


Data Mining Final Test

- Purchase Prediction using Caravan dataset

학번 : 182STG12

이름 : 오혜윤



Problem Description

Caravan dataset is about a direct marketing case from the insurance sector which was to predict policy ownership. It is about predicting who would be interested in buying a caravan insurance policy.

Each real customer record consists of 86 variables, containing sociodemographic data (variables 1-43) and product ownership data (variables 44-86). The sociodemographic data is derived from zip codes. All customers living in areas with the same zip code have the same sociodemographic attributes. Variable 86 (Purchase) is the target variable which indicates whether the customer purchase a caravan insurance policy or not.

The following pie chart shows the number of customers who purchased(Yes) the Caravan policy which is 348 and who have not purchased(NO) the Caravan policy which is 5474. You can see that the Caravan dataset is the biased dataset with YES/NO classes and the ratio of the two classes is about 1:15

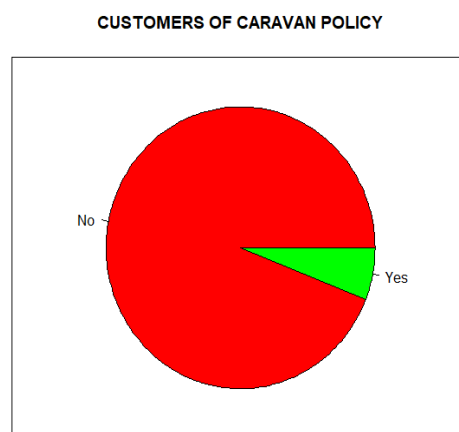


Figure 1: PIE CHART OF YES/NO FOR PURCHASE OF CARAVAN POLICY BY CUSTOMERS

Results

This question uses the Caravan data set.

- (a) Create a training set consisting of the first 1,000 observations, and a test set consisting of the remaining observations.

R codes:

```
>set.seed(1234)
>train <- 1:1000
>Caravan$Purchase <- ifelse(Caravan$Purchase == "Yes", 1, 0)
>Caravan.train <- Caravan[train, ]; Caravan.test <- Caravan[-train, ]
```

- (b) Fit a boosting model (GBM and XGBoost) to the training set with Purchase as the response and the other variables as predictors. Use 1,000 trees, and a shrinkage value of 0.01. Which predictors appear to be the most important? Answer the questions for each model.

① GBM model

The following chart indicates the relative influence values of the top 5 variables.

Coefficients:	PPERSAUT	MKOOKPLA	MOPLHOOG	MBERMIDD	PBRAND
	15.3073	8.9568	7.5445	5.9768	5.5327

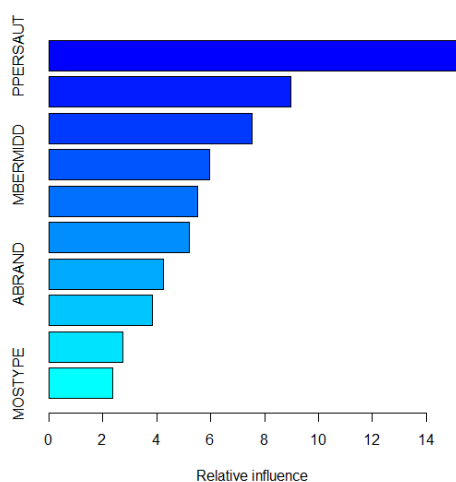


Figure 2: Relative Influence graph of the top 10 variables (gbm)

According to the GBM model, the top 5 variables are PPERSAUT, MKOOPKLA, MOPLHOOG, MBERMIDD, and PBRAND.

② XGboost model

With the same parameters (nrounds = 1000, eta = 0.01), we do the same thing with the XGboost model. The following chart indicates the relative influence values of the top 5 variables.

Coefficients:	MGODGE	MBERMIDD	PPERSAUT	MINK3045	MOSTYPE
	0.0624	0.0608	0.0603	0.0435	0.0423

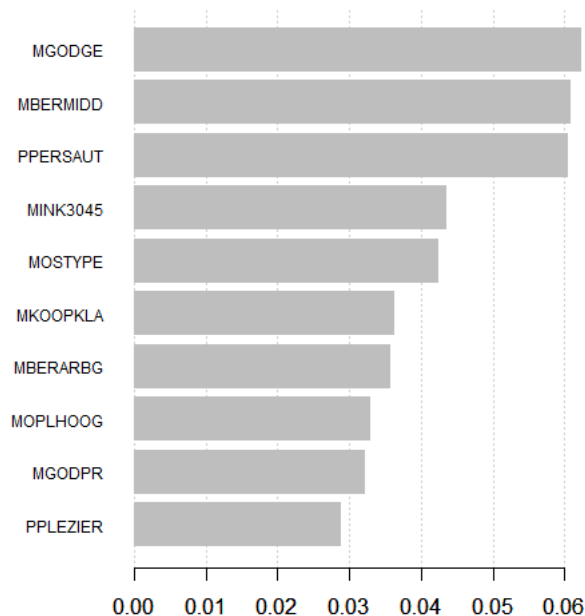


Figure 3: Importance plot of the top 10 variables (XGboost)

According to the XGboost model, the top 5 variables are MGODGE, MBERMIDD, PPERSAUT, MINK3045, and MOSTYPE.

As a result, there are two variables, MBERMIDD and PPERSAUT, which are selected features from the both models.

- (c) Use the boosting model to predict the response on the test data. Predict that a person will make a purchase if the estimated probability of purchase is greater than 20%. Form a confusion matrix. What fraction of the people predicted to make a purchase do in fact make one? How does this compare with the results obtained from applying KNN or logistic regression or random forest model to this data set?

According to the GBM model, of the 258 people predicted to make a purchase, only 31 of them did so. Therefore, about 24% of the people predicted to make a purchase do in fact make one.

Table 1: Confusion Matrix of gbm model

	Predicted		
true	No	Yes	-err.-
No	4406	127	127
Yes	258	31	258
-err.-	258	127	385

On the other hand, according to the XGboost model, among the 169 people predicted to make a purchase, only 29 of them did so. Therefore, about 17% of the people predicted to make a purchase do in fact make one.

Table 2: Confusion Matrix of xgboost model

	Predicted		
true	No	Yes	-err.-
No	4364	169	169
Yes	260	29	260
-err.-	260	169	429

Also, according to the KNN model, of the 273 people predicted to make a purchase, only 23 of them did so. Therefore, about 8% of the people predicted to make a purchase do in fact make one.

Table 3: Confusion Matrix of KNN model

	Predicted		
true	No	Yes	-err.-

No	4260	273	273
Yes	266	23	266
-err.-	266	273	539

Finally, according to the Logistic Regression model, of the 350 people predicted to make a purchase, only 58 of them did so. Therefore, about 17% of the people predicted to make a purchase do in fact make one.

Table 4: Confusion Matrix of logistic regression model

	Predicted		
true	No	Yes	-err.-
No	4183	350	350
Yes	231	58	231
-err.-	231	350	581

As a result, the GBM model is the best model to predict that people will make a purchase with the accuracy 24%. The second-best models are the XGboost model and the Logistic Regression model with the accuracy 17%.

Discussion

We applied various boost-based models, KNN and logistic regression model to the Caravan dataset. In the dataset we used, the boosting model was the best. We cannot always conclude that the bagging or boosting model is always the best. However, with the Caravan model, the boost models including gbm and xgboost were the best models.

All the models had the lower accuracies and I think this is because we did not do sampling when we split the dataset into the training set and the test set. We just used the first 1000 observations for the training set. Also, the Caravan dataset has too many zeros (y) and so we need to oversample the dataset. Therefore, if we do oversample, its model accuracy will be higher than these results. Also, I think Naïve Bayes model will be the adequate model to this dataset because we can apply the probit model using Naïve Bayes.

Appendix

```

setwd("C:/Users/user/Desktop/대학원 3 학기/data
mining/종합시험/")

library(tidyverse) ; library(dplyr) ; library(ggplot2) ;
library(DataExplorer)

##data-----

Caravan = read.csv("Caravan.csv"); head(Caravan)

ncol(Caravan); nrow(Caravan)

##eda-----

a<-table(Caravan$Purchase)

colors=c("red","green")

pie(a,main = "CUSTOMERS OF CARAVAN
POLICY",col=colors);box()

#plot_boxplot(Caravan,by = 'Purchase')

#plot_density(Caravan)

##a)-----

set.seed(1234)

train <- 1:1000

Caravan.train <- Caravan[train, ];Caravan.test <- Caravan[-
train, ]

##b)-----

library(gbm) ; library(mlr);library(xgboost)

set.seed(1234)

traintask.caravan <- makeClassifTask(data = Caravan.train,
target = "Purchase", fixup.data = "quiet")

##gbm

gbm_learner <- makeLearner("classif.gbm", par.vals =
list(distribution = "bernoulli", n.trees = 1000, shrinkage = 0.01),
predict.type = "prob")

```

```

mod.gbm <- train(gbm_learner, traintask.caravan)

# max.features=relative.influence(getLearnerModel(mod.gbm),
n.trees = 1000, scale = TRUE) %>% sort(decreasing =
TRUE) %>% head(10)

summary( getLearnerModel(mod.gbm), cBars = 10,
method = relative.influence)

##xgboost

xgb_learner <- makeLearner("classif.xgboost", predict.type =
"prob", par.vals = list(objective = "binary:logistic",nrounds =
1000, eta = 0.01))

mod.xgb <- train(xgb_learner, traintask.caravan)

xgb_model<-getLearnerModel(mod.xgb)

importance_matrix <- xgb.importance(model = xgb_model)

xgb.plot.importance(importance_matrix, top_n =
10);importance_matrix

##c)-----

threshold <- c(Yes = 0.2, No = 0.8) #Setting up 20% Probability
Threshold

##gbm

pred <- predict(mod.gbm, newdata = data.frame(Caravan.test))

pred <- setThreshold(pred, threshold = threshold)

calculateConfusionMatrix(pred)

#31/127 *100

##xgboost

pred2 <- predict(mod.xgb, newdata = data.frame(Caravan.test))

pred2 <- setThreshold(pred2, threshold = threshold)

calculateConfusionMatrix(pred2)

#29/169*100

```


Final Test

```
##knn

traintask.knn.caravan <-
createDummyFeatures(traintask.caravan)

testtask.knn.caravan <-
createDummyFeatures(makeClassifTask(data = Caravan.test,
target = "Purchase", fixup.data = "quiet"))

knn_learner <- makeLearner("classif.knn", par.vals = list(prob =
TRUE))

set.seed(1234)

mod.knn <- train(knn_learner, traintask.knn.caravan)

pred3 <- predict(mod.knn, testtask.knn.caravan)

calculateConfusionMatrix(pred3)

#23/273*100

##logistic

logit.caravan <- glm(Purchase ~ ., data = Caravan.train, family =
"binomial")

pred4 <- predict(logit.caravan, Caravan.test, type = "response")

pred4 <- ifelse(pred4 > 0.2, 1, 0)

table(Caravan.test$Purchase, pred4)

#58/350*100
```