Jake Kruse

Lab 3

Geog 560

(1) (6 pts) Which are the **mean accuracy values for Logistic Regression** and Naïve Bayes’ Classification **using the 10-fold cross validation approach**? You can try different combination of predictors and describe your findings.

Using AIC values to determine the best Logistic Regression model, it was determined that all three variables produce the best model, with a mean accuracy of 0.9733 for a 10-fold CV. Overall accuracy (from the confusion matrix) was 97.4% when using all three variables in NB. The True/False accuracies were generated with 10-fold CV for NB (bottom portion of table).



(2) (4 pts) Generate the confusion matrix (table) for any of the 10-fold in previous step of Logistic Regression or Naïve Bayes’ Classification. Comment about your findings in the table.

This Confusion Matrix was generated for the NB Classifier using all variables. Overall accuracy was 97%, with a 95% CI of 97%-98%. Balanced accuracy was only 60%, showing the how the “No” category influenced the overall accuracy much more than the “Yes” category, which is problematic given the assumption that banks care more about detecting defaults than non-defaults. Specificity is very low, meaning that some true negatives cases are missed.

Confusion Matrix and Statistics

Reference

Prediction No Yes

No 1935 46

Yes 7 12

Accuracy : 0.9735

95% CI : (0.9655, 0.9801)

No Information Rate : 0.971

P-Value [Acc > NIR] : 0.2789

Kappa : 0.3017

Mcnemar's Test P-Value : 1.792e-07

Sensitivity : 0.9964

Specificity : 0.2069

Pos Pred Value : 0.9768

Neg Pred Value : 0.6316

Prevalence : 0.9710

Detection Rate : 0.9675

Detection Prevalence : 0.9905

Balanced Accuracy : 0.6016

'Positive' Class : No

Code

## Load data from external packages

if(!require("ISLR")) install.packages("ISLR")

library("ISLR") #load an installed package before usage

if(!require("e1071")) install.packages("e1071")

library(e1071)

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

library("caret")

if(!require("klaR")) install.packages("klaR")

library("klaR")

data(Default) #load data from the installed package

summary(Default) # get summary statistics about this dataset

#Graphs to visualize the data

par(mar=c(2,2,2,2)) #set the margin of figure

par(mfrow=c(1, 3)) # divide the graph area in 1 row, 2 columns

plot(Default$balance, Default$default, col="gold", main='Default ~ Balance') #scatter plot of balance and default

plot(Default$income, Default$default, col="sky blue", main='Default ~ Income') #scatter plot of income and default

boxplot(Default$student, Default$balance, col="sky blue", main='Balance ~ Student')

##############1

head(Default$student)

##Accuracy of LR using 10-fold CV

# Run the logistic regression model for categorical label prediction (logisitc regression is used for binary data)

help(glm) #search for this function Generalized Linear Models

#balance

logit1b <- glm(default ~ balance, family=binomial(link='logit'),data=Default)

summary(logit1b) #model result details

fitVal1b <- logit1b$fitted.values #predicted the probablity values from the logit model

#income

logit1i <- glm(default ~ income, family=binomial(link='logit'),data=Default)

summary(logit1i) #model result details

fitVal1i <- logit1i$fitted.values #predicted the probablity values from the logit model

#student status

logit1s <- glm(default ~ student, family=binomial(link='logit'),data=Default)

summary(logit1s) #model result details

fitVal1s <- logit1s$fitted.values #predicted the probablity values from the logit model

#balance and income

logit2bi <- glm(default ~ balance+income, family=binomial(link='logit'),data=Default)

summary(logit2bi) #model result details

fitVal2bi <- logit2bi$fitted.values #predicted the probablity values from the logit model

#student and income

logit2si <- glm(default ~ student+income, family=binomial(link='logit'),data=Default)

summary(logit2si) #model result details

fitVal2si <- logit2si$fitted.values #predicted the probablity values from the logit model

#student and income and balance

logit3sib <- glm(default ~ student+income+balance, family=binomial(link='logit'),data=Default)

summary(logit3sib) #model result details

fitVal3sib <- logit3sib$fitted.values #predicted the probablity values from the logit model

###mean accuracy values LR

# Define training control

train.control <- trainControl(method = "cv", number = 10)

# Train the model

model <- train(default ~ student+income+balance, data = Default, method = "glm",

trControl = train.control)

# Summarize the results

print(model)

###Naive Bayes accuracy

naiveBayesmodel <- naiveBayes(default ~ ., data = Default) # consider all predictors

print(naiveBayesmodel)

# Define training control

train.controlnb <- trainControl(method = "cv", number = 10)

# Train the model

modelnb <- train(default ~ ., data = Default, method = "nb",

trControl = train.control)

# Summarize the results

print(modelnb)

###Naive Bayes accuracy

naiveBayesmodel2 <- naiveBayes(default ~ income+balance, data = Default) # consider 2 predictors

print(naiveBayesmodel2)

# Define training control

train.controlnb <- trainControl(method = "cv", number = 10)

# Train the model

modelnb <- train(default ~ income+balance, data = Default, method = "nb",

trControl = train.control)

# Summarize the results

print(modelnb)

###Naive Bayes accuracy

naiveBayesmodel2si <- naiveBayes(default ~ income+student, data = Default) # consider 2 predictors

print(naiveBayesmodel2si)

# Define training control

train.control2si <- trainControl(method = "cv", number = 10)

# Train the model

model2si <- train(default ~ income+student, data = Default, method = "nb",

trControl = train.control)

# Summarize the results

print(model2si)

###Naive Bayes accuracy

naiveBayesmodelb<- naiveBayes(default ~ balance, data = Default) # consider 1 predictor

print(naiveBayesmodelb)

# Define training control

train.controlb <- trainControl(method = "cv", number = 10)

# Train the model

modelb <- train(default ~ balance, data = Default, method = "nb",

trControl = train.control)

# Summarize the results

print(modelb)

###Naive Bayes accuracy

naiveBayesmodeli<- naiveBayes(default ~ income, data = Default) # consider 1 predictor

print(naiveBayesmodeli)

# Define training control

train.controli <- trainControl(method = "cv", number = 10)

# Train the model

modeli <- train(default ~ income, data = Default, method = "nb",

trControl = train.control)

# Summarize the results

print(modeli)

###Naive Bayes accuracy

naiveBayesmodels<- naiveBayes(default ~ student, data = Default) # consider 1 predictor

print(naiveBayesmodels)

# Define training control

train.controls <- trainControl(method = "cv", number = 10)

# Train the model

models <- train(default ~ student, data = Default, method = "nb",

trControl = train.control)

# Summarize the results

print(models)

##########2

#Seperate Training and Testing datasets

sample\_index <- sample(10000,8000) # randomly generate the row index for 80% trainning dataset (the size ccould vary)

training\_data<- Default[sample\_index,] # create the trainning dataset that contains 80% of the whole data

testing\_data<- Default[-sample\_index,] # get the testing dataset that contains 20% of the whole data

naiveBayesmodeltrd <- naiveBayes(default ~ balance, data = training\_data) # only consider one predictor

print(naiveBayesmodeltrd)

pred <- predict(naiveBayesmodelb,testing\_data) #predict the class label of default behavior with different balance values in the testing data

accuracy\_table <- table(pred, testing\_data[,"default"]) #create the prediction result table

accuracy <- sum(diag(accuracy\_table))/sum(accuracy\_table)

print(accuracy)

confusionMatrix(data=pred, testing\_data$default) # get the confusion matrix