ASSINGMENT_2

Universal Bank (MIS 64060)

#Importing data set (Universal Bank CSV FILE)

library(readr)

UniversalBank <- read_csv("UniversalBank.csv")

spec(UniversalBank)

Assigning names to column

colnames(UniversalBank) <- c('ID', 'Age', 'Experience', 'Income', 'ZIP_Code', 'Family', 'CCAvg', 'Education', 'Mortgage', 'Personal_Loan', 'Securities_Account', 'CD_Account', 'Online', 'Credit_Card')

summary(UniversalBank)

Getting Rid of Zip Code and ID

UniversalBank$ID <-NULL

UniversalBank$ZIP_Code<-NULL

summary(UniversalBank)

Factoring Education and personal loan

UniversalBank$Education = as.factor(UniversalBank$ Education)

UniversalBank$Personal_Loan = as.factor(UniversalBank$ Personal_Loan)

summary(UniversalBank)

library(caret)

library(class)

dummies <- dummyVars(Personal_Loan ~ ., data = UniversalBank)

UniversalBank_dummy=as.data.frame(predict(dummies, newdata=UniversalBank))

print(UniversalBank_dummy)

head(UniversalBank_dummy)

Normalizing Data

Norm_model <- preProcess(UniversalBank_dummy,method= c("center","scale"))

UniversalBank_norm = predict(Norm_model, UniversalBank_dummy)

summary(UniversalBank_norm)

print(UniversalBank_norm)

Adding back the target attribute

UniversalBank_norm$Personal_Loan = UniversalBank$ Personal_Loan

Dividing the data into train and validation.(60/40)

Train1_Index = createDataPartition(UniversalBank$Personal_Loan,p=0.6, list=FALSE) # 60% reserved for Train Train1.df=UniversalBank_norm[Train1_Index,] Validation.df=UniversalBank_norm[-Train1_Index,]

Task 1

#(a k-NN classification with all predictors except ID and ZIP code using k = 1. How would this customer be classified)

#(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0, Education_2 = 1, Education_3 = 0, Mortgage = 0, Securities Account = 0, CD Account = 0, Online = 1, and Credit Card = 1.)

To_Predict = data.frame(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education.1 = 0, Education.2 = 1, Education.3 = 0, Mortgage = 0, Securities_Account = 0, CD_Account = 0, Online = 1, Credit_Card = 1)

print(To_Predict)

Applying Normalization

To_Predict_norm= predict(Norm_model,To_Predict)

print(To_Predict_norm)

print(Norm_model)

Using knn for Prediction

Prediction <-knn(train=Train1.df[,1:13], test=To_Predict_norm[,1:13], cl=Train1.df$Personal_Loan, k=1)

print(Prediction)

TASK 2

Right choice of k to reduce the effect of overfitting and underfitting

##k=Number of cross fold Validation

#setting random number variables for reproducible results

set.seed(123)

fitControl <- trainControl(method = "repeatedcv", number = 3, repeats = 2)

searchGrid=expand.grid(k = 1:10)

Knn.model=train(Personal_Loan~., data=Train1.df, method='knn', tuneGrid=searchGrid, trControl = fitControl)

Knn.model

##RMSE was used to select the optimal model using the smallest value. The final value used for the model was k = 3.

TASK 3

##Confusion matrix for the validation data that results from using the best k

Predictions<- predict(Knn.model,Validation.df)

confusionMatrix(Predictions, Validation.df$Personal_Loan)

TASK 4

classifying customers using best k

#Considerations = (Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg= 2, Education_1 = 0, Education_2 = 1, Education_3 = 0, Mortgage = 0, Securities Account = 0, CD Account = 0, Online = 1 and Credit Card = 1.)

Prediction2 <-knn(train=Train1.df[,1:13], test=To_Predict_norm[,1:13], cl=Train1.df$Personal_Loan, k=3)

print(Prediction2)

TASK 5

Repartition of the data into train, test and validation.(50/30/20)

Train2_Index = createDataPartition(UniversalBank$Personal_Loan,p=0.5, list=FALSE) # 50% reserved for Train Train2.df=UniversalBank_norm[Train2_Index,] validation1.df=UniversalBank_norm[-Train2_Index,]

validation1_Index = createDataPartition(validation1.df$Personal_Loan,p=0.6, list=FALSE) # 60% reserved for validation validation2.df=validation1.df[validation1_Index,] Test1.df=validation1.df[-validation1_Index,]

#(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0, Education_2 = 1, Education_3 = 0, Mortgage = 0, Securities Account = 0, CD Account = 0, Online = 1, and Credit Card = 1.)

To_Predict1 = data.frame(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education.1 = 0, Education.2 = 1, Education.3 = 0, Mortgage = 0, Securities_Account = 0, CD_Account = 0, Online = 1, Credit_Card = 1)

print(To_Predict1)

Applying Normalization

Norm_model2 <- preProcess(Train2.df[,-13], method = c("center", "scale"))

Train2_Norm <- predict(Norm_model2, Train2.df[,-13])

Validation2_Norm <- predict(Norm_model2, validation2.df [,-13])

Test1_Norm <- predict(Norm_model2, Test1.df[,-13])

Prediction3 <- knn(Train2_Norm, Validation2_Norm , cl=Train2.df$ `Personal_Loan` , k=3,)

Prediction3

confusionMatrix(Prediction3, validation2.df$ `Personal_Loan`)

Comparision-

(Here we can see difference in test set with validation and training set. Major statistical difference are ;Accuracy level increased from 0.964 to 0.976 KAPPA increased from 0.7615 to 0.8454 TEST SET (False positive = 1 False Negative = 35) VALIDATION SET (False positive = 8 False Negative = 64)

(Overall we can see better result in set with test, validation and train set as using same k also, there is high level of accuracy. Using train test and validation model.extract increase the chance of efficiency by reducing the chance of over-fitting as well as adjusting the bias level. As the validation models hyper-parameters are specifically tuned into validation dataset improving results of evaluation.)