FML- KNN

Aamir Sohaib

2026-02-18

###Answer:1

```
#after importing dataset for my project, I will use summary function to know the d
ata and look at the important things and figures.
UniversalBank_1 <- read.csv("/Users/roc/Downloads/UniversalBank-1.csv")
summary(UniversalBank_1)
```

```
##        ID            Age          Experience       Income          ZIP.Code    
##  Min.   :   1   Min.   :23.00   Min.   :-3.0   Min.   :  8.00   Min.   : 9307  
##  1st Qu.:1251   1st Qu.:35.00   1st Qu.:10.0   1st Qu.: 39.00   1st Qu.:91911  
##  Median :2500   Median :45.00   Median :20.0   Median : 64.00   Median :93437  
##  Mean   :2500   Mean   :45.34   Mean   :20.1   Mean   : 73.77   Mean   :93153  
##  3rd Qu.:3750   3rd Qu.:55.00   3rd Qu.:30.0   3rd Qu.: 98.00   3rd Qu.:94608  
##  Max.   :5000   Max.   :67.00   Max.   :43.0   Max.   :224.00   Max.   :96651  
##      Family          CCAvg          Education        Mortgage    
##  Min.   :1.000   Min.   : 0.000   Min.   :1.000   Min.   :  0.0  
##  1st Qu.:1.000   1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0  
##  Median :2.000   Median : 1.500   Median :2.000   Median :  0.0  
##  Mean   :2.396   Mean   : 1.938   Mean   :1.881   Mean   : 56.5  
##  3rd Qu.:3.000   3rd Qu.: 2.500   3rd Qu.:3.000   3rd Qu.:101.0  
##  Max.   :4.000   Max.   :10.000   Max.   :3.000   Max.   :635.0  
##  Personal.Loan   Securities.Account   CD.Account         Online      
##  Min.   :0.000   Min.   :0.0000     Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.000   1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :0.000   Median :0.0000     Median :0.0000   Median :1.0000  
##  Mean   :0.096   Mean   :0.1044     Mean   :0.0604   Mean   :0.5968  
##  3rd Qu.:0.000   3rd Qu.:0.0000     3rd Qu.:0.0000   3rd Qu.:1.0000  
##  Max.   :1.000   Max.   :1.0000     Max.   :1.0000   Max.   :1.0000  
##    CreditCard   
##  Min.   :0.000  
##  1st Qu.:0.000  
##  Median :0.000  
##  Mean   :0.294  
##  3rd Qu.:1.000  
##  Max.   :1.000  
```

```
#Zip code and ID is not the predictor, so I'll remove these two items first from d
ata.
clean<- UniversalBank_1[, c(-1,-5)]
colnames(clean)<- make.names(colnames(clean))
```

```
#here I will create dummy variables for categorical values that are being used to
predict.
clean$Education<- as.factor(clean$Education)
dummy_func<- dummyVars(Personal.Loan ~., data = clean)
data_predictors<- as.data.frame(predict(dummy_func,clean))
data_transformed<- cbind(data_predictors, Personal.Loan = clean$Personal.Loan)
```

```
#I'll partition the data as required in question i.e. 60% training and 40% validat
ion.
set.seed(1)
train_index<- createDataPartition(data_transformed$Personal.Loan, p=0.6, list = FA
LSE)
train<- data_transformed[train_index,]
valid<- data_transformed[-train_index, ]
```

```
#I'll now add new customer data
new_customer<- 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, CreditCard = 1)
```

```
#now I'll use normalization process.
new_customer$Personal.Loan <- 0
norm_model<- preProcess(train,method = "range")

#apply normalization
train_norm<- predict(norm_model, train)
valid_norm<- predict(norm_model, valid)
new_customer_norm<- predict(norm_model, new_customer)
new_customer_norm$Personal.Loan <- NULL
```

```
#predictors
train_x <- train_norm[,!(names(train_norm)%in% "Personal.Loan")]
valid_x<-valid_norm[,!(names(valid_norm) %in%"Personal.Loan")]
new_x <- new_customer_norm[, !(names(new_customer_norm) %in%"Personal.Loan")]

#labels
train_y <-train_norm$Personal.Loan
valid_y <-valid_norm$Personal.Loan
```

```
#Now finally i'll use KNN
prediction<- knn(train = train_x,
                  test = new_x,
                  cl = train_y,
                  k = 1)

prediction
```

```
## [1] 0
## Levels: 0 1
```

###Answer:2

```
#It's common to check odd numbers to avoid ties.
search_grid <- expand.grid(k = seq(1, 25, 1))

#now i'll run the model using caret. We use the KNN method and specify the range n
ormalization within the function
#I had to onvert the target variable to factor in both sets, without it my model w
as giving me regression results including MSE etc instead of accuracy and Kappa.

train$Personal.Loan<- as.factor(train$Personal.Loan)
valid$Personal.Loan<- as.factor(valid$Personal.Loan)

set.seed(1)
model_tuned<- train(Personal.Loan ~ .,data =train, method = "knn", tuneGrid =searc
h_grid,
                     preProcess ="range")

#now i'll print the results to see what is the optimal value.
print(model_tuned)
```

```
## k-Nearest Neighbors
##
## 3000 samples
##   13 predictor
##    2 classes: '0', '1'
##
## Pre-processing: re-scaling to [0, 1] (13)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 3000, 3000, 3000, 3000, 3000, 3000, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.9567462  0.7062791
##    2  0.9483326  0.6413098
##    3  0.9458278  0.6087128
##    4  0.9454624  0.5995467
##    5  0.9454712  0.5886403
##    6  0.9444650  0.5746052
##    7  0.9435973  0.5587052
##    8  0.9431547  0.5491212
##    9  0.9416348  0.5295054
##   10  0.9402629  0.5119360
##   11  0.9386008  0.4920613
##   12  0.9378358  0.4832945
##   13  0.9362358  0.4646036
##   14  0.9348511  0.4483304
##   15  0.9336124  0.4313866
##   16  0.9330323  0.4228638
##   17  0.9317832  0.4073346
##   18  0.9309864  0.3947135
##   19  0.9302749  0.3859606
##   20  0.9292918  0.3729634
##   21  0.9284977  0.3593039
##   22  0.9272263  0.3415269
##   23  0.9267822  0.3328631
##   24  0.9255794  0.3165633
##   25  0.9244499  0.2991948
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 1.
```

###Answer:3

```
#Get predictions for validation set using tuned model in Q2
valid_predicts<- predict(model_tuned, valid)

#I'll plot values as follow'
#x=Actual values from valid set
#y=Predicted values from model
CrossTable(x= valid$Personal.Loan,
           y =valid_predicts,
           prop.chisq= FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2000
##
##
##                    | valid_predicts
## valid$Personal.Loan |         0 |         1 | Row Total |
## --------------------|-----------|-----------|-----------|
##                  0 |      1770 |        25 |      1795 |
##                    |     0.986 |     0.014 |     0.897 |
##                    |     0.971 |     0.141 |           |
##                    |     0.885 |     0.013 |           |
## --------------------|-----------|-----------|-----------|
##                  1 |        53 |       152 |       205 |
##                    |     0.259 |     0.741 |     0.102 |
##                    |     0.029 |     0.859 |           |
##                    |     0.026 |     0.076 |           |
## --------------------|-----------|-----------|-----------|
##       Column Total |      1823 |       177 |      2000 |
##                    |     0.911 |     0.088 |           |
## --------------------|-----------|-----------|-----------|
##
##
```

###Answer:4

```
#model_tuned already contains best k and normalization, I can apply it directly to
new_customer data created earlier.
predict_best_k <- predict(model_tuned,new_customer)
print(predict_best_k)
```

```
## [1] 0
## Levels: 0 1
```

###Answer:5

```
#I will set aside 50% of data for training first
set.seed(1)
trainIndex2<- createDataPartition(data_transformed$Personal.Loan, p= 0.5, list = F
ALSE)
train_data<- data_transformed[trainIndex2, ]
remaining_data<- data_transformed[-trainIndex2, ]

#Now I'll split remaining 50% into validation and test
#30% is 0.6 of remaining 50%

validIndex2 <- createDataPartition(remaining_data$Personal.Loan, p= 0.6, list = FA
LSE)
valid_data<-remaining_data[validIndex2, ]
test_data<- remaining_data[-validIndex2, ]
```

```
#Now I'll normalize all sets and -14 is the number of column we remove as Persona
l.Loan is located in 14th column.
prepro<- preProcess(train_data[, -14],method = "range")
train_alpha<- predict(prepro, train_data[,-14])
valid_alpha<- predict(prepro,valid_data[, -14])
test_alpha<- predict(prepro, test_data[,-14])

#now i'll run KNN using the Best k=1, calculated earlier
best_k <- 1
knn_test<- knn(train = train_alpha, test= test_alpha, cl = train_data$Personal.Loa
n, k = best_k)
knn_train<- knn(train = train_alpha, test = train_alpha, cl = train_data$Personal.
Loan, k = best_k)
knn_valid<- knn(train = train_alpha, test = valid_alpha, cl = train_data$Personal.
Loan, k = best_k)
```

```
#plotting confusion matrix
CrossTable(x= test_data$Personal.Loan, y= knn_test, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1000
##
##
##                        | knn_test
## test_data$Personal.Loan |         0 |         1 | Row Total |
## ------------------------|-----------|-----------|-----------|
##                       0 |       874 |        14 |       888 |
##                         |     0.984 |     0.016 |     0.888 |
##                         |     0.972 |     0.139 |           |
##                         |     0.874 |     0.014 |           |
## ------------------------|-----------|-----------|-----------|
##                       1 |        25 |        87 |       112 |
##                         |     0.223 |     0.777 |     0.112 |
##                         |     0.028 |     0.861 |           |
##                         |     0.025 |     0.087 |           |
## ------------------------|-----------|-----------|-----------|
##            Column Total |       899 |       101 |      1000 |
##                         |     0.899 |     0.101 |           |
## ------------------------|-----------|-----------|-----------|
##
##
```

We observe highest accuracy and performance metrics on the Training set. kNN retains training data. When model generates predictions based on training data, “neighbors” it identifies are precisely the data points it is already familiar with, resulting in overly positive outcomes. High level of training accuracy paired with considerably lower validation/test accuracy indicates overfitting, particularly if a small value of k was utilized.

The outcomes should be fairly comparable; however, the Validation set may exhibit marginally superior performance compared to Test set. Validation set was utilized to fine-tune the model i.e. select the optimal value. Consequently, the model is “refined” for that particular dataset.Test set serves as ultimate, impartial evaluator. It embodies data that model has never encountered and was not employed in determining k value. This provides the most accurate anticipation of how model will function with new bank clients in the future.

False Negatives: In context of a bank loan, false negative signifies missed opportunity for the financial institution.

False Positives: A false positive leads to unnecessary marketing expenditures.

Overall Accuracy: Significant decline in accuracy from Training set indicates that the model is having difficulty generalizing to new data.