Report: SVM Bank Loan - Predictive Modelling using SVM in RStudio

---

Use the dataset to perform the following tasks. If you use a seed, set it to a value of 1209.

1) Prepare the dataset. Describe the process in detail. Compute and describe the Summary Statistics. (10 pts)

ANS 1)

- The libraries which I used to conduct the SVM model were (caret and kernlab).

- The dataset comprises 807 observations of 15 variables. While I was exploring the dataset, I found no missing(n/a) values.

- I performed stratified sampling where I performed the 80% – 20% split where 80% will serve as the training data and 20% will serve as the test data.

- I performed factorization upon variables such as Family , Education, PersonalLoan , SecuritiesAccount , CDAccount, Online , Credit Card , I discarded variables such as " ID" ,"ZIPCode" and X (row number), they were unique Identifiers.

- Next, I normalized the variables and evaluated the final summary (normalized data)

```
> # Import the CSV file
> ub.org <- read.csv("C:/Users/smukherjee3/Downloads/UB6PM-1.csv")
> # Structure of dataset
> str(ub.org)
'data.frame':   807 obs. of  15 variables:
 $ X                : int  1 2 3 6 7 9 10 11 13 15 ...
 $ ID               : int  4533 4409 955 4860 3272 3920 3767 4942 4302 1321 ...
 $ Age              : int  48 64 37 34 52 64 59 28 49 31 ...
 $ Experience       : int  22 40 12 8 27 34 35 4 24 7 ...
 $ Income           : int  133 181 169 165 93 179 108 112 130 192 ...
 $ ZIPCode          : int  90073 93403 91107 91107 90291 90024 90245 90049 92677 90250 ...
 $ Family           : int  2 2 2 1 4 2 4 2 4 1 ...
 $ CCAvg            : num  3.1 2.3 5.2 7 4.1 4.5 3.8 1.6 1.1 0 ...
 $ Education        : int  2 2 3 3 2 3 2 2 1 2 ...
 $ Mortgage         : int  0 0 249 541 0 400 304 0 281 0 ...
 $ PersonalLoan     : int  1 1 1 1 1 1 1 1 1 1 ...
 $ SecuritiesAccount: int  0 0 0 0 0 0 0 0 0 0 ...
 $ CDAccount        : int  0 1 0 0 0 0 0 0 1 0 ...
 $ Online           : int  1 1 1 0 0 0 1 1 1 1 ...
 $ CreditCard       : int  0 1 0 0 1 0 0 0 0 0 ...
> # List few records from dataset
> head(ub.org)
  X   ID Age Experience Income ZIPCode Family CCAvg Education Mortgage PersonalLoan SecuritiesAccount CDAccount Online
1 1 4533  48         22    133   90073      2   3.1         2        0            1                 0         0      1
2 2 4409  64         40    181   93403      2   2.3         2        0            1                 0         1      1
3 3  955  37         12    169   91107      2   5.2         3      249            1                 0         0      1
4 6 4860  34          8    165   91107      1   7.0         3      541            1                 0         0      0
5 7 3272  52         27     93   90291      4   4.1         2        0            1                 0         0      0
6 9 3920  64         34    179   90024      2   4.5         3      400            1                 0         0      0
  CreditCard
1          0
2          1
3          0
4          0
5          1
6          0
```

```
> summary(ub.org[c("Age","Experience","Income","Family","CCAvg",
+                  "Education","Mortgage","PersonalLoan",
+                  "SecuritiesAccount","CDAccount","Online","CreditCard")])
      Age          Experience        Income           Family          CCAvg          Education        Mortgage    
 Min.   :24.00   Min.   :-2.00   Min.   :  8.00   Min.   :1.000   Min.   : 0.000   Min.   :1.000   Min.   :  0.00  
 1st Qu.:36.00   1st Qu.:11.00   1st Qu.: 42.00   1st Qu.:1.000   1st Qu.: 0.800   1st Qu.:1.000   1st Qu.:  0.00  
 Median :46.00   Median :21.00   Median : 72.00   Median :2.000   Median : 1.600   Median :2.000   Median :  0.00  
 Mean   :45.74   Mean   :20.51   Mean   : 83.75   Mean   :2.414   Mean   : 2.166   Mean   :1.953   Mean   : 58.98  
 3rd Qu.:55.00   3rd Qu.:30.00   3rd Qu.:122.00   3rd Qu.:3.000   3rd Qu.: 2.900   3rd Qu.:3.000   3rd Qu.: 99.00  
 Max.   :67.00   Max.   :42.00   Max.   :204.00   Max.   :4.000   Max.   :10.000   Max.   :3.000   Max.   :612.00  
  PersonalLoan    SecuritiesAccount   CDAccount           Online         CreditCard  
 Min.   :0.0000   Min.   :0.0000    Min.   :0.00000   Min.   :0.0000   Min.   :0.00  
 1st Qu.:0.0000   1st Qu.:0.0000    1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.00  
 Median :0.0000   Median :0.0000    Median :0.00000   Median :1.0000   Median :0.00  
 Mean   :0.2292   Mean   :0.1165    Mean   :0.08922   Mean   :0.5948   Mean   :0.28  
 3rd Qu.:0.0000   3rd Qu.:0.0000    3rd Qu.:0.00000   3rd Qu.:1.0000   3rd Qu.:1.00  
 Max.   :1.0000   Max.   :1.0000    Max.   :1.00000   Max.   :1.0000   Max.   :1.00  
> set.seed(1209)
> idx <- sample(nrow(ub.org), 0.8*nrow(ub.org))   # 80% training
> length(idx)
[1] 645
> idx <- createDataPartition(y = ub.org$PersonalLoan, p = 0.8, list = FALSE)
> length(idx)
[1] 646
> ub.org$Family <- factor(ub.org$Family)
> ub.org$Education <- factor(ub.org$Education)
> ub.org$PersonalLoan <- factor(ub.org$PersonalLoan)
> ub.org$SecuritiesAccount <- factor(ub.org$SecuritiesAccount)
> ub.org$CDAccount <- factor(ub.org$CDAccount)
> ub.org$Online <- factor(ub.org$Online)
> ub.org$CreditCard <- factor(ub.org$CreditCard)
> normalize <- function(x){
+   return( (x - min(x)) / (max(x) - min(x)) )
+ }
> ub.df <- as.data.frame(lapply(
+   ub.org[c("Age","Experience","Income","CCAvg","Mortgage")], normalize))
> ub.final <- cbind(ub.df,
+                   ub.org[c("Family","Education","PersonalLoan",
+                            "SecuritiesAccount","CDAccount",
+                            "Online","CreditCard")])
> summary(ub.final)
      Age           Experience         Income            CCAvg            Mortgage       Family  Education PersonalLoan
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   1:223   1:304     0:622       
 1st Qu.:0.2791   1st Qu.:0.2955   1st Qu.:0.1735   1st Qu.:0.0800   1st Qu.:0.00000   2:223   2:237     1:185       
 Median :0.5116   Median :0.5227   Median :0.3265   Median :0.1600   Median :0.00000   3:165   3:266                 
 Mean   :0.5056   Mean   :0.5116   Mean   :0.3865   Mean   :0.2166   Mean   :0.09636   4:196                         
 3rd Qu.:0.7209   3rd Qu.:0.7273   3rd Qu.:0.5816   3rd Qu.:0.2900   3rd Qu.:0.16176                                 
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.00000                                 
 SecuritiesAccount CDAccount Online  CreditCard
 0:713             0:735     0:327   0:581     
 1: 94             1: 72     1:480   1:226     
```

Here, above you can see two summaries 1st one(un-normalised data) denoted by **summary(ub.org)** and the next summary (normalized data) denoted by **summary(ub.final)**

2) Build an SVM classifier to predict whether a customer will accept a personal loan or not. Describe your process to build and fine tune the classifier. (20 pts)

(ANS 2)

The target Variable for this model is Personal Loan . next I loaded the training the in the svm_model by using method = “ svmRadial” , I added the train control(tr control) where I implemented cross-validation(cv) for 10 folds , I used this step to nullify the class imbalance of the target variable (personalLoan), I used “center” and “scale” methods to perform data pre-processing. The Tuning parameter used was sigma , hence the final values used for this model were sigma= 0.0485394 and c = 1.

RESULTS SHOWN BELOW BEFORE FINE- TUNING THE MODEL.

```
> train.df <- ub.final[idx, ]
> test.df <- ub.final[-idx, ]
> train_labels <- ub.final$PersonalLoan[idx]   # 80%
> test_labels  <- ub.final$PersonalLoan[-idx]  # 20%
> svm_model <- train(PersonalLoan ~ .,
+                    data = train.df,
+                    method = "svmRadial",
+                    trControl = trainControl(method = "cv", number = 10),
+                    preProcess = c("center", "scale"))
> svm_model
Support Vector Machines with Radial Basis Function Kernel

646 samples
 11 predictor
  2 classes: '0', '1'

Pre-processing: centered (14), scaled (14)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 582, 582, 580, 581, 582, 581, ...
Resampling results across tuning parameters:

  C     Accuracy   Kappa
  0.25  0.9069442  0.7175673
  0.50  0.9348543  0.8108255
  1.00  0.9410810  0.8293514

Tuning parameter 'sigma' was held constant at a value of 0.04853934
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were sigma = 0.04853934 and C = 1.
```

EVALUATING CONFUSION MATRIX AND STATISTICS

```
The final values used for the model were sigma = 0.04853934 and C = 1.
> svm_predictions <- predict(svm_model, test.df)
> confusionMatrix(svm_predictions, test_labels, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 126   2
         1   3  30

               Accuracy : 0.9689
                 95% CI : (0.929, 0.9898)
    No Information Rate : 0.8012
    P-Value [Acc > NIR] : 2.913e-10

                  Kappa : 0.9036

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9375
            Specificity : 0.9767
         Pos Pred Value : 0.9091
         Neg Pred Value : 0.9844
             Prevalence : 0.1988
         Detection Rate : 0.1863
   Detection Prevalence : 0.2050
      Balanced Accuracy : 0.9571

       'Positive' Class : 1
```

- We can see initial accuracy of 0.969 and a sensitivity of 0.9375, the true -negative value is 126.

NEXT STEP , FINE TUNING THE MODEL

In order , to fine tune the model I used sigma from **c(0.01 to 0.05)** and C from **c(0.5 to 3),** method used was **“SVMRadial”,** tune grid selected was **“svm_grid”,** I performed training control by implementing **5 fold cross- validation**

```
        Positive Class : 1

> #FINE TUNING THE MODE
> svm_grid <- expand.grid(
+   sigma = c(0.01, 0.02, 0.03, 0.05),
+   C = c(0.5, 1, 2, 3)
+ )
> svm_model_tuned <- train(PersonalLoan ~ .,
+                          data = train.df,
+                          method = "svmRadial",
+                          tuneGrid = svm_grid,
+                          trControl = trainControl(method = "cv", number = 5),
+                          preProcess = c("center", "scale"))
> svm_model_tuned
Support Vector Machines with Radial Basis Function Kernel

646 samples
 11 predictor
  2 classes: '0', '1'

Pre-processing: centered (14), scaled (14)
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 516, 517, 517, 517, 517
Resampling results across tuning parameters:

  sigma  C    Accuracy   Kappa
  0.01   0.5  0.8900894  0.6626501
  0.01   1.0  0.9055814  0.7139624
  0.01   2.0  0.9117710  0.7368823
  0.01   3.0  0.9179487  0.7562332
  0.02   0.5  0.9117591  0.7334834
  0.02   1.0  0.9179368  0.7551452
  0.02   2.0  0.9349553  0.8086301
  0.02   3.0  0.9333930  0.8070955
  0.03   0.5  0.9148479  0.7447393
  0.03   1.0  0.9303041  0.7943282
  0.03   2.0  0.9303041  0.7990624
  0.03   3.0  0.9349434  0.8141153
  0.05   0.5  0.9318664  0.8001043
  0.05   1.0  0.9365176  0.8155925
  0.05   2.0  0.9396064  0.8275846
  0.05   3.0  0.9411330  0.8341785

Accuracy was used to select the optimal model using the largest value.
The final values used for the model were sigma = 0.05 and C = 3.
```

NOW, EVALUATING THE CONFUSION MATRIX FOR TUNED MODEL

```
> svm_predictions_tuned <- predict(svm_model_tuned, test.df)
> confusionMatrix(svm_predictions_tuned, test_labels, positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 125   2
         1   4  30

               Accuracy : 0.9627
                 95% CI : (0.9207, 0.9862)
    No Information Rate : 0.8012
    P-Value [Acc > NIR] : 1.936e-09

                  Kappa : 0.8857

 Mcnemar's Test P-Value : 0.6831

            Sensitivity : 0.9375
            Specificity : 0.9690
         Pos Pred Value : 0.8824
         Neg Pred Value : 0.9843
             Prevalence : 0.1988
         Detection Rate : 0.1863
   Detection Prevalence : 0.2112
      Balanced Accuracy : 0.9532

       'Positive' Class : 1
```

- **I noticed that the accuracy slightly dropped from 0.9689 to 0.9627 after fine tuning it.**
- **The sensitivity remained the same which is 0.9375**
- **The True Negatives were 125.**

Hence conclusion after comparing both the models were that, the accuracy reduced extremely slightly and the sensitivity were the same, hence fine-tuning did not play any significant role in improving model performance.

3) How would the following customer be classified? Produce the output in the report. (10 pts)
Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education =2, Mortgage = 0, Securities Account = 0, CD Account = 0, Online = 1, and Credit Card = 1.

Ans**) It was found that this customer won't be accepting the loan.**

I made sure to normalize all the variables again before predicting whether the customer would be accepting the Loan or not.

```
> #Q3. Evaluating new data point
> new.customer <- data.frame(Age=40, Experience=10, Income=84,
+                            CCAvg=2, Mortgage=0,
+                            Family=2, Education=2,
+                            SecuritiesAccount=0, CDAccount=0,
+                            Online=1, CreditCard=1)
> normalize2 <- function(x, min_val, max_val){
+   return( (x - min_val) / (max_val - min_val) )
+ }
> new.customer.norm <- data.frame(
+   Age = normalize2(40, min(ub.org$Age), max(ub.org$Age)),
+   Experience = normalize2(10, min(ub.org$Experience), max(ub.org$Experience)),
+   Income = normalize2(84, min(ub.org$Income), max(ub.org$Income)),
+   CCAvg = normalize2(2, min(ub.org$CCAvg), max(ub.org$CCAvg)),
+   Mortgage = normalize2(0, min(ub.org$Mortgage), max(ub.org$Mortgage)),
+   Family = factor(2, levels=levels(ub.org$Family)),
+   Education = factor(2, levels=levels(ub.org$Education)),
+   SecuritiesAccount = factor(0, levels=levels(ub.org$SecuritiesAccount)),
+   CDAccount = factor(0, levels=levels(ub.org$CDAccount)),
+   Online = factor(1, levels=levels(ub.org$Online)),
+   CreditCard = factor(1, levels=levels(ub.org$CreditCard))
+ )
> predict(svm_model_tuned, new.customer.norm)
[1] 0
Levels: 0 1
>
```

The predicted class 0, the customer will not be accepting the loan.

APPENDIX

#ANS 1.

```
install.packages("caret")
install.packages("kernlab")
library(caret)
library(kernlab)   # for svmRadial
```

 # Import the CSV file

```
ub.org <- read.csv("C:/Users/smukherjee3/Downloads/UB6PM-1.csv")
```

 # Structure of dataset

```
str(ub.org)
```

 # List few records from dataset

```
head(ub.org)
summary(ub.org[c("Age","Experience","Income","Family","CCAvg",
            "Education","Mortgage","PersonalLoan",
            "SecuritiesAccount","CDAccount","Online","CreditCard")])

set.seed(1209)
idx <- sample(nrow(ub.org), 0.8*nrow(ub.org))   # 80% training
length(idx)
idx <- createDataPartition(y = ub.org$PersonalLoan, p = 0.8, list = FALSE)
length(idx)

#Factorising the variables
ub.org$Family <- factor(ub.org$Family)
ub.org$Education <- factor(ub.org$Education)
ub.org$PersonalLoan <- factor(ub.org$PersonalLoan)
ub.org$SecuritiesAccount <- factor(ub.org$SecuritiesAccount)
ub.org$CDAccount <- factor(ub.org$CDAccount)
ub.org$Online <- factor(ub.org$Online)
ub.org$CreditCard <- factor(ub.org$CreditCard)
```

#normalising the variables

```
normalize <- function(x){
  return( (x - min(x)) / (max(x) - min(x)) )
}

ub.df <- as.data.frame(lapply(
  ub.org[c("Age","Experience","Income","CCAvg","Mortgage")], normalize))

ub.final <- cbind(ub.df,
             ub.org[c("Family","Education","PersonalLoan",
                  "SecuritiesAccount","CDAccount",
```

```
                    "Online","CreditCard")])

summary(ub.final)

train.df <- ub.final[idx, ]  # ANSWER 2

test.df <- ub.final[-idx, ]

train_labels <- ub.final$PersonalLoan[idx]   # 80%
test_labels  <- ub.final$PersonalLoan[-idx]  # 20%


svm_model <- train(PersonalLoan ~ .,
               data = train.df,
               method = "svmRadial",
               trControl = trainControl(method = "cv", number = 10),
               preProcess = c("center", "scale"))
svm_model

svm_predictions <- predict(svm_model, test.df)

confusionMatrix(svm_predictions, test_labels, positive = "1")



#FINE TUNING THE MODEL
svm_grid <- expand.grid(
  sigma = c(0.01, 0.02, 0.03, 0.05),
  C = c(0.5, 1, 2, 3)
)

svm_model_tuned <- train(PersonalLoan ~ .,
                    data = train.df,
                    method = "svmRadial",
                    tuneGrid = svm_grid,
                    trControl = trainControl(method = "cv", number = 5),
                    preProcess = c("center", "scale"))
svm_model_tuned


svm_predictions_tuned <- predict(svm_model_tuned, test.df)

confusionMatrix(svm_predictions_tuned, test_labels, positive = "1")
```

#Q3. Evaluating new data point

```
new.customer <- data.frame(Age=40, Experience=10, Income=84,
                           CCAvg=2, Mortgage=0,
                           Family=2, Education=2,
                           SecuritiesAccount=0, CDAccount=0,
                           Online=1, CreditCard=1)
```

#Normalising the variables of the new data point

```
normalize2 <- function(x, min_val, max_val){
return( (x - min_val) / (max_val - min_val) )
}
```

#ANS 3)

```
new.customer.norm <- data.frame(
  Age = normalize2(40, min(ub.org$Age), max(ub.org$Age)),
  Experience = normalize2(10, min(ub.org$Experience), max(ub.org$Experience)),
  Income = normalize2(84, min(ub.org$Income), max(ub.org$Income)),
  CCAvg = normalize2(2, min(ub.org$CCAvg), max(ub.org$CCAvg)),
  Mortgage = normalize2(0, min(ub.org$Mortgage), max(ub.org$Mortgage)),
  Family = factor(2, levels=levels(ub.org$Family)),
  Education = factor(2, levels=levels(ub.org$Education)),
  SecuritiesAccount = factor(0, levels=levels(ub.org$SecuritiesAccount)),
  CDAccount = factor(0, levels=levels(ub.org$CDAccount)),
  Online = factor(1, levels=levels(ub.org$Online)),
  CreditCard = factor(1, levels=levels(ub.org$CreditCard))
)

predict(svm_model_tuned, new.customer.norm)
```