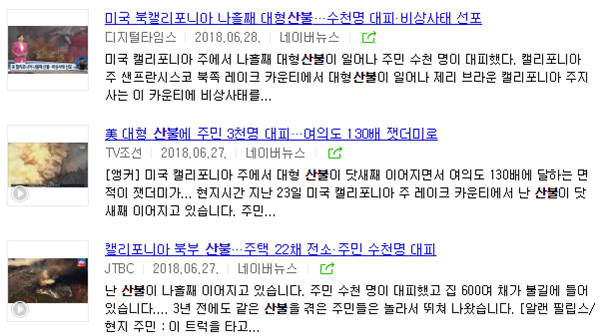
기상 데이터를 활용한 산불 예측 데이터 분석

1. 전민규(19940810)

2. jskstar12@naver.com

3. 전민규(19940810) 이잉걸(19941006) 서석현(19951222)

### ForestFires





매년 전세계 곳곳에서 산불이 일어나며 그로 인한 피해는 매우 크기 때문에, 늘 인류의 골머리를 앓게 하는 환경 문제였다. 산불 발생의 대부분은 등산객의 담배 꽁초 투기, 등산 중 조리 행위 등 물리적 발화 요인에서 기인하지만, 산림과 인명, 재산에 피해를 주는 규모의 산불은 건조한 기후나 강수량 등 자연 발화 요인에서 기인하기도 한다. 따라서 합당한 통계적 추론을 통해 산불 발생지를 사전에 예측하여 인명, 재산 등의 예상 피해를 산정하고 그에 따른 대비책을 강구한다면 큰 도움이 될 것이다.

이러한 통찰에 따라 기상, 토양 데이터를 수집하여 향후 분석 계획을 수립, 초기 단계의 분석과 기초 모델링을 진행했다. 이후 추가적으로 주어지는 분석 기간에 더욱 구체적이고 정교한 분석 방법과 더 많은 변수들을 사용하여 산불 예측에 기여하고자 한다. 분석에는 UCI에서 제공하는 ‘Forest Fires Data Set ‘을 이용하였다.

- 데이터 출처 : UCI Machine Learning Repository

                       (<https://archive.ics.uci.edu/zml/datasets/Forest+Fires>)

- 포르투갈 Tra’s-os-Montes 동북지역에 있는 Montesinho 자연 공원의 산불에 대한 FWI지수와 기상 데이터

- 총 517개의 데이터

**변수 이름과 설명**

**1) 장소 정보**

1. X : Montesinho 자연 공원 지도의 x축 공간 좌표 - 1 to 9

2. Y : Montesinho 자연 공원 지도의 y축 공간 좌표 - 2 to 9

**2) 시간 정보**

3. month : 1월~12월 - jan to dec

4. day : 월요일~일요일 - mon to sun

**3) FWI: The forest Fire Weather Index (FWI) is the Canadian system for rating fire danger**

-> BUI값이 80을 기준으로 정해짐

  5. FFMC : the moisture content surface litter and influences ignition and fire spread - 18.7 to 96.20

   - 낙엽층(숲 바닥에서 0~1.2cm 깊이)의 깔짚 및 다른 경화 미세 연료의 수분 함량에 대한 수치 등급

   - 점화 확률과 확산에 큰 영향을 미침

6. DMC : the moisture content of shallow organic layers: 1.1 to 291.3

 - 느슨하게 압축되어 있는 토양(숲 바닥에서 1.2~7cm 깊이,유기층(Duff))의 평균 수분 함량에 대한 수치 등급

 - 번개 수용력과 모든 화재 강도에 기여

7. DC : DC index represent the moisture content of deep organic layers - 7.9 to 860.6

 - 수분 함량이 소형 유기물의 깊은 층(숲 바닥에서 7+cm 깊이)에 얼마나 많이 포함되어 있는지를 나타냄

-long term memory

 - DC값이 높을수록 화재 진압에 큰 어려움이 있음 -> 수분함량이니 당연!

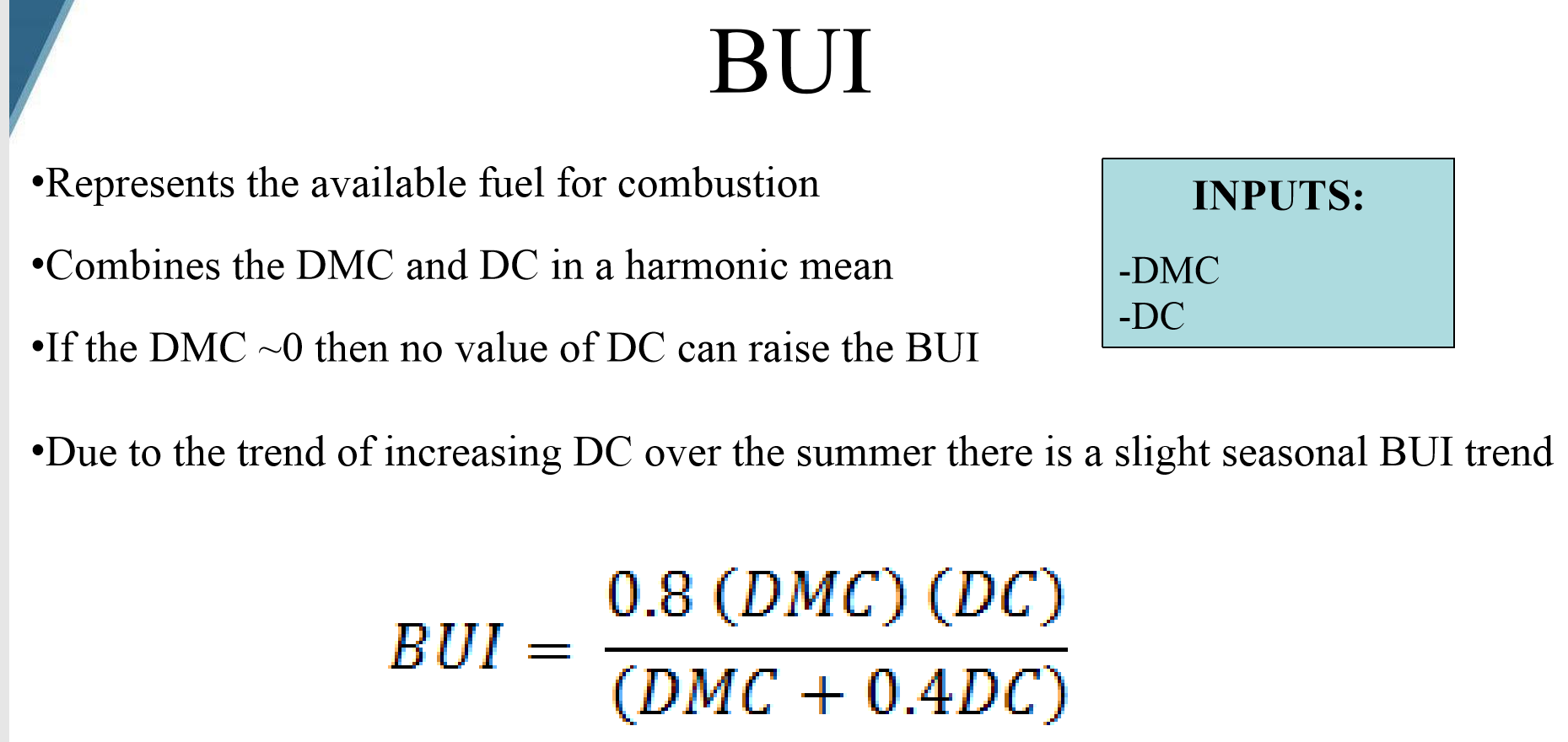
8. ISI : the score that correlates with fire velocity spread - 0.0 to 56.10

 - 화재 지시계 및 화재 확산 비율 측정치

 - ISI 지수가 높을수록 화재 통제가 어렵다

->Input으로 WS(풍속)과 FFM(Fine Fuel Moisture)가 들어감(확산 비율 측정치니깐 바람이!)

 - 풍속과 미세 연료 수분 코드(FFMC)의 효과를 조합한 값



**4) 기상 정보**

9. temp : 섭씨 온도(℃) - 2.2 to 33.30

10. RH : 상대 습도(%) - 15.0 to 100

 11. wind : 풍속(km/h) - 0.40 to 9.40

 12. rain : 강수량(mm/m2) - 0.0 to 6.4

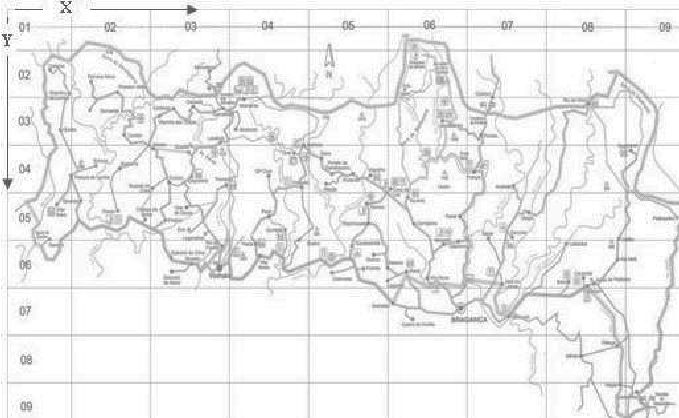
**5) 반응 변수**

 13. area : 산불로 인해 불탄 지역 면적(ha) - 0.00 to 1090.84

1.특정일 이전의 기간 동안 비가 적게 내릴수록 산불 발생의 확률이 높아짐

2.평균 강수량 데이터는 4가지 기상 데이터 중 산불 발생 확률에 가장 많은 영향을 미친다

3. 산불이 발생하였을 때 산불의 피해규모 예측 모형의 적합에서 회귀분석을 하지 못하고 대신 분류(classification)분석 방법을 이용하였다. 그 이유는 반응변수가 매우 꼬리가 두터운 분포를 가지므로 선형모형이나 비선형 모형을 이용한 모형의 적합이 어렵기 때문이다. 따라서 평균이 존재하지 않는 반응변수(여기는 화재가 발생할 확률값)에 대한 회귀분석 방법을 고려해야 할 것이다

날씨가 덥다 -> 그리스 산불 -> 산불을 조심하자(의식의 흐름)  By Hurtuv - Holidays in MontesinhoPreviously published: no, CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=39454624> ### 변수 확인 

**데이터 정제**

데이터를 살펴보고 시각화하는 과정에서 가장 먼저 든 의문은 ‘과연 각 지역의 기상 정보가 산불 발생에 얼마나 영향을 미치는가?’였다. 다양한 통찰을 얻기 위해서 다른 시각화 자료도 확인해보았다. 우선적으로 진행한 시각화는 데이터가 각자 어떻게 생겼는지에 대한 분포를 확인하기 위해서 진행하였다. 아래에 있는 그림을 보게 되면 수치형 변수들로 되어 있는 변수들에 대해서 모두 분포도를 확인하였다. 불이 났는지에 대한 area 변수를 보았을 때 극단적으로 0의 값들이 많고 치우쳐져있는 것을 알 수 있었다. 즉 화재의 규모를 예측하는 것보다 불이 났는지에 대한 여부를 판단하는 것이 더 적합하다고 판단하였고 모델링을 할 때도 그 중심적으로 하게 되었다.

 두 번 째 진행한 시각화는 각 x,y 지역에서 산불의 빈도수를 확인하기 위해서 진행한 시각화이며 세 번째는 각 달과 요일에서 얼마나 산불이 많이 진행되었는지 확인할 수 있는 시각화이다. 네 번째 시각화 자료는  앞서 두 번째 시각화를 직관적으로 좌표 평면상으로 확인해 보기 위해서 진행한 시각화이다. (6,5) 지역에서 가장 화재가 많이 발생했다는 것을 알 수 있었다. 여기서 지역 변수가 어느 정도 화재에 대해서 영향을 미친다고 유추할 수 있었다.  5 번째 시각화 자료는 각 지역별로 화재 정도를 1-6으로 구분하고 빈도 수를 확인한 작업을 진행하였다. 6번째는 각 지역별 화재가 발생했는지 여부에 따라 분포를 확인해본 시각화 자료이다. 종합적으로 유추해봤을 때 지역 (6,5) 같은 경우에는 안 발생하는 것보다 더 화재가 많이 발생한다는 것을 알 수 있었고 계절과 요일 같은 경우에도 어느 정도 화재에 대해서 영향을 미친다는 것을 유추 할 수 있었다.

library(data.table)  
forestfire <- fread("~/Downloads/forestfires.csv")  
library(ggplot2)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':  
##   
## between, first, last

## The following objects are masked from 'package:stats':  
##   
## filter, lag

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

library(cowplot)

##   
## Attaching package: 'cowplot'

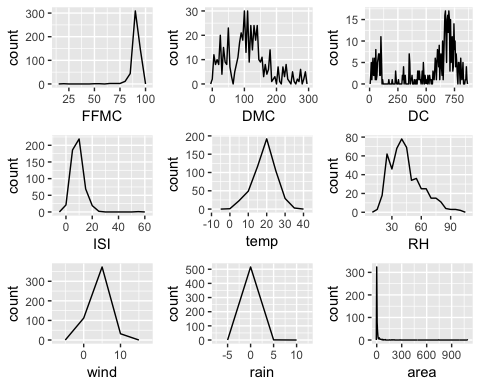
## The following object is masked from 'package:ggplot2':  
##   
## ggsave

library(caret)

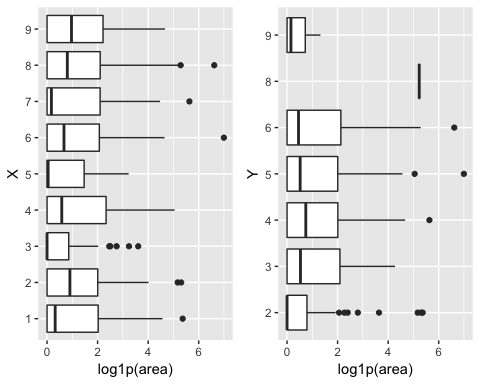
## Loading required package: lattice

theme\_set(theme\_grey())

# 연속형 변수들 히스토그램  
n <- names(forestfire)  
plots <- apply(forestfire[,c(5:13), with=FALSE], 2,function(x){  
 ggplot(forestfire, aes(x))+ geom\_freqpoly(binwidth = 5) #freqpoly : histogram을 뾰족하게 그리는것  
 })  
for(i in 1:9){ #제목 붙이기 위함  
 plots[[i]] <- plots[[i]] + xlab(n[i+4])  
}  
plot\_grid(plotlist = plots) #데이터를 histogram / cowplot이라는 패키지 : 9개를 원하는 형태로



# 지역별 화재 발생 확인 박스 플롯  
plots2 <- lapply(forestfire[,c(1:2), with =FALSE], function(x){  
 ggplot(forestfire, aes(factor(x), log1p(area))) + geom\_boxplot() + coord\_flip()   
}) #x좌표, y좌표를 90도로 회전(coord\_flip)  
for(i in 1:2){#제목  
 plots2[[i]] <- plots2[[i]] + xlab(n[i])  
}  
plot\_grid(plotlist = plots2)

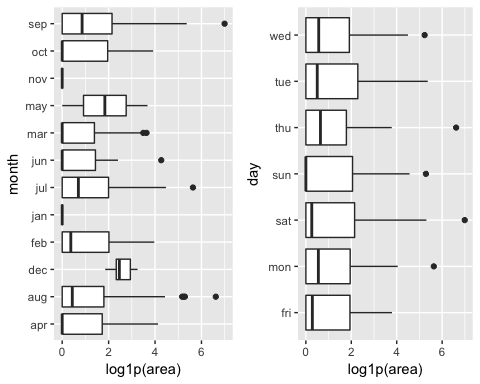


x좌표별로 boxplot

0~6 -> (log1p : 0이 들어가도 0을 뱉어냄) log값을 거쳐서

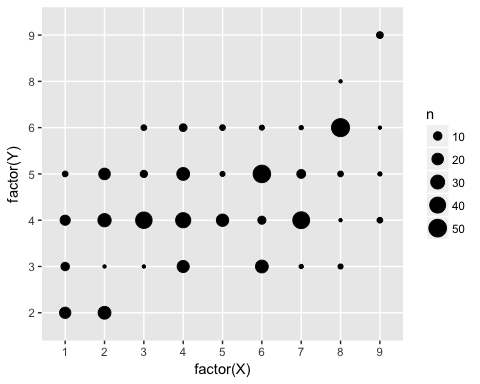
-지역별 화재 발생이 어느정도 되는지, x,y별로

# 일,월 별 화재 발생 확인 박스 플롯  
plots3 <- lapply(forestfire[,c(3,4), with =FALSE], function(x){  
 ggplot(forestfire, aes(x, log1p(area))) + geom\_boxplot() + coord\_flip()   
})  
for(i in 1:2){  
 plots3[[i]] <- plots3[[i]] + xlab(n[i+2])  
}  
plot\_grid(plotlist = plots3)



Day는 거의 random

# 변수간 상관 관계랑 모두 볼 수 있음 하지만 시간이 조금 걸림  
#install.packages("GGally")   
#library(GGally)  
#ggpairs(forestfire[,c(-1,-2,-3,-4)])  
  
  
# 지역 별 데이터 빈도수 확인 -> 많을수록 불이 날 확률도 업  
ggplot(forestfire, aes(factor(X),factor(Y))) + geom\_count() + scale\_size\_area()



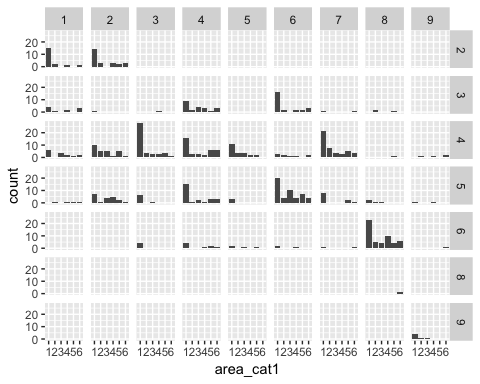
->공원지도를 그린 것

->6,5부분이 불이 불이 엄청 많이남 -> 고지대?라고 유추를 할 수 있겠다(번개가 많이 치지 않았을까…?)

#화재를 6단계로 나누어서 계산 1이 작은 화재 6은 큰 화재(화재의 크기에 따라 나눔)  
# 연속형 변수인 area를 카테고리로 잠깐 만들어서 확인  
area\_cat1<- cut(log1p(forestfire$area), breaks = quantile(log1p(forestfire$area), probs = c(0,seq(0.5, 1, 0.1))),  
 include.lowest = TRUE, right = FALSE, labels = 1:6)  
data2 <- forestfire %>% mutate(area\_cat1 = area\_cat1)  
#시각화  
data2 %>% group\_by(X, Y, area\_cat1) %>% summarise(n())

## # A tibble: 129 x 4  
## # Groups: X, Y [?]  
## X Y area\_cat1 `n()`  
## <int> <int> <fct> <int>  
## 1 1 2 1 15  
## 2 1 2 2 2  
## 3 1 2 4 1  
## 4 1 2 6 1  
## 5 1 3 1 4  
## 6 1 3 2 1  
## 7 1 3 4 2  
## 8 1 3 6 3  
## 9 1 4 1 6  
## 10 1 4 3 4  
## # ... with 119 more rows

ggplot(data2, aes(area\_cat1)) + geom\_bar() + facet\_grid(Y~X)

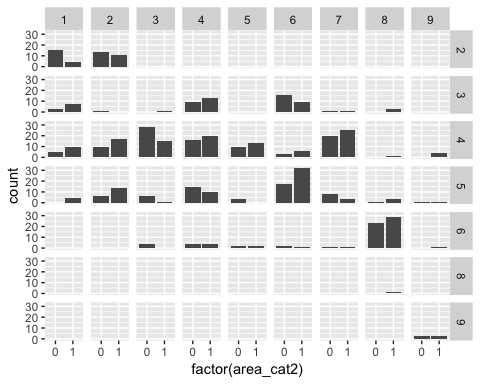


6,5지역에 큰 화재들이 많이 난 것을 볼 수 있음

#화재를 2단계로 나누어서 계산, 0은 화재가 0 1은 0보다 큰 경우 (화재를 발생했는지 안했는지로 나눔 , 1이 발생한거)  
data2 <- data2 %>% mutate(area\_cat2 = ifelse(forestfire$area > 0, 1, 0))  
data2 %>% group\_by(X,Y, area\_cat2) %>% summarise(n())

## # A tibble: 62 x 4  
## # Groups: X, Y [?]  
## X Y area\_cat2 `n()`  
## <int> <int> <dbl> <int>  
## 1 1 2 0 15  
## 2 1 2 1 4  
## 3 1 3 0 3  
## 4 1 3 1 7  
## 5 1 4 0 5  
## 6 1 4 1 10  
## 7 1 5 1 4  
## 8 2 2 0 14  
## 9 2 2 1 11  
## 10 2 3 0 1  
## # ... with 52 more rows

ggplot(data2, aes(factor(area\_cat2))) + geom\_bar() + facet\_grid(Y~X)



역시 6,5가 완전!

data3 <- forestfire #월 데이터를 계절로!(참고한 논문에서 계절성을 넣었을 때 성능이 괜찮았다!, 참고했음)   
data3 <- data3 %>% mutate(weather = ifelse(month %in% c("feb","jan","dec"), "winter",  
 ifelse(month %in% c("oct","nov","sep"), "autoumn",  
 ifelse(month %in% c("aug","jul","jun"), "summer", "spring"))))  
  
data3 <- data3 %>% mutate(weekend = ifelse(day %in% c("mon", "tue", "wed", "thu", "fri"), "week", "weekend"))  
  
data3 <- data3 %>% mutate(area1 = ifelse(area > 0, 1, 0))  
  
data3$weather <- factor(data3$weather, levels = c("spring", "summer", "autoumn", "winter"))  
data3$weekend <- factor(data3$weekend, levels = c("week", "weekend"))  
  
data\_model1 <- subset(data3, select = c(-month, - day, - area))  
data\_model2 <- data\_model1  
data\_model2[,c(3:10)] <- scale(data\_model1[, c(3:10)])

# 실험 1 scale을 하는게 좋을까? - 안한 친구 부터 확인  
library(caret)  
set.seed(12345)  
index <- createDataPartition(data\_model1$area1, p = 0.7, list = FALSE)  
train <- data\_model1[index, ]; test <- data\_model1[-index, ]  
  
#glm model1 saturated model  
model1 <- glm(area1 ~., data = train, family = binomial(link = "logit"))  
summary(model1)

##   
## Call:  
## glm(formula = area1 ~ ., family = binomial(link = "logit"), data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.672 -1.167 0.702 1.078 1.762   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -6.694735 3.190796 -2.098 0.03589 \*   
## X 0.059937 0.055879 1.073 0.28344   
## Y 0.013511 0.108423 0.125 0.90083   
## FFMC 0.043244 0.032879 1.315 0.18843   
## DMC -0.003608 0.002960 -1.219 0.22289   
## DC 0.002158 0.001422 1.518 0.12901   
## ISI -0.001565 0.037217 -0.042 0.96647   
## temp 0.053681 0.038571 1.392 0.16399   
## RH 0.004622 0.010582 0.437 0.66231   
## wind 0.138739 0.066191 2.096 0.03608 \*   
## rain -0.857958 1.261966 -0.680 0.49659   
## weathersummer -0.135917 0.792432 -0.172 0.86382   
## weatherautoumn -0.287873 0.938085 -0.307 0.75894   
## weatherwinter 1.800700 0.689111 2.613 0.00897 \*\*  
## weekendweekend -0.033404 0.231528 -0.144 0.88528   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 501.74 on 361 degrees of freedom  
## Residual deviance: 476.96 on 347 degrees of freedom  
## AIC: 506.96  
##   
## Number of Fisher Scoring iterations: 4

pred\_model1 <- as.numeric(predict(model1, newdata = test, type = "response") > 0.5)#response면 확률 출력  
result1 <- confusionMatrix(table(pred\_model1, test$area1))$overall[1]

#train, test set divide  
library(caret)  
set.seed(12345)  
index <- createDataPartition(data\_model2$area1, p = 0.7, list = FALSE)  
train <- data\_model2[index, ]; test <- data\_model2[-index, ]  
  
#glm model1 saturated model  
model2 <- glm(area1 ~., data = train, family = binomial(link = "logit"))  
summary(model2)

##   
## Call:  
## glm(formula = area1 ~ ., family = binomial(link = "logit"), data = train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.672 -1.167 0.702 1.078 1.762   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.249171 0.810092 -0.308 0.75840   
## X 0.059937 0.055879 1.073 0.28344   
## Y 0.013511 0.108423 0.125 0.90083   
## FFMC 0.238710 0.181495 1.315 0.18843   
## DMC -0.231101 0.189601 -1.219 0.22289   
## DC 0.535323 0.352644 1.518 0.12901   
## ISI -0.007133 0.169692 -0.042 0.96647   
## temp 0.311708 0.223966 1.392 0.16399   
## RH 0.075413 0.172678 0.437 0.66231   
## wind 0.248572 0.118591 2.096 0.03608 \*   
## rain -0.253921 0.373490 -0.680 0.49659   
## weathersummer -0.135917 0.792432 -0.172 0.86382   
## weatherautoumn -0.287873 0.938085 -0.307 0.75894   
## weatherwinter 1.800700 0.689111 2.613 0.00897 \*\*  
## weekendweekend -0.033404 0.231528 -0.144 0.88528   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 501.74 on 361 degrees of freedom  
## Residual deviance: 476.96 on 347 degrees of freedom  
## AIC: 506.96  
##   
## Number of Fisher Scoring iterations: 4

pred\_model2 <- as.numeric(predict(model2, newdata = test, type = "response") > 0.5)#response면 확률 출력  
result2 <- confusionMatrix(table(pred\_model2, test$area1))$overall[1]  
  
cat("sclae x", result1, "scale o", result2) # scale 쪽이 조금 더 나음

## sclae x 0.5225806 scale o 0.5225806

library(caret)  
set.seed(12345)  
index <- createDataPartition(data\_model2$area1, p = 0.7, list = FALSE)  
train <- data\_model2[index, ]; test <- data\_model2[-index, ]  
  
model\_full <- glm(area1~., data = train, family = "binomial")  
model\_non <- glm(area1~1, data = train, family = "binomial")  
  
model\_forward <- step(model\_non, list(lower=model\_non, upper=model\_full), direction = "forward")

## Start: AIC=503.74  
## area1 ~ 1  
##   
## Df Deviance AIC  
## + DC 1 495.58 499.58  
## + weather 3 491.64 499.64  
## + FFMC 1 498.31 502.31  
## + temp 1 498.61 502.61  
## + ISI 1 499.29 503.29  
## + wind 1 499.54 503.54  
## <none> 501.74 503.74  
## + RH 1 500.25 504.25  
## + DMC 1 500.54 504.54  
## + X 1 500.69 504.69  
## + Y 1 501.57 505.57  
## + weekend 1 501.62 505.62  
## + rain 1 501.72 505.72  
##   
## Step: AIC=499.58  
## area1 ~ DC  
##   
## Df Deviance AIC  
## + wind 1 491.16 497.16  
## <none> 495.58 499.58  
## + X 1 494.05 500.05  
## + weather 3 490.27 500.27  
## + FFMC 1 494.31 500.31  
## + RH 1 494.34 500.34  
## + ISI 1 494.61 500.61  
## + Y 1 494.98 500.98  
## + DMC 1 495.04 501.04  
## + temp 1 495.16 501.16  
## + weekend 1 495.52 501.52  
## + rain 1 495.52 501.52  
##   
## Step: AIC=497.16  
## area1 ~ DC + wind  
##   
## Df Deviance AIC  
## <none> 491.16 497.16  
## + RH 1 489.57 497.57  
## + X 1 489.62 497.62  
## + FFMC 1 489.96 497.96  
## + temp 1 490.05 498.05  
## + weather 3 486.39 498.39  
## + Y 1 490.43 498.43  
## + DMC 1 490.51 498.51  
## + ISI 1 490.65 498.65  
## + rain 1 490.90 498.90  
## + weekend 1 491.12 499.12

pred\_forward <- as.numeric(predict(model\_forward, newdata = test, type = "response") > 0.5)  
result1 <- confusionMatrix(table(pred\_forward, test$area1))$overall[1]  
  
model\_backward <- step(model\_full, list(lower=model\_non, upper=model\_full), direction = "backward")

## Start: AIC=506.96  
## area1 ~ X + Y + FFMC + DMC + DC + ISI + temp + RH + wind + rain +   
## weather + weekend  
##   
## Df Deviance AIC  
## - ISI 1 476.96 504.96  
## - Y 1 476.97 504.97  
## - weekend 1 476.98 504.98  
## - RH 1 477.15 505.15  
## - rain 1 477.41 505.41  
## - X 1 478.11 506.11  
## - DMC 1 478.45 506.45  
## - temp 1 478.91 506.91  
## <none> 476.96 506.96  
## - FFMC 1 479.19 507.19  
## - DC 1 479.30 507.30  
## - wind 1 481.43 509.43  
## - weather 3 485.83 509.83  
##   
## Step: AIC=504.96  
## area1 ~ X + Y + FFMC + DMC + DC + temp + RH + wind + rain + weather +   
## weekend  
##   
## Df Deviance AIC  
## - Y 1 476.97 502.97  
## - weekend 1 476.98 502.98  
## - RH 1 477.15 503.15  
## - rain 1 477.41 503.41  
## - X 1 478.11 504.11  
## - DMC 1 478.45 504.45  
## - temp 1 478.93 504.93  
## <none> 476.96 504.96  
## - DC 1 479.31 505.31  
## - FFMC 1 479.77 505.77  
## - wind 1 481.62 507.62  
## - weather 3 485.83 507.83  
##   
## Step: AIC=502.97  
## area1 ~ X + FFMC + DMC + DC + temp + RH + wind + rain + weather +   
## weekend  
##   
## Df Deviance AIC  
## - weekend 1 476.99 500.99  
## - RH 1 477.17 501.17  
## - rain 1 477.43 501.43  
## - DMC 1 478.45 502.45  
## - X 1 478.77 502.77  
## <none> 476.97 502.97  
## - temp 1 478.98 502.98  
## - DC 1 479.31 503.31  
## - FFMC 1 479.78 503.78  
## - wind 1 481.62 505.62  
## - weather 3 486.01 506.01  
##   
## Step: AIC=500.99  
## area1 ~ X + FFMC + DMC + DC + temp + RH + wind + rain + weather  
##   
## Df Deviance AIC  
## - RH 1 477.17 499.17  
## - rain 1 477.44 499.44  
## - DMC 1 478.48 500.48  
## - X 1 478.79 500.79  
## - temp 1 478.98 500.98  
## <none> 476.99 500.99  
## - DC 1 479.34 501.34  
## - FFMC 1 479.82 501.82  
## - wind 1 481.64 503.64  
## - weather 3 486.01 504.01  
##   
## Step: AIC=499.17  
## area1 ~ X + FFMC + DMC + DC + temp + wind + rain + weather  
##   
## Df Deviance AIC  
## - rain 1 477.52 497.52  
## - DMC 1 478.49 498.49  
## - X 1 479.03 499.03  
## <none> 477.17 499.17  
## - DC 1 479.39 499.39  
## - temp 1 479.62 499.62  
## - FFMC 1 479.83 499.83  
## - wind 1 481.74 501.74  
## - weather 3 486.07 502.07  
##   
## Step: AIC=497.52  
## area1 ~ X + FFMC + DMC + DC + temp + wind + weather  
##   
## Df Deviance AIC  
## - DMC 1 478.82 496.82  
## - X 1 479.29 497.29  
## <none> 477.52 497.52  
## - DC 1 479.62 497.62  
## - temp 1 479.98 497.98  
## - FFMC 1 480.16 498.16  
## - wind 1 481.86 499.86  
## - weather 3 486.37 500.37  
##   
## Step: AIC=496.82  
## area1 ~ X + FFMC + DC + temp + wind + weather  
##   
## Df Deviance AIC  
## - DC 1 479.74 495.74  
## - X 1 480.53 496.53  
## <none> 478.82 496.82  
## - FFMC 1 481.01 497.01  
## - temp 1 481.18 497.18  
## - wind 1 483.13 499.13  
## - weather 3 487.90 499.90  
##   
## Step: AIC=495.74  
## area1 ~ X + FFMC + temp + wind + weather  
##   
## Df Deviance AIC  
## - X 1 481.25 495.25  
## <none> 479.74 495.74  
## - temp 1 482.01 496.01  
## - FFMC 1 482.37 496.37  
## - wind 1 484.19 498.19  
## - weather 3 492.21 502.21  
##   
## Step: AIC=495.25  
## area1 ~ FFMC + temp + wind + weather  
##   
## Df Deviance AIC  
## <none> 481.25 495.25  
## - temp 1 483.36 495.36  
## - FFMC 1 483.95 495.95  
## - wind 1 485.72 497.72  
## - weather 3 493.46 501.46

pred\_backward <- as.numeric(predict(model\_backward, newdata = test, type = "response") > 0.5)  
result2 <- confusionMatrix(table(pred\_backward, test$area1))$overall[1]  
  
model\_step <- step(model\_non, list(lower=model\_non, upper=model\_full), direction = "both")

## Start: AIC=503.74  
## area1 ~ 1  
##   
## Df Deviance AIC  
## + DC 1 495.58 499.58  
## + weather 3 491.64 499.64  
## + FFMC 1 498.31 502.31  
## + temp 1 498.61 502.61  
## + ISI 1 499.29 503.29  
## + wind 1 499.54 503.54  
## <none> 501.74 503.74  
## + RH 1 500.25 504.25  
## + DMC 1 500.54 504.54  
## + X 1 500.69 504.69  
## + Y 1 501.57 505.57  
## + weekend 1 501.62 505.62  
## + rain 1 501.72 505.72  
##   
## Step: AIC=499.58  
## area1 ~ DC  
##   
## Df Deviance AIC  
## + wind 1 491.16 497.16  
## <none> 495.58 499.58  
## + X 1 494.05 500.05  
## + weather 3 490.27 500.27  
## + FFMC 1 494.31 500.31  
## + RH 1 494.34 500.34  
## + ISI 1 494.61 500.61  
## + Y 1 494.98 500.98  
## + DMC 1 495.04 501.04  
## + temp 1 495.16 501.16  
## + weekend 1 495.52 501.52  
## + rain 1 495.52 501.52  
## - DC 1 501.74 503.74  
##   
## Step: AIC=497.16  
## area1 ~ DC + wind  
##   
## Df Deviance AIC  
## <none> 491.16 497.16  
## + RH 1 489.57 497.57  
## + X 1 489.62 497.62  
## + FFMC 1 489.96 497.96  
## + temp 1 490.05 498.05  
## + weather 3 486.39 498.39  
## + Y 1 490.43 498.43  
## + DMC 1 490.51 498.51  
## + ISI 1 490.65 498.65  
## + rain 1 490.90 498.90  
## + weekend 1 491.12 499.12  
## - wind 1 495.58 499.58  
## - DC 1 499.54 503.54

pred\_step <- as.numeric(predict(model\_step, newdata = test, type = "response") > 0.5)  
result3 <- confusionMatrix(table(pred\_step, test$area1))$overall[1]  
  
cat("forward ", result1, "backward ", result2, "step ", result3)

## forward 0.5483871 backward 0.5483871 step 0.5483871

#svm  
library(e1071)  
set.seed(12345)  
index <- createDataPartition(data\_model2$area1, p = 0.7, list = FALSE)  
train <- data\_model2[index, ]; test <- data\_model2[-index, ]  
  
svm.model<-svm(area1~.,data=train, probability=TRUE)  
summary(svm.model)

##   
## Call:  
## svm(formula = area1 ~ ., data = train, probability = TRUE)  
##   
##   
## Parameters:  
## SVM-Type: eps-regression   
## SVM-Kernel: radial   
## cost: 1   
## gamma: 0.06666667   
## epsilon: 0.1   
##   
## Sigma: 0.9143796   
##   
##   
## Number of Support Vectors: 337

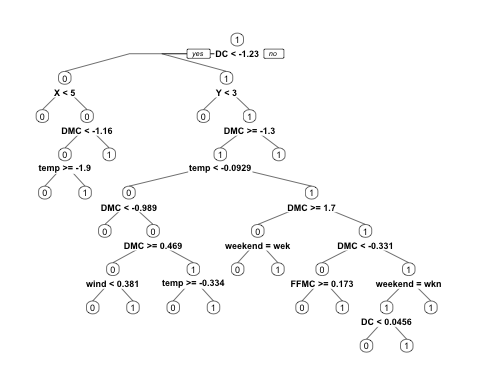
predicted.prob<- as.numeric(predict(svm.model,newdata=test) > 0.5)  
confusionMatrix(table(predicted.prob, test$area1))$overall[1]

## Accuracy   
## 0.5741935

#Decision Tree  
library(rpart)  
set.seed(12345)  
index <- createDataPartition(data\_model2$area1, p = 0.7, list = FALSE)  
train <- data\_model2[index, ]; test <- data\_model2[-index, ]  
rpartmodel1 <- rpart(factor(area1)~., data=train)  
rpartmodel1

## n= 362   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 362 178 1 (0.4917127 0.5082873)   
## 2) DC< -1.228463 58 18 0 (0.6896552 0.3103448)   
## 4) X< 4.5 28 4 0 (0.8571429 0.1428571) \*  
## 5) X>=4.5 30 14 0 (0.5333333 0.4666667)   
## 10) DMC< -1.158102 21 7 0 (0.6666667 0.3333333)   
## 20) temp>=-1.901133 14 2 0 (0.8571429 0.1428571) \*  
## 21) temp< -1.901133 7 2 1 (0.2857143 0.7142857) \*  
## 11) DMC>=-1.158102 9 2 1 (0.2222222 0.7777778) \*  
## 3) DC>=-1.228463 304 138 1 (0.4539474 0.5460526)   
## 6) Y< 2.5 29 9 0 (0.6896552 0.3103448) \*  
## 7) Y>=2.5 275 118 1 (0.4290909 0.5709091)   
## 14) DMC>=-1.304089 267 118 1 (0.4419476 0.5580524)   
## 28) temp< -0.09285398 80 35 0 (0.5625000 0.4375000)   
## 56) DMC< -0.9894742 7 0 0 (1.0000000 0.0000000) \*  
## 57) DMC>=-0.9894742 73 35 0 (0.5205479 0.4794521)   
## 114) DMC>=0.4688417 22 6 0 (0.7272727 0.2727273)   
## 228) wind< 0.3808765 12 0 0 (1.0000000 0.0000000) \*  
## 229) wind>=0.3808765 10 4 1 (0.4000000 0.6000000) \*  
## 115) DMC< 0.4688417 51 22 1 (0.4313725 0.5686275)   
## 230) temp>=-0.3339579 20 7 0 (0.6500000 0.3500000) \*  
## 231) temp< -0.3339579 31 9 1 (0.2903226 0.7096774) \*  
## 29) temp>=-0.09285398 187 73 1 (0.3903743 0.6096257)   
## 58) DMC>=1.704663 20 8 0 (0.6000000 0.4000000)   
## 116) weekend=week 10 2 0 (0.8000000 0.2000000) \*  
## 117) weekend=weekend 10 4 1 (0.4000000 0.6000000) \*  
## 59) DMC< 1.704663 167 61 1 (0.3652695 0.6347305)   
## 118) DMC< -0.3313584 29 13 0 (0.5517241 0.4482759)   
## 236) FFMC>=0.1730616 14 3 0 (0.7857143 0.2142857) \*  
## 237) FFMC< 0.1730616 15 5 1 (0.3333333 0.6666667) \*  
## 119) DMC>=-0.3313584 138 45 1 (0.3260870 0.6739130)   
## 238) weekend=weekend 49 22 1 (0.4489796 0.5510204)   
## 476) DC< 0.04559251 9 2 0 (0.7777778 0.2222222) \*  
## 477) DC>=0.04559251 40 15 1 (0.3750000 0.6250000) \*  
## 239) weekend=week 89 23 1 (0.2584270 0.7415730) \*  
## 15) DMC< -1.304089 8 0 1 (0.0000000 1.0000000) \*

library(rpart.plot)  
prp(rpartmodel1, type=2, digits=3) #rpart전용 plot #Class models: Classification rate (ncorrect/nobservations)



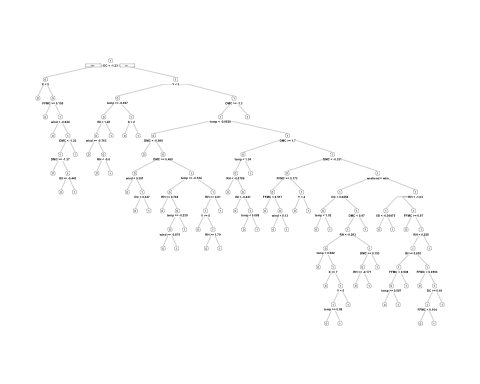
rpart\_pred1 <- predict(rpartmodel1, newdata = subset(test, select = c(- area1)), type="class")  
confusionMatrix(rpart\_pred1, factor(test$area1))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 35 32  
## 1 34 54  
##   
## Accuracy : 0.5742   
## 95% CI : (0.4923, 0.6532)  
## No Information Rate : 0.5548   
## P-Value [Acc > NIR] : 0.3441   
##   
## Kappa : 0.1355   
## Mcnemar's Test P-Value : 0.9020   
##   
## Sensitivity : 0.5072   
## Specificity : 0.6279   
## Pos Pred Value : 0.5224   
## Neg Pred Value : 0.6136   
## Prevalence : 0.4452   
## Detection Rate : 0.2258   
## Detection Prevalence : 0.4323   
## Balanced Accuracy : 0.5676   
##   
## 'Positive' Class : 0   
##

#minsplit,cp 조정  
set.seed(12345)  
rpartmodel4 <- rpart(factor(area1)~.,data=train,minsplit=6,cp=0.005)  
rpartmodel4

## n= 362   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 362 178 1 (0.49171271 0.50828729)   
## 2) DC< -1.228463 58 18 0 (0.68965517 0.31034483)   
## 4) X< 4.5 28 4 0 (0.85714286 0.14285714) \*  
## 5) X>=4.5 30 14 0 (0.53333333 0.46666667)   
## 10) FFMC>=0.154946 5 0 0 (1.00000000 0.00000000) \*  
## 11) FFMC< 0.154946 25 11 1 (0.44000000 0.56000000)   
## 22) wind< -0.6237825 5 0 0 (1.00000000 0.00000000) \*  
## 23) wind>=-0.6237825 20 6 1 (0.30000000 0.70000000)   
## 46) DMC< -1.218995 13 6 1 (0.46153846 0.53846154)   
## 92) DMC>=-1.365763 3 0 0 (1.00000000 0.00000000) \*  
## 93) DMC< -1.365763 10 3 1 (0.30000000 0.70000000)   
## 186) ISI>=-0.4433981 5 2 0 (0.60000000 0.40000000) \*  
## 187) ISI< -0.4433981 5 0 1 (0.00000000 1.00000000) \*  
## 47) DMC>=-1.218995 7 0 1 (0.00000000 1.00000000) \*  
## 3) DC>=-1.228463 304 138 1 (0.45394737 0.54605263)   
## 6) Y< 2.5 29 9 0 (0.68965517 0.31034483)   
## 12) temp>=-0.06702142 22 5 0 (0.77272727 0.22727273)   
## 24) ISI< 1.453749 19 3 0 (0.84210526 0.15789474)   
## 48) wind>=-0.7633185 11 0 0 (1.00000000 0.00000000) \*  
## 49) wind< -0.7633185 8 3 0 (0.62500000 0.37500000)   
## 98) RH< -0.5998602 4 0 0 (1.00000000 0.00000000) \*  
## 99) RH>=-0.5998602 4 1 1 (0.25000000 0.75000000) \*  
## 25) ISI>=1.453749 3 1 1 (0.33333333 0.66666667) \*  
## 13) temp< -0.06702142 7 3 1 (0.42857143 0.57142857)   
## 26) X< 1.5 4 1 0 (0.75000000 0.25000000) \*  
## 27) X>=1.5 3 0 1 (0.00000000 1.00000000) \*  
## 7) Y>=2.5 275 118 1 (0.42909091 0.57090909)   
## 14) DMC>=-1.304089 267 118 1 (0.44194757 0.55805243)   
## 28) temp< -0.09285398 80 35 0 (0.56250000 0.43750000)   
## 56) DMC< -0.9894742 7 0 0 (1.00000000 0.00000000) \*  
## 57) DMC>=-0.9894742 73 35 0 (0.52054795 0.47945205)   
## 114) DMC>=0.4688417 22 6 0 (0.72727273 0.27272727)   
## 228) wind< 0.3808765 12 0 0 (1.00000000 0.00000000) \*  
## 229) wind>=0.3808765 10 4 1 (0.40000000 0.60000000)   
## 458) DC< 0.326969 3 0 0 (1.00000000 0.00000000) \*  
## 459) DC>=0.326969 7 1 1 (0.14285714 0.85714286) \*  
## 115) DMC< 0.4688417 51 22 1 (0.43137255 0.56862745)   
## 230) temp>=-0.3339579 20 7 0 (0.65000000 0.35000000)   
## 460) RH>=0.748388 6 0 0 (1.00000000 0.00000000) \*  
## 461) RH< 0.748388 14 7 0 (0.50000000 0.50000000)   
## 922) temp>=-0.2392385 11 4 0 (0.63636364 0.36363636)   
## 1844) wind>=-0.8749473 8 1 0 (0.87500000 0.12500000) \*  
## 1845) wind< -0.8749473 3 0 1 (0.00000000 1.00000000) \*  
## 923) temp< -0.2392385 3 0 1 (0.00000000 1.00000000) \*  
## 231) temp< -0.3339579 31 9 1 (0.29032258 0.70967742)   
## 462) RH>=0.8096721 16 7 1 (0.43750000 0.56250000)   
## 924) Y>=4.5 3 0 0 (1.00000000 0.00000000) \*  
## 925) Y< 4.5 13 4 1 (0.30769231 0.69230769)   
## 1850) RH>=1.790216 5 2 0 (0.60000000 0.40000000) \*  
## 1851) RH< 1.790216 8 1 1 (0.12500000 0.87500000) \*  
## 463) RH< 0.8096721 15 2 1 (0.13333333 0.86666667) \*  
## 29) temp>=-0.09285398 187 73 1 (0.39037433 0.60962567)   
## 58) DMC>=1.704663 20 8 0 (0.60000000 0.40000000)   
## 116) temp< 1.336548 17 5 0 (0.70588235 0.29411765)   
## 232) RH< -0.07894614 6 0 0 (1.00000000 0.00000000) \*  
## 233) RH>=-0.07894614 11 5 0 (0.54545455 0.45454545)   
## 466) ISI< -0.4214657 3 0 0 (1.00000000 0.00000000) \*  
## 467) ISI>=-0.4214657 8 3 1 (0.37500000 0.62500000)   
## 934) temp< 0.08797394 3 1 0 (0.66666667 0.33333333) \*  
## 935) temp>=0.08797394 5 1 1 (0.20000000 0.80000000) \*  
## 117) temp>=1.336548 3 0 1 (0.00000000 1.00000000) \*  
## 59) DMC< 1.704663 167 61 1 (0.36526946 0.63473054)   
## 118) DMC< -0.3313584 29 13 0 (0.55172414 0.44827586)   
## 236) FFMC>=0.1730616 14 3 0 (0.78571429 0.21428571)   
## 472) FFMC< 0.5172576 7 0 0 (1.00000000 0.00000000) \*  
## 473) FFMC>=0.5172576 7 3 0 (0.57142857 0.42857143)   
## 946) wind< 0.1297118 3 0 0 (1.00000000 0.00000000) \*  
## 947) wind>=0.1297118 4 1 1 (0.25000000 0.75000000) \*  
## 237) FFMC< 0.1730616 15 5 1 (0.33333333 0.66666667)   
## 474) Y< 3.5 2 0 0 (1.00000000 0.00000000) \*  
## 475) Y>=3.5 13 3 1 (0.23076923 0.76923077) \*  
## 119) DMC>=-0.3313584 138 45 1 (0.32608696 0.67391304)   
## 238) weekend=weekend 49 22 1 (0.44897959 0.55102041)   
## 476) DC< 0.04559251 9 2 0 (0.77777778 0.22222222)   
## 952) temp< 1.017946 7 0 0 (1.00000000 0.00000000) \*  
## 953) temp>=1.017946 2 0 1 (0.00000000 1.00000000) \*  
## 477) DC>=0.04559251 40 15 1 (0.37500000 0.62500000)   
## 954) DMC< 0.9700402 33 15 1 (0.45454545 0.54545455)   
## 1908) RH< -0.2934402 21 9 0 (0.57142857 0.42857143)   
## 3816) temp< 0.6821228 7 1 0 (0.85714286 0.14285714) \*  
## 3817) temp>=0.6821228 14 6 1 (0.42857143 0.57142857)   
## 7634) X>=6.5 2 0 0 (1.00000000 0.00000000) \*  
## 7635) X< 6.5 12 4 1 (0.33333333 0.66666667)   
## 15270) Y< 4.5 9 4 1 (0.44444444 0.55555556)   
## 30540) temp>=0.8801725 6 2 0 (0.66666667 0.33333333) \*  
## 30541) temp< 0.8801725 3 0 1 (0.00000000 1.00000000) \*  
## 15271) Y>=4.5 3 0 1 (0.00000000 1.00000000) \*  
## 1909) RH>=-0.2934402 12 3 1 (0.25000000 0.75000000)   
## 3818) DMC>=0.1526651 6 3 0 (0.50000000 0.50000000)   
## 7636) RH>=-0.1708722 3 0 0 (1.00000000 0.00000000) \*  
## 7637) RH< -0.1708722 3 0 1 (0.00000000 1.00000000) \*  
## 3819) DMC< 0.1526651 6 0 1 (0.00000000 1.00000000) \*  
## 955) DMC>=0.9700402 7 0 1 (0.00000000 1.00000000) \*  
## 239) weekend=week 89 23 1 (0.25842697 0.74157303)   
## 478) RH< -1.028848 15 7 1 (0.46666667 0.53333333)   
## 956) ISI< -0.0047513 6 1 0 (0.83333333 0.16666667) \*  
## 957) ISI>=-0.0047513 9 2 1 (0.22222222 0.77777778) \*  
## 479) RH>=-1.028848 74 16 1 (0.21621622 0.78378378)   
## 958) FFMC>=0.970147 2 0 0 (1.00000000 0.00000000) \*  
## 959) FFMC< 0.970147 72 14 1 (0.19444444 0.80555556)   
## 1918) RH< 0.2581159 61 14 1 (0.22950820 0.77049180)   
## 3836) ISI>=0.6532189 20 8 1 (0.40000000 0.60000000)   
## 7672) FFMC< 0.5081998 9 3 0 (0.66666667 0.33333333)   
## 15344) temp>=0.5874034 4 0 0 (1.00000000 0.00000000) \*  
## 15345) temp< 0.5874034 5 2 1 (0.40000000 0.60000000) \*  
## 7673) FFMC>=0.5081998 11 2 1 (0.18181818 0.81818182) \*  
## 3837) ISI< 0.6532189 41 6 1 (0.14634146 0.85365854)   
## 7674) FFMC< 0.05531033 3 1 0 (0.66666667 0.33333333) \*  
## 7675) FFMC>=0.05531033 38 4 1 (0.10526316 0.89473684)   
## 15350) DC>=0.8103078 12 3 1 (0.25000000 0.75000000)   
## 30700) FFMC< 0.1640038 2 0 0 (1.00000000 0.00000000) \*  
## 30701) FFMC>=0.1640038 10 1 1 (0.10000000 0.90000000) \*  
## 15351) DC< 0.8103078 26 1 1 (0.03846154 0.96153846) \*  
## 1919) RH>=0.2581159 11 0 1 (0.00000000 1.00000000) \*  
## 15) DMC< -1.304089 8 0 1 (0.00000000 1.00000000) \*

prp(rpartmodel4 , type=2, digits=3) #cp값을 낮추니 복잡해짐



rpart\_pred4 <- predict(rpartmodel4,subset(test,select=-area1),type="class")  
confusionMatrix(rpart\_pred4,factor(test$area1))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 46 42  
## 1 23 44  
##   
## Accuracy : 0.5806   
## 95% CI : (0.4988, 0.6593)  
## No Information Rate : 0.5548   
## P-Value [Acc > NIR] : 0.28657   
##   
## Kappa : 0.1736   
## Mcnemar's Test P-Value : 0.02557   
##   
## Sensitivity : 0.6667   
## Specificity : 0.5116   
## Pos Pred Value : 0.5227   
## Neg Pred Value : 0.6567   
## Prevalence : 0.4452   
## Detection Rate : 0.2968   
## Detection Prevalence : 0.5677   
## Balanced Accuracy : 0.5891   
##   
## 'Positive' Class : 0   
##

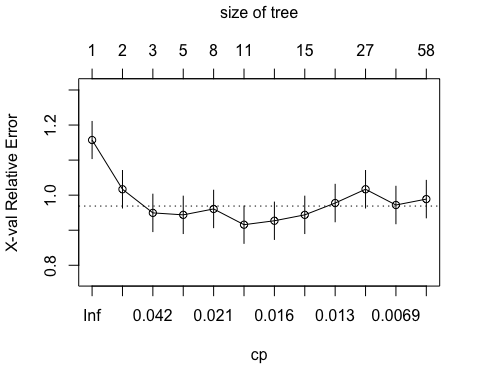
rpart\_pred4\_train <- predict(rpartmodel4,subset(train,select=-area1),type="class")  
confusionMatrix(rpart\_pred4\_train,factor(train$area1))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 159 16  
## 1 19 168  
##   
## Accuracy : 0.9033   
## 95% CI : (0.8681, 0.9317)  
## No Information Rate : 0.5083   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.8065   
## Mcnemar's Test P-Value : 0.7353   
##   
## Sensitivity : 0.8933   
## Specificity : 0.9130   
## Pos Pred Value : 0.9086   
## Neg Pred Value : 0.8984   
## Prevalence : 0.4917   
## Detection Rate : 0.4392   
## Detection Prevalence : 0.4834   
## Balanced Accuracy : 0.9032   
##   
## 'Positive' Class : 0   
##

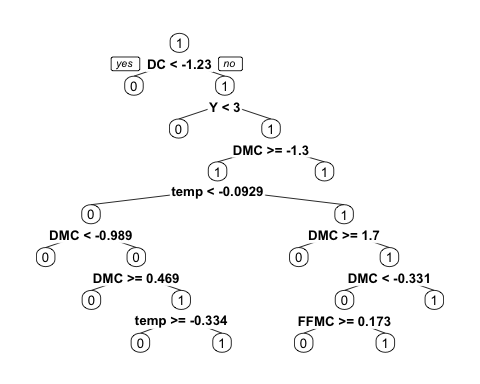
# overfit이 생긴다!  
  
########################################## pruning ##################################################  
  
printcp(rpartmodel4) # == rpartmodel4$cptable

##   
## Classification tree:  
## rpart(formula = factor(area1) ~ ., data = train, minsplit = 6,   
## cp = 0.005)  
##   
## Variables actually used in tree construction:  
## [1] DC DMC FFMC ISI RH temp weekend wind   
## [9] X Y   
##   
## Root node error: 178/362 = 0.49171  
##   
## n= 362   
##   
## CP nsplit rel error xerror xstd  
## 1 0.123596 0 1.00000 1.15730 0.052932  
## 2 0.061798 1 0.87640 1.01685 0.053445  
## 3 0.028090 2 0.81461 0.94944 0.053327  
## 4 0.022472 4 0.75843 0.94382 0.053307  
## 5 0.019663 7 0.69101 0.96067 0.053363  
## 6 0.016854 10 0.61798 0.91573 0.053180  
## 7 0.014981 11 0.60112 0.92697 0.053235  
## 8 0.014045 14 0.55618 0.94382 0.053307  
## 9 0.011236 18 0.50000 0.97753 0.053405  
## 10 0.008427 26 0.41011 1.01685 0.053445  
## 11 0.005618 38 0.30337 0.97191 0.053392  
## 12 0.005000 57 0.19663 0.98876 0.053424

plotcp(rpartmodel4)



rpartmodel4\_prune <- prune(rpartmodel4,  
 cp=rpartmodel4$cptable[which.min(rpartmodel4$cptable[,"xerror"])],"CP")  
prp(rpartmodel4\_prune, type=2, digits=3)



rpart\_pred4\_prune <- predict(rpartmodel4\_prune,subset(test,select=-area1),type="class") #test accuracy  
confusionMatrix(rpart\_pred4\_prune,factor(test$area1))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 39 42  
## 1 30 44  
##   
## Accuracy : 0.5355   
## 95% CI : (0.4537, 0.6159)  
## No Information Rate : 0.5548   
## P-Value [Acc > NIR] : 0.7148   
##   
## Kappa : 0.0755   
## Mcnemar's Test P-Value : 0.1949   
##   
## Sensitivity : 0.5652   
## Specificity : 0.5116   
## Pos Pred Value : 0.4815   
## Neg Pred Value : 0.5946   
## Prevalence : 0.4452   
## Detection Rate : 0.2516   
## Detection Prevalence : 0.5226   
## Balanced Accuracy : 0.5384   
##   
## 'Positive' Class : 0   
##

#accuracy : 0.5677 tree가 간단해졌는데 acc는 올랐다.  
rpart\_pred4\_prune\_train <- predict(rpartmodel4\_prune,subset(train,select=-area1),type="class") #train accuracy  
confusionMatrix(rpart\_pred4\_prune\_train, factor(train$area1))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 119 51  
## 1 59 133  
##   
## Accuracy : 0.6961   
## 95% CI : (0.6459, 0.7431)  
## No Information Rate : 0.5083   
## P-Value [Acc > NIR] : 3.078e-13   
##   
## Kappa : 0.3917   
## Mcnemar's Test P-Value : 0.5045   
##   
## Sensitivity : 0.6685   
## Specificity : 0.7228   
## Pos Pred Value : 0.7000   
## Neg Pred Value : 0.6927   
## Prevalence : 0.4917   
## Detection Rate : 0.3287   
## Detection Prevalence : 0.4696   
## Balanced Accuracy : 0.6957   
##   
## 'Positive' Class : 0   
##

#xgboost  
set.seed(12345)  
index <- createDataPartition(data\_model2$area1, p = 0.7, list = FALSE)  
train <- data\_model2[index, ]; test <- data\_model2[-index, ]  
  
library(xgboost)

##   
## Attaching package: 'xgboost'

## The following object is masked from 'package:dplyr':  
##   
## slice

model.xg <- xgboost(data = data.matrix(train[,-13]), label = train[,13], max\_depth = 4, nrounds = 30, objective = "binary:logistic" )

## [1] train-error:0.328729   
## [2] train-error:0.290055   
## [3] train-error:0.281768   
## [4] train-error:0.243094   
## [5] train-error:0.209945   
## [6] train-error:0.209945   
## [7] train-error:0.196133   
## [8] train-error:0.190608   
## [9] train-error:0.182320   
## [10] train-error:0.174033   
## [11] train-error:0.157459   
## [12] train-error:0.143646   
## [13] train-error:0.132597   
## [14] train-error:0.127072   
## [15] train-error:0.116022   
## [16] train-error:0.116022   
## [17] train-error:0.110497   
## [18] train-error:0.096685   
## [19] train-error:0.096685   
## [20] train-error:0.096685   
## [21] train-error:0.091160   
## [22] train-error:0.088398   
## [23] train-error:0.077348   
## [24] train-error:0.069061   
## [25] train-error:0.069061   
## [26] train-error:0.069061   
## [27] train-error:0.060773   
## [28] train-error:0.060773   
## [29] train-error:0.058011   
## [30] train-error:0.058011

pred <- as.numeric(predict(model.xg, data.matrix(test[,-13])) > 0.5)  
confusionMatrix(table(pred, test$area1))

## Confusion Matrix and Statistics  
##   
##   
## pred 0 1  
## 0 37 33  
## 1 32 53  
##   
## Accuracy : 0.5806   
## 95% CI : (0.4988, 0.6593)  
## No Information Rate : 0.5548   
## P-Value [Acc > NIR] : 0.2866   
##   
## Kappa : 0.1523   
## Mcnemar's Test P-Value : 1.0000   
##   
## Sensitivity : 0.5362   
## Specificity : 0.6163   
## Pos Pred Value : 0.5286   
## Neg Pred Value : 0.6235   
## Prevalence : 0.4452   
## Detection Rate : 0.2387   
## Detection Prevalence : 0.4516   
## Balanced Accuracy : 0.5763   
##   
## 'Positive' Class : 0   
##

#Random Forest  
set.seed(12345)  
index <- createDataPartition(data\_model2$area1, p = 0.7, list = FALSE)  
train <- data\_model2[index, ]; test <- data\_model2[-index, ]  
  
library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':  
##   
## combine

## The following object is masked from 'package:ggplot2':  
##   
## margin

model.rf <- randomForest(factor(area1) ~., data = train, proximity = T)  
round(importance(model.rf))

## MeanDecreaseGini  
## X 15  
## Y 11  
## FFMC 17  
## DMC 22  
## DC 20  
## ISI 18  
## temp 27  
## RH 22  
## wind 18  
## rain 1  
## weather 4  
## weekend 3

pred.rf <- predict(model.rf, test[,-13])  
confusionMatrix(table(pred.rf, test$area1))

## Confusion Matrix and Statistics  
##   
##   
## pred.rf 0 1  
## 0 40 28  
## 1 29 58  
##   
## Accuracy : 0.6323   
## 95% CI : (0.5512, 0.7082)  
## No Information Rate : 0.5548   
## P-Value [Acc > NIR] : 0.03084   
##   
## Kappa : 0.2545   
## Mcnemar's Test P-Value : 1.00000   
##   
## Sensitivity : 0.5797   
## Specificity : 0.6744   
## Pos Pred Value : 0.5882   
## Neg Pred Value : 0.6667   
## Prevalence : 0.4452   
## Detection Rate : 0.2581   
## Detection Prevalence : 0.4387   
## Balanced Accuracy : 0.6271   
##   
## 'Positive' Class : 0   
##

**분석 방안**

이진(binary) 변수로 표현한 ‘is\_fire’변수를 예측하기 위해 제시한 접근 방법은 Logistic Regression(LR), Decision Tree(DT), Random Forest(RF), XG Boost로 총 네 가지이다.

먼저 각 방법의 산불 여부 예측 정확도는 상기한 순서대로 54%, 69.6%, 63%, 58%가 나왔다. 예상과는 달리 복잡한 모델을 사용했을 때보다 가장 간단한 모델인 DT를 사용했을 때 제일 높은 정확도를 얻을 수 있었다. 직관적으로는 단순한 모델로도 큰 효과를 낼 수 있다는 결과를 얻었다고 생각할 수 있지만, 전체적인 예측률이 만족할 수준에 크게 미치지 못하기 때문에 개선해야 할 점이 많다고 할 수 있다.

**향후 개선 방안**

모델을 여러개 사용해봤지만 성능이 크게 개선되지 않았다. 이유는 더 다른 다양한 모델을 사용하지 않은것도 있을 수 있지만 데이터 전처리가 부족했기 떄문이라고 여겨진다.

1. 변수에 가중치를 준다든가 영향을 주지 않는 변수를 제거하는 것을 더 생각해봐야 할 것 같다.

2. FWI의 일종인 BUI 변수를 DMC와 DC로 계산해낼 수 있기에 계산하여 추가하는 방안도 있을 것이다.

3. 필요없는 변수를 적절히 처리하고 주어진 변수 이외에 화재에 영향을 줄만한 번개 등의 다른 변수들을 추가하는 방안도 생각해봐야 할 것이다.

4. 독립변수에서 random effect와 fixed effect를 식별해내어 glmm모델을 사용하는 것도 고려해봐야 할 사항이다.