---

title: “Assignment_02” author: “Jovan Zivak” date: “2025-10-03” output: pdf_document: default html_document: default

```
library(class)
library(caret)

# helper to force console-style output into knitted doc
show_console_output <- function(expr) {
  out <- capture.output(force(expr))
  cat(paste(out, collapse = "\n"), sep = "\n")
}

# import dataset
library(readxl)
UniversalBank <- read_excel("C:/Users/jovan/Downloads/UniversalBank.xlsx")
View(UniversalBank)

# dummies for Education
edu_mm <- model.matrix(~ factor(Education) - 1, data = UniversalBank)
colnames(edu_mm) <- c("Education_1", "Education_2", "Education_3")

UniversalBank <- cbind(
  UniversalBank[, !(names(UniversalBank) %in% c("Education"))],
  as.data.frame(edu_mm)
)

# normalize before modeling
norm_model <- preProcess(UniversalBank, method = c("range"))
UnivBank_normalized <- predict(norm_model, UniversalBank)

# tune k on Income + CCAvg
set.seed(123)
Search_Grid <- expand.grid(k = c(2, 7, 9, 15))
model <- train(`Personal Loan` ~ Income + CCAvg,
               data = UniversalBank,
               method = "knn",
               tuneGrid = Search_Grid,
               preProcess = "range")

best_k <- model$bestTune$k
cat("### Best k from caret tuning:", best_k, "\n\n")
```

```
## ### Best k from caret tuning: 15
```

```
# split 60/40 train/test
set.seed(123)
Index_Train <- createDataPartition(UnivBank_normalized$`Personal Loan`,
                                   p = 0.6, list = FALSE)
Train <- UnivBank_normalized[Index_Train[,1], ]
Test  <- UnivBank_normalized[-Index_Train[,1], ]
```

```r
# labels
y_col <- "Personal Loan"
Train_labels <- factor(Train[[y_col]])
Test_labels  <- factor(Test[[y_col]])

# predictors = drop IDs
drop_cols <- c("ID", "ZIP Code", y_col)
Train_Predictors <- Train[, !(names(Train) %in% drop_cols), drop = FALSE]
Test_Predictors  <- Test[,  !(names(Test)  %in% drop_cols), drop = FALSE]

# kNN with "best k"
set.seed(123)
Predicted_Test_bestk <- knn(train = Train_Predictors,
                            test  = Test_Predictors,
                            cl    = Train_labels,
                            k     = best_k)

# --- QUESTION 1: classify given customer with k = 1 ---
new_customer <- as.data.frame(t(rep(0, ncol(UniversalBank))))
names(new_customer) <- names(UniversalBank)

# values from question 1
new_customer$Age <- 40
new_customer$Experience <- 10
new_customer$Income <- 84
new_customer$Family <- 2
new_customer$CCAvg <- 2
new_customer$Education_1 <- 0
new_customer$Education_2 <- 1
new_customer$Education_3 <- 0
new_customer$Mortgage <- 0
new_customer$`Securities Account` <- 0
new_customer$`CD Account` <- 0
new_customer$Online <- 1
new_customer$CreditCard <- 1
new_customer[[y_col]] <- NA


# normalize with same model
new_customer_norm <- predict(norm_model, new_customer)
new_customer_pred <- new_customer_norm[, !(names(new_customer_norm) %in% drop_cols), drop = FALSE]

# classify with k = 1
set.seed(123)
Predicted_new_k1 <- knn(train = Train_Predictors,
                        test  = new_customer_pred,
                        cl    = Train_labels,
                        k     = 1)

# find a "best k"
cat("### Prediction for new customer (k = 1):", as.character(Predicted_new_k1), "\n\n")
```

```
## ### Prediction for new customer (k = 1): 0
```

```r
# determine balanced choice of k (question 2)
cat("## Determining a balanced choice of k\n\n")
```

```
## ## Determining a balanced choice of k
```

```r
# Review tuning results from caret (model object already created earlier)
show_console_output(model)
```

```
## k-Nearest Neighbors
##
## 5000 samples
##    2 predictor
##
## Pre-processing: re-scaling to [0, 1] (2)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 5000, 5000, 5000, 5000, 5000, 5000, ...
## Resampling results across tuning parameters:
##
##   k   RMSE       Rsquared   MAE
##    2  0.2788240  0.2505694  0.09286784
##    7  0.2525954  0.3005501  0.10252022
##    9  0.2487865  0.3122690  0.10390541
##   15  0.2433937  0.3311692  0.10632997
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 15.
```

```r
# Extract and display best k value
cat("\n\n### Optimal k chosen by cross-validation:", best_k, "\n")
```

```
##
##
## ### Optimal k chosen by cross-validation: 15
```

```r
# print confusion matrix for "best k" (question 3)
cat("## Confusion Matrix: 60/40 Split (best k)\n\n")
```

```
## ## Confusion Matrix: 60/40 Split (best k)
```

```r
show_console_output(
  gmodels::CrossTable(x = Test_labels,
                      y = Predicted_Test_bestk,
                      prop.chisq = FALSE)
)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
```

```
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2000
##
##
##              | Predicted_Test_bestk
##  Test_labels |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |      1788 |        10 |      1798 |
##              |     0.994 |     0.006 |     0.899 |
##              |     0.927 |     0.139 |           |
##              |     0.894 |     0.005 |           |
## -------------|-----------|-----------|-----------|
##            1 |       140 |        62 |       202 |
##              |     0.693 |     0.307 |     0.101 |
##              |     0.073 |     0.861 |           |
##              |     0.070 |     0.031 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      1928 |        72 |      2000 |
##              |     0.964 |     0.036 |           |
## -------------|-----------|-----------|-----------|
##
##
```

```
# classify with "best k" (question 4)
set.seed(123)
Predicted_new_bestk <- knn(train = Train_Predictors,
                           test  = new_customer_pred,
                           cl    = Train_labels,
                           k     = best_k)
cat("### Prediction for new customer (best k):", as.character(Predicted_new_bestk), "\n\n")
```

```
## ### Prediction for new customer (best k): 0
```

```
# set "best k" to determined value from before
best_k <- 15
cat("### Using best_k =", best_k, "for 50/30/20 experiment\n\n")
```

```
## ### Using best_k = 15 for 50/30/20 experiment
```

```
# 50/30/20 split (question 5)
set.seed(123)
Index_Train50 <- createDataPartition(UnivBank_normalized[[y_col]], p = 0.5, list = FALSE)
Train50 <- UnivBank_normalized[Index_Train50[,1], ]
Temp    <- UnivBank_normalized[-Index_Train50[,1], ]

Index_Valid30 <- createDataPartition(Temp[[y_col]], p = 0.6, list = FALSE)
Valid30 <- Temp[Index_Valid30[,1], ]
```

```r
Test20  <- Temp[-Index_Valid30[,1], ]

Train50_labels <- factor(Train50[[y_col]])
Valid30_labels <- factor(Valid30[[y_col]])
Test20_labels  <- factor(Test20[[y_col]])

# predictors by dropping IDs + target
Train50_Predictors <- Train50[, !(names(Train50) %in% drop_cols), drop = FALSE]
Valid30_Predictors <- Valid30[, !(names(Valid30) %in% drop_cols), drop = FALSE]
Test20_Predictors  <- Test20[,  !(names(Test20)  %in% drop_cols), drop = FALSE]

set.seed(123)
Pred_Train50 <- knn(train = Train50_Predictors, test = Train50_Predictors, cl = Train50_labels, k = bes
set.seed(123)
Pred_Valid30 <- knn(train = Train50_Predictors, test = Valid30_Predictors, cl = Train50_labels, k = bes
set.seed(123)
Pred_Test20  <- knn(train = Train50_Predictors, test = Test20_Predictors,  cl = Train50_labels, k = bes

# not-so-confusing matrices being printed
cat("## Confusion Matrix: TRAIN (50%)\n\n")
```

```
## ## Confusion Matrix: TRAIN (50%)
```

```r
show_console_output(
  gmodels::CrossTable(x = Train50_labels, y = Pred_Train50, prop.chisq = FALSE)
)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2500
##
##
##                | Pred_Train50
## Train50_labels |         0 |         1 | Row Total |
## ---------------|-----------|-----------|-----------|
##              0 |      2265 |         6 |      2271 |
##                |     0.997 |     0.003 |     0.908 |
##                |     0.940 |     0.066 |           |
##                |     0.906 |     0.002 |           |
## ---------------|-----------|-----------|-----------|
##              1 |       144 |        85 |       229 |
##                |     0.629 |     0.371 |     0.092 |
##                |     0.060 |     0.934 |           |
##                |     0.058 |     0.034 |           |
```

```
## ---------------|-----------|-----------|-----------|
##   Column Total |      2409 |        91 |      2500 |
##                |     0.964 |     0.036 |           |
## ---------------|-----------|-----------|-----------|
##
##
```

```
cat("\n\n## Confusion Matrix: VALID (30%)\n\n")
```

```
##
##
## ## Confusion Matrix: VALID (30%)
```

```
show_console_output(
  gmodels::CrossTable(x = Valid30_labels, y = Pred_Valid30, prop.chisq = FALSE)
)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1500
##
##
##                | Pred_Valid30
## Valid30_labels |         0 |         1 | Row Total |
## ---------------|-----------|-----------|-----------|
##              0 |      1349 |         8 |      1357 |
##                |     0.994 |     0.006 |     0.905 |
##                |     0.932 |     0.154 |           |
##                |     0.899 |     0.005 |           |
## ---------------|-----------|-----------|-----------|
##              1 |        99 |        44 |       143 |
##                |     0.692 |     0.308 |     0.095 |
##                |     0.068 |     0.846 |           |
##                |     0.066 |     0.029 |           |
## ---------------|-----------|-----------|-----------|
##   Column Total |      1448 |        52 |      1500 |
##                |     0.965 |     0.035 |           |
## ---------------|-----------|-----------|-----------|
##
##
```

```
cat("\n\n## Confusion Matrix: TEST (20%)\n\n")
```

```
##
##
## ## Confusion Matrix: TEST (20%)
```

```
show_console_output(
  gmodels::CrossTable(x = Test20_labels, y = Pred_Test20, prop.chisq = FALSE)
)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1000
##
##
##                | Pred_Test20
## Test20_labels |         0 |         1 | Row Total |
## --------------|-----------|-----------|-----------|
##             0 |       889 |         3 |       892 |
##               |     0.997 |     0.003 |     0.892 |
##               |     0.923 |     0.081 |           |
##               |     0.889 |     0.003 |           |
## --------------|-----------|-----------|-----------|
##             1 |        74 |        34 |       108 |
##               |     0.685 |     0.315 |     0.108 |
##               |     0.077 |     0.919 |           |
##               |     0.074 |     0.034 |           |
## --------------|-----------|-----------|-----------|
##  Column Total |       963 |        37 |      1000 |
##               |     0.963 |     0.037 |           |
## --------------|-----------|-----------|-----------|
##
##
```