# 데이터마이닝 2018 기말 프로젝트 제출 파일

#

# - 아래 파일의 내용을 고쳐 최종 예측을 하는 코드를 작성할 것

# - 이 파일의 파일명은 "이름.R" 형태로 제출

# - 예측 파일은 제출하지 않음

# - 작업 디렉토리 설정 후 전체 코드를 실행했을 때 에러 없이 예측 파일이 생성되어야 함

# - 다른 사람의 코드 또는 아이디어를를 베끼면 양측 모두 0점 처리

# 이름 (자기 이름으로 고쳐 넣을 것)

name = '안상준'

# 데이터 불러오기 (수정하지 말것)

df <- read.csv('apt.csv') # 학습용 데이터

test <- read.csv('mid\_X.csv') # 테스트용 데이터(실제 테스트는 중간테스트 데이터와 파일명은 같고 내용은 다름)

# 학습 및 예측 (수정하여 자신의 모형을 학습시킬 것)

if(!require(caret)) install.packages("caret") ; library(caret)

if(!require(mice)) install.packages("mice") ; library(mice)

if(!require(Amelia)) install.packages("Amelia") ; library(Amelia)

data <- df[,c(-1, -5, -12, -20, -25, -49)]

test <- test[,c(-4, -11, -19, -24, -48)]

levels(data[,1]) <- c(NA, "A", "M", "S", "T")

levels(data[,24]) <- c(NA, "N", "Y")

levels(data[,28]) <- c(NA, "CHP", "LNG", "LPG", "OIL")

levels(data[,29]) <- c(NA, "C", "I", "R")

data\_factor <- data[, c(1, 24, 28, 29)] ; head(data\_factor, 20) # Factor

data\_numeric <- data[, c(-1, -24, -28, -29)] ; head(data\_numeric, 20) # Numeric

# Factor & Numeric - Test

levels(test[,1]) <- c(NA, "A", "M", "S", "T")

levels(test[,24]) <- c(NA, "N", "Y")

levels(test[,28]) <- c(NA, "CHP", "LNG", "LPG", "OIL")

levels(test[,29]) <- c(NA, "C", "I", "R")

test\_factor <- test[, c(1, 24, 28, 29)] ; head(test\_factor, 20) # Factor

test\_numeric <- test[, c(-1, -24, -28, -29)] ; head(test\_numeric, 20) # Numeric

########################################################################################

# NA

set.seed(1130)

# Factor

library(mice)

data\_factor\_NA <- mice(data\_factor, 5)

df1 <- complete(data\_factor\_NA, 1) ; head(df1, 20)

df2 <- complete(data\_factor\_NA, 2) ; head(df2, 20)

df3 <- complete(data\_factor\_NA, 3) ; head(df3, 20)

df4 <- complete(data\_factor\_NA, 4) ; head(df4, 20)

df5 <- complete(data\_factor\_NA, 5) ; head(df5, 20)

test\_factor\_NA <- mice(test\_factor, 5)

tdf1 <- complete(test\_factor\_NA, 1) ; head(tdf1, 20)

tdf2 <- complete(test\_factor\_NA, 2) ; head(tdf2, 20)

tdf3 <- complete(test\_factor\_NA, 3) ; head(tdf3, 20)

tdf4 <- complete(test\_factor\_NA, 4) ; head(tdf4, 20)

tdf5 <- complete(test\_factor\_NA, 5) ; head(tdf5, 20)

########################################################################################

# Numeric

library(Amelia)

data\_numeric\_NA <- amelia(data\_numeric, m = 5)

adf1 = data\_numeric\_NA$imputations[[1]] ; adf1[adf1 < 0] = 0 ; head(adf1)

adf2 = data\_numeric\_NA$imputations[[2]] ; adf2[adf2 < 0] = 0 ; head(adf2)

adf3 = data\_numeric\_NA$imputations[[3]] ; adf3[adf3 < 0] = 0 ; head(adf3)

adf4 = data\_numeric\_NA$imputations[[4]] ; adf4[adf4 < 0] = 0 ; head(adf4)

adf5 = data\_numeric\_NA$imputations[[5]] ; adf5[adf5 < 0] = 0 ; head(adf5)

test\_numeric\_NA <- amelia(test\_numeric, m = 5)

tadf1 = test\_numeric\_NA$imputations[[1]] ; tadf1[tadf1 < 0] = 0 ; head(tadf1)

tadf2 = test\_numeric\_NA$imputations[[2]] ; tadf2[tadf2 < 0] = 0 ; head(tadf2)

tadf3 = test\_numeric\_NA$imputations[[3]] ; tadf3[tadf3 < 0] = 0 ; head(tadf3)

tadf4 = test\_numeric\_NA$imputations[[4]] ; tadf4[tadf4 < 0] = 0 ; head(tadf4)

tadf5 = test\_numeric\_NA$imputations[[5]] ; tadf5[tadf5 < 0] = 0 ; head(tadf5)

########################################################################################

# Merge

data1 <- cbind(df1, adf1) ; test1 <- cbind(tdf1, tadf1)

data2 <- cbind(df2, adf2) ; test2 <- cbind(tdf2, tadf2)

data3 <- cbind(df3, adf3) ; test3 <- cbind(tdf3, tadf3)

data4 <- cbind(df4, adf4) ; test4 <- cbind(tdf4, tadf4)

data5 <- cbind(df5, adf5) ; test5 <- cbind(tdf5, tadf5)

########################################################################################

# Preprocess - Train

data <- data4

# Outlier

w1 <- which(data[,6] == 651.9200)

w2 <- which(data[,27] == 249024.5600)

w3 <- which(data[,27] == 201802.7800)

w4 <- which(data[,28] == 2222)

w5 <- which(data[,28] == 1113)

data <- data[-c(w1, w2, w3, w4, w5),]

# Log

for (i in c(5, 27, 28, 29, 32, 36, 38, 39, 40, 41, 42, 44))(

data[,i] = log(data[,i]))

for (i in c(5, 27, 28, 29, 32, 36, 38, 39, 40, 41, 42, 44))(

data[,i][is.infinite(data[,i])] = 0

)

# Preprocess - Test

test <- test4

# Outlier

t1 <- which(test[,27] == 249024.5600)

t2 <- which(test[,27] == 201802.7800)

# Log

for (i in c(5, 27, 28, 29, 32, 36, 38, 39, 40, 41, 42))(

test[,i] <- log(test[,i])

)

for (i in c(5, 27, 28, 29, 32, 36, 38, 39, 40, 41, 42))(

test[,i][is.infinite(test[,i])] = 0

)

########################################################################################

# Setting

set.seed(1130)

library(caret)

control <- trainControl(method = 'cv', search = 'grid', number = 5)

#2500

xgb.grid = expand.grid(.nrounds = 2500,.max\_depth = 8,.eta = 0.1,.gamma = 0,.colsample\_bytree = 1,.min\_child\_weight = 1,.subsample = 0.8)

xgb.model <- train(price ~ .,data= data,tuneGrid = xgb.grid,tuneLength = 1,trControl = control,method = 'xgbTree')

price.xgb1 <- predict(xgb.model, test)

#1000

xgb.grid = expand.grid(.nrounds = 1000,.max\_depth = 7,.eta = 0.1,.gamma = 0,.colsample\_bytree = 1,.min\_child\_weight = 0.5,.subsample = 1)

xgb.model <- train(price ~ .,data = data,tuneGrid = xgb.grid,tuneLength = 1,trControl = control,method = 'xgbTree')

price.xgb2 <- predict(xgb.model, test)

#1200

xgb.grid = expand.grid(.nrounds = 1200,.max\_depth = 7,.eta = 0.05,.gamma = 0,.colsample\_bytree = 1,.min\_child\_weight = 0.75,.subsample = 0.8)

xgb.model <- train(price ~ .,data = data,tuneGrid = xgb.grid,tuneLength = 1,trControl = control,method = 'xgbTree')

price.xgb3 <- predict(xgb.model, test)

#1500

xgb.grid = expand.grid(.nrounds = 1500,.max\_depth = 7,.eta = 0.05,.gamma = 0,.colsample\_bytree = 1,.min\_child\_weight = 0.75,.subsample = 0.7)

xgb.model <- train(price ~ .,data = data,tuneGrid = xgb.grid,tuneLength = 1,trControl = control,method = 'xgbTree')

price.xgb4 <- predict(xgb.model, test)

#1700

xgb.grid = expand.grid(.nrounds = 1700,.max\_depth = 7,.eta = 0.05,.gamma = 0,.colsample\_bytree = 1,.min\_child\_weight = 1,.subsample = 0.8)

xgb.model <- train(price ~ .,data = data,tuneGrid = xgb.grid,tuneLength = 1,trControl = control,method = 'xgbTree')

price.xgb5 <- predict(xgb.model, test)

#2000

xgb.grid = expand.grid(.nrounds = 2000,.max\_depth = 8,.eta = 0.05,.gamma = 0,.colsample\_bytree = 1,.min\_child\_weight = 1,.subsample = 0.8)

xgb.model <- train(price ~ .,data = data,tuneGrid = xgb.grid,tuneLength = 1,trControl = control,method = 'xgbTree')

price.xgb6 <- predict(xgb.model, test)

#########################################################################################################################################################

#20

rf.model <- train(price ~ .,data = data,tuneGrid = data.frame(.mtry = 20),trControl = control,method = 'rf')

price.rf1 <- predict(rf.model, test)

#25

rf.model <- train(price ~ .,data = data,tuneGrid = data.frame(.mtry = 25),trControl = control,method = 'rf')

price.rf2 <- predict(rf.model, test)

#45

rf.model <- train(price ~ .,data = data,tuneGrid = data.frame(.mtry = 45),trControl = control,method = 'rf')

price.rf3 <- predict(rf.model, test)

#23

rf.model <- train(price ~ .,data = data,tuneGrid = data.frame(.mtry = 23),trControl = control,method = 'rf')

price.rf4 <- predict(rf.model, test)

#16

rf.model <- train(price ~ .,data = data,tuneGrid = data.frame(.mtry = 16),trControl = control,method = 'rf')

price.rf5 <- predict(rf.model, test)

#12

rf.model <- train(price ~ .,data = data,tuneGrid = data.frame(.mtry = 12),trControl = control,method = 'rf')

price.rf6 <- predict(rf.model, test)

#########################################################################################################################################################

result <- data.frame(price.xgb1, price.xgb2, price.xgb3, price.xgb4, price.xgb5, price.xgb6, price.rf1, price.rf2, price.rf3, price.rf4, price.rf5, price.rf6)

entire.ensemble <- apply(result, 1, mean)

#########################################################################################################################################################

price <- exp(entire.ensemble)

########################################################################################

# 결과 저장 (수정하지 말것)

write.csv(data.frame(price=price), paste0(name, '.csv'))