이름 : 정희철

학번 : 2016314895

library(mice)

library(rms)

library(finalfit)

library(CORElearn)

library(pROC)

^ 사용한 패키지들

#먼저 데이터를 불러와 대략적으로 데이터에 대한 정보를 얻어본다.

training.data <- read.csv("Google Drive/My Drive/skku class files/2021년도 2학기/통계적 모델링과 머신러닝 실습/13주차/train-1.csv", header = TRUE)

test.data <- read.csv("Google Drive/My Drive/skku class files/2021년도 2학기/통계적 모델링과 머신러닝 실습/13주차/test-1.csv", header = TRUE)

describe(training.data)

# 추후 model training에 차질이 없게 하기 위해 알맞은 data type으로 설정해준다.

training.data[,1] <- as.numeric(training.data[,1])

training.data[,2] <- as.factor(training.data[,2])

training.data[,3] <- as.factor(training.data[,3])

training.data[,4] <- as.factor(training.data[,4])

training.data[,5] <- as.factor(training.data[,5])

training.data[,6] <- as.numeric(training.data[,6])

training.data[,7] <- as.factor(training.data[,7])

training.data[,8] <- as.factor(training.data[,8])

training.data[,9] <- as.factor(training.data[,9])

training.data[,10] <- as.factor(training.data[,10])

training.data[,11] <- as.factor(training.data[,11])

training.data[,12] <- as.numeric(training.data[,12])

training.data[,13] <- as.integer(training.data[,13])

training.data[,14] <- as.integer(training.data[,14])

training.data[,15] <- as.integer(training.data[,15])

training.data[,16] <- as.factor(training.data[,16])

training.data[,17] <- as.factor(training.data[,17])

test.data[,1] <- as.numeric(test.data[,1])

test.data[,2] <- as.factor(test.data[,2])

test.data[,3] <- as.factor(test.data[,3])

test.data[,4] <- as.factor(test.data[,4])

test.data[,5] <- as.factor(test.data[,5])

test.data[,6] <- as.numeric(test.data[,6])

test.data[,7] <- as.factor(test.data[,7])

test.data[,8] <- as.factor(test.data[,8])

test.data[,9] <- as.factor(test.data[,9])

test.data[,10] <- as.factor(test.data[,10])

test.data[,11] <- as.factor(test.data[,11])

test.data[,12] <- as.numeric(test.data[,12])

test.data[,13] <- as.integer(test.data[,13])

test.data[,14] <- as.integer(test.data[,14])

test.data[,15] <- as.integer(test.data[,15])

test.data[,16] <- as.factor(test.data[,16])

test.data[,17] <- as.factor(test.data[,17])

#데이터에 unknown으로 채워져있는 부분들이 매우 많다. 이것을 missing values로 취급하여 data modelling을 진행해보겠다. 먼저 모든 unknown들을 NA로 치환해주고, missing values에 패턴이 있는지 알아본다.

training.data[training.data=='unknown'] <- NA

md.pattern(training.data)

# 변수 X2, X4, X9, X16에 missing values가 있는 것으로 보인다. 이들을 집중적으로 조사해봐야 한다.

missing\_pairs(training.data,'Y',c('X2','X4','X9','X16') )

md.pairs(training.data)

# X16과 X9변수에 너무나 많은 비율의 observations들이 missing values로 되어 있기에 imputation을 진행할 경우 데이터를 왜곡하게 되기 때문에 버리기로 선택한다.

training.data <- training.data[,c(-9,-16)]

md.pairs(training.data)

missing\_pairs(training.data,'Y',c('X2','X4') )

# missing values에 뚜렷한 패턴이 없는 듯 하다. Imputation을 진행해도 무관한 것으로 판단된다. X2는 고객의 직업으로 unordered categorical variable이므로 polyreg를 사용할 것이다. X4는 고객의 교육수준으로 ordered categorical variable이므로 polr을 사용한다.

set.seed(0)

imp = mice(training.data, m=10, method=c('','polyreg','','polr','','','','','','','','','','',''), print=F)

pred = imp$predictorMatrix

pred[,'Y'] = 0

imp1 = mice(training.data, m=10, method=c('','polyreg','','polr','','','','','','','','','','',''), predictorMatrix = pred, print=F)

imp1$predictorMatrix

colnames(training.data)

stripplot(imp1, pch=20, cex=1.2)

# 딱히 패턴이 보이지 않는다.

training.set <- complete(imp)

# Filtering : Y변수는 categorical, X 변수는 continuous와 categorical이 섞여 있기 때문에 Relief와 ReliefF 알고리즘을 사용하기 적합하다.

# Relief algorithm

RE = attrEval(Y ~ ., data=training.set, estimator='Relief',ReliefIterations=30)

SRE = sort(RE, decreasing = T)

SRE

plot(1:length(SRE),SRE, type='b', ylab='Separability',

xlab ='variables', main='Variable Importance', xaxt='n')

axis(side=1, at=1:length(SRE), labels=names(SRE), cex.axis=0.8,las=2)

# X11, X10, X12 변수들이 유의하다고 판단된다.

# ReliefF algorithm

REF = attrEval(Y ~ ., data=training.set, estimator='ReliefFequalK', ReliefIterations=30)

SREF = sort(REF, decreasing = T)

SREF

plot(1:length(SREF),SREF, type='b', ylab='Purchase',

xlab ='variables', main='Variable Importance', xaxt='n')

axis(side=1, at=1:length(SREF), labels=names(SREF), cex.axis=0.8,las=2)

# X12, X11, X10 변수들이 유의하다고 판단된다.

# Building a model

fit = glm(Y ~ X10 + X11 + X12, data=training.set, family=binomial )

summary(fit)

new.obs <- test.data[,c('X10','X11','X12')]

predicted <- predict(fit, new.obs)

predicted[predicted>=0] <- 'yes'

predicted[predicted<0] <- 'no'

actual <- test.data[,'Y']

# F-measure

cm = table(true = actual, predict=predicted)

cm

misclass = function(cm) 1 - sum(diag(cm))/sum(cm)

fmeasure = function(cm)

{

TPR = cm[2,2]/sum(cm[2,])

PPV = cm[2,2]/sum(cm[,2])

return((2\*TPR\*PPV)/(TPR + PPV))

}

misclass(cm)

fmeasure(cm)

# Misclassification Rate은 0.204, F-measure은 0.6525886이다.

# 여기서 끝낼 수 있지만, cut-off value를 조정하여 model을 향상시킬 수 있는지 확인해본다.

phat.tr = predict(fit, training.set, type='response')

lr.roc = roc(training.set$Y ~ phat.tr)

plot(lr.roc)

auc(lr.roc)

th = coords(lr.roc,x='best', best.method = 'closest.topleft')

th # new threshold = 0.2928077

predicted <- predict(fit, new.obs)

predicted[predicted>=0.2928077] <- 'yes'

predicted[predicted<0.2928077] <- 'no'

actual <- test.data[,'Y']

# F-measure

cm = table(true = actual, predict=predicted)

cm

fmeasure = function(cm)

{

TPR = cm[2,2]/sum(cm[2,])

PPV = cm[2,2]/sum(cm[,2])

return((2\*TPR\*PPV)/(TPR + PPV))

}

misclass(cm)

fmeasure(cm)

# Misclassification Rate은 0.2072, F-measure은 0.6273381이다.

# Cut-off value를 조정한 model의 performance가 더 안좋게 나왔기 때문에 첫번째 model을 선택한다.

최종 F-measure : 0.6525886