때 어떻게 받아들일지 아직 혼란스러움이 크다. 최대한 주관을 빼고 분석하고 분석한 결과만 가지고 판단하는 게 맞는 방법일지 고민해 볼 필요가 있다.

아직 그래프 분석 방법에 대해 잘 모르겠다. 어떤 판단을 내려야 할지 감이 안 온다. 더 많은 공부와 배경지식이 필요하다는 걸 깨닫게 됨.