Report for 479 Project II

Jiyun Chen

[Result]

I randomly select a test data(sample size =18000) to verify the model.

The result of my prediction, the following table shows the cost value of the train data according to the different model methods and different training subsets and the cost value of the test data.(train data cost/test data cost, the single value refers to the train data cost)

model method \ train dataset method	select 100000 records randomly in one year(2016)	select the important variable through the 100000 records dataset	select 400000 records randomly in six years	select 2400000 records randomly in six years	select whole 4000000 records in one year(2016)
guide + single tree	0.2863	0.2863	/	0.2861	0.2837/0.2883
guide + ensemble tree	0.2874/0.2934	0.2894/0.2978	0.2847/0.2923	0.2835/0.2821	/
xgboost	0.2673	/	0.2554/0.3286	0.2622/0.3119	/
randomforest	/	0.4535	/	/	/
svm	/	0.5054	/	/	/

From the above table, I choose the minimal cost value of test data. Thus, I choose the method "guide+ensemble tree according to the 2400000 records subset". (**My first column of the result**)

And I notice that the results of the methods in guide are very close, and the other three methods are not as good as guide. Thus, I combine the five methods(all of them are got from guide) together and count the prediction and modify the "first column" a bit to get **my second column of the result**.

[Procedure]

Step1: Pre-processing the dataset

1. Delete some redundant variables according to the file natl2016.pdf.

   According to the PDF, we can know that there are some flag variable, meaning that some two variables refer to the same meaning. Thus, I delete these variables , which are called,

   "bmi_r","fagerec11","gestrec3","illb_r","ilop_r","mager9","mager14","wtgain_rec"

2. Set the data type for each variable.

   According to the description in PDF for the every variable, I assign "numerical" data type to the following variables (including the ordinal categorical variables),

"bmi","cig_0","cig_1","cig_2","cig_3","combgest","dwgt_r","fagecomb","feduc","gestrec10","lbo_rec",

"m_ht_in","meduc","precare","previs_rec","priorterm","pwgt_r","rf_cesarn","setorder_r","tbo_rec",

"wtgain"

3. Assign the "NA" to the "Unknown Group" in the numerical variables.

   According to the train data, we can see in the numerical variables there are something like "999=Unknown", they will influence the result. So, we transfer these "999" to NA.

4. Change the variable name in different years and randomly get the data subset.

   The variables' names in 2014-2016 are different from the names in 2011-2013. I transfer them to the same, and then get the different subset to train.

Step2: Build the model through different method

For guide method, I get the description file according to the data type of each variables. Then, get the input file and use guide to build the different model.

For xgboost method, I read the data and assign the data type to each column. Then I transfer the data frame into the matrix, using "xgboost" package to build the different according to the different subsets. Next I notice the xgboost method("binary:logistic") only can predict the probability for the class, so here I try to find the boundary of the probability to get the minimal cost train data. Last, I change the value of each parameter to improve my model.

For randomforest method, almost the same as the "xgboost" method (the difference is that it can return the class for the test data)

For svm method, almost the same as above.

Step3: Verify the model

I randomly select a test data(sample size =18000) to verify the model and calculate the cost value of the test data. And I choose the method which the cost value of test data is the smallest.

Step4: Combine the guide method and count

I notice that the results of the methods in guide are very close, and the other three methods are not as good as guide. Thus, I combine the five methods(all of them are got from guide) together and count the prediction, meaning that I denote the class "1" as 1, and class "0" as 0, and sum the five columns. If the sum value is 0 or 1, I regard this one as class "0"; if the sum value is 4 or 5, I regard this one as class "1"; if the sum is 2 or 3, I retain the answer in the method "guide+ensemble tree according to the 2400000 records subset", the final answer will be my second answer.

[Appendix]

Rode for method "xgboost" in R

```
rm(list=ls())
library(data.table)
library(xgboost)
require(ROCR)
```

```
#step1: get the data type of the all variables
dsc <- readLines("dscfile.txt")
roles <- NULL
for(j in 4:length(dsc)){ # store original roles of variables
  strng <- strsplit(dsc[j]," +")[[1]]
  roles <- c(roles,strng[3])
}
vartype <- c()
for(i in 1:length(roles)){
  if(roles[i]=="c"){
    vartype[i]="factor"
  }
  else{
    vartype[i]="numeric"
  }
}

#step2: fix up the missing value with the median
z=read.table("2016_allvar_allyear.rdata",header = TRUE,colClasses = vartype)
z$wt<-NULL
for (i in 1:ncol(z)) {
  if (class(z[, i]) == "numeric") {
    index = which(is.na(z[, i]))

    if (length(index) == 0) {
      next
    } else{
      z[index, i] <- median(z[, i][-index])
    }
  }
}
dim(na.omit(z));z_all<-z
set.seed(0)
index=sample(1:(nrow(z)-18000),500000)
z<-z[index,]
dim(na.omit(z))

#step3: get the train data,sample sizes=50000
setDT(z)
traindata=z[1:400000,]
testdata=z[400001:500000,]
testdata=z[2236989:2254988,]
label_train=traindata$lowbwt
labels <- sign(as.numeric(label_train) ==2 )
xg_train <- model.matrix(~.+0,data = traindata[,-c("lowbwt"),with=F])
xg_test <- model.matrix(~.+0,data = testdata[,-c("lowbwt"),with=F])
model_xg <- xgboost(data = xg_train, label = labels, max.depth = 3, eta = 0.3,nround = 300,
objective = "binary:logistic")
predict_xg_train=predict(model_xg,xg_train)

#step4: calculate auc
label_test=sign(as.numeric(testdata$lowbwt) ==2 )

pre_train=prediction(predict_xg_train,labels = labels)
```

```
auc_train <- performance(pre_train, "auc")@y.values
auc_train

#step5: calculate the cost
#table(label_test);table(sign(predict_xg>=0.088))
cost_sum<-c();
rank=predict_xg_train[order(predict_xg_train,decreasing = T)]
for (i in 365557:dim(traindata)[1]){
  res <- sign(predict_xg_train>=rank[i])-labels
  ind_1 <- which(res == 1)
  ind_2 <- which(res ==-1)
  cost <- sum(rep(1,length(ind_1)))+sum(rep(10,length(ind_2)))
  cost_sum[i]<-cost/dim(traindata)[1]
  print(c(i,cost_sum[i]))
}
bound=predict_xg_train[which(cost_sum==min(cost_sum))]#0.08652855_40wan

#step6 predict
predict_xg_test=predict(model_xg,xg_test)
res <- sign(predict_xg_test>=bound)-label_test
ind_1 <- which(res == 1)
ind_2 <- which(res ==-1)
cost <- sum(rep(1,length(ind_1)))+sum(rep(10,length(ind_2)))
cost/dim(testdata)[1]
pre=prediction(sign(predict_xg_test>=bound),labels = label_test)
per<-performance(pre,"tpr","fpr" )
auc <- performance(pre, "auc")@y.values
auc
```

Rode for method "randomforest" in R

```
rm(list=ls())
library(randomForest)
library(ROCR)
a <- "factor"; n <- "numeric"
vartype <-
c(a,a,a,n,n,n,n,a,n,a,a,a,a,a,n,a,n,a,n,a,n,a,a,a,a,a,a,n,a,a,a,a,a,a,n,a,a,a,a,a,n,n,a,a,a,a,a,n,a,a,a,a,a,
n,n,n,n,a,a,a,a,a,a,n,a,n,a,n,a)
z=read.table(file.choose(),header = TRUE,colClasses = vartype)
z$wt<-NULL
for (i in 1:ncol(z)) {
  if (class(z[, i]) == "numeric") {
    index = which(is.na(z[, i]))
    if (length(index) == 0) {
      next
    } else{
      z[index, i] <- median(z[, i][-index])
    }
  }
}
traindata=z[1:20000,]

testdata<-z[-(1:100000),]
```

```
set.seed(1234)
n <- length(names(traindata))
for (i in 1:(n-1)){
  model <- randomForest(lowbwt~., data = traindata, mtry = i, ntree=100)
  err <- mean(model$err.rate)
  print(err)
}
rf_ntree <- randomForest(lowbwt~.,data=traindata,ntree=300,mtry=13)#
rf_ntree <- grow(rf_ntree,100)
predict=predict(rf_ntree,data=traindata)
rf_ntree
plot(rf_ntree)
names(predict)<-NULL
pre=prediction(as.integer(predict),labels = as.integer(traindata$lowbwt))
per<-performance(pre,"tpr","fpr" )
auc <- performance(pre, "auc")@y.values
auc
sum=0;index<-c();j=1
for(i in 1:100000){
  if((as.integer(traindata$lowbwt)[i]-as.integer(predict)[i])==0){
    sum=0+sum
  }
  else{
    if(as.integer(predict)[i]==1){
      sum=sum+10
      index[j]=i
      j=j+1
    }
    else{
      sum=sum+1
    }
  }
}
sum/100000
```

Rode for method "svm" in R

```
rm(list=ls())
library(randomForest)
library(ROCR)
a <- "factor"; n <- "numeric"
vartype <-
c(a,a,a,n,n,n,n,a,n,a,a,a,a,a,n,a,n,a,n,a,n,a,a,a,a,a,a,n,a,a,a,a,a,n,a,a,a,a,a,n,n,a,a,a,a,a,n,a,a,a,a,a,
n,n,n,n,a,a,a,a,a,a,n,a,n,a,n,a)
z=read.table(file.choose(),header = TRUE,colClasses = vartype)
z$wt<-NULL
for (i in 1:ncol(z)) {
  if (class(z[, i]) == "numeric") {
    index = which(is.na(z[, i]))
    if (length(index) == 0) {

      next
```

```
    } else{
      z[index, i] <- median(z[, i][-index])
    }
  }
}
traindata=z[1:20000,]
testdata<-z[-(1:100000),]
model.svm<-svm(train$lowbwt~.,data = train,type='C-
classification',cross=5,cost=c(1,10),kernal="radial")

pred <- predict(model.svm, train[,1:112])

svmroc = prediction(sign(as.integer(model.svm$fitted) == 2),labels = as.integer(train$lowbwt))
svmauc = performance(svmroc,"auc")@y.values
```