Support Vector Machine (SVM) Verkhovod

# Download the data

set.seed(123)  
setwd('C:/Users/VerkhovodTS/Desktop/R')  
f\_train <- read.csv2('clients\_train.csv', header = TRUE, encoding = 'UNICOD')  
f\_test <- read.csv2('clients\_test.csv', header = TRUE, encoding = 'UNICOD')  
f\_train <- f\_train[-1]  
f\_test <- f\_test[-1]

# Fitting SVM model

# install.packages('e1071')  
library(e1071)

## Warning: package 'e1071' was built under R version 3.6.3

class\_svm\_l = svm(DELAY ~ LOAN\_PRODUCT\_TYPE + LOAN.TERM, data = f\_train, kernel = 'linear')  
summary(class\_svm\_l)

##   
## Call:  
## svm(formula = DELAY ~ LOAN\_PRODUCT\_TYPE + LOAN.TERM, data = f\_train,   
## kernel = "linear")  
##   
##   
## Parameters:  
## SVM-Type: eps-regression   
## SVM-Kernel: linear   
## cost: 1   
## gamma: 0.5   
## epsilon: 0.1   
##   
##   
## Number of Support Vectors: 3047

Висновок: для навчання базової моделі, заснованої на методі опорних векторів, вибрано лінійне ядро.

# Predicting

p <- predict(class\_svm\_l, f\_test[, c('LOAN\_PRODUCT\_TYPE','LOAN.TERM')])  
y <- ifelse(p > 0.5, 1, 0)

Висновок: визначено класи об’єктів (вектор у). ## Confusion Matrix

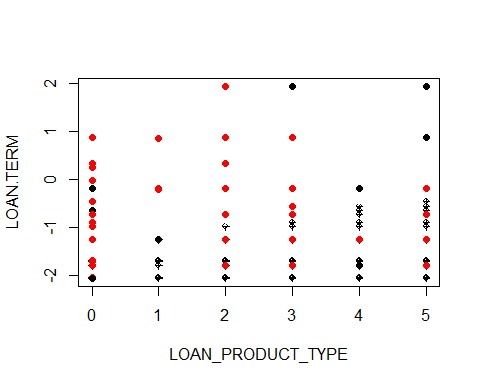
cm = table(f\_test[, 'DELAY'], y)  
print(cm)

## y  
## 0 1  
## 0 200 139  
## 1 161 300

Висновок: точність моделі – (200+300) / 800 = 62,5 %, частка невірно класифікованих випадків – (161+139) / 800 = 37,5 %. Чутливість – 300 / (161+300) = 65,08 %, специфічність – 200 / (200+139) = 59,00 %, тобто модель більш чутлива до виявлення позитивних випадків (кредиторів, що мають заборгованість).

# Visualising the Test set results

xgrid = expand.grid(LOAN\_PRODUCT\_TYPE = f\_test$LOAN\_PRODUCT\_TYPE, LOAN.TERM = f\_test$LOAN.TERM)  
ygrid = predict(class\_svm\_l, xgrid)  
plot(xgrid, col = as.numeric(ygrid), pch = 10, cex = .9)  
points(f\_test[, c('LOAN\_PRODUCT\_TYPE','LOAN.TERM')], col = as.factor(f\_test$DELAY), pch = 19)



Висновок: на графіку світлим позначені випадки затримки з повернення кредиту, темним – хороші кредитори. Червоним виділена зона високої ймовірності неповернення кредиту. Модель описує лінійний варіант розподіляючої кривої.

**Fitting RBF-kernel model**

# install.packages('e1071')  
library(e1071)  
class\_svm\_r = svm(DELAY ~ LOAN\_PRODUCT\_TYPE + LOAN.TERM, data = f\_train, kernel = 'radial')  
summary(class\_svm\_r)

##   
## Call:  
## svm(formula = DELAY ~ LOAN\_PRODUCT\_TYPE + LOAN.TERM, data = f\_train,   
## kernel = "radial")  
##   
##   
## Parameters:  
## SVM-Type: eps-regression   
## SVM-Kernel: radial   
## cost: 1   
## gamma: 0.5   
## epsilon: 0.1   
##   
##   
## Number of Support Vectors: 2240

Висновок: для навчання моделі, заснованої на методі опорних векторів, вибрано нелінійне ядро.

**Predicting**

p <- predict(class\_svm\_r, f\_test[, c('LOAN\_PRODUCT\_TYPE','LOAN.TERM')])  
y <- ifelse(p > 0.5, 1, 0)

Висновок: визначені класи об’єктів (вектор у). ## Confusion Matrix

cm = table(f\_test[, 'DELAY'], y)  
print(cm)

## y  
## 0 1  
## 0 181 158  
## 1 123 338

Висновок: точність моделі – (181+338) / 800 = 64,88 %, частка невірно класифікованих випадків – (123+158) / 800 = 35,13 %. Чутливість – 338 / (123+338) = 73,32 %, специфічність – 181 / (181+158) = 53,39 %, тобто модель більш чутлива до виявлення позитивних випадків (кредиторів, що мають заборгованість). Часка вірно класифікованих випадків у моделі з нелінійним ядром краща, ніж з лінійним. **Visualising the Test set results**

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.6.3

set = f\_test[,c('LOAN\_PRODUCT\_TYPE','LOAN.TERM','DELAY')]  
X1 = seq(min(set['LOAN\_PRODUCT\_TYPE']) - 1, max(set['LOAN\_PRODUCT\_TYPE']) + 1, by = 0.01)  
X2 = seq(min(set['LOAN.TERM']) - 1, max(set['LOAN.TERM']) + 1, by = 0.01)  
grid\_set = expand.grid(X1, X2)  
colnames(grid\_set) = c('LOAN\_PRODUCT\_TYPE', 'LOAN.TERM')  
p\_grid = predict(class\_svm\_r, grid\_set)  
y\_grid <- ifelse(p\_grid > 0.5, 1, 0)  
plot(set[, -3],  
 main = 'SVM',  
 xlab = 'LOAN\_PRODUCT\_TYPE', ylab = 'LOAN.TERM',  
 xlim = range(X1), ylim = range(X2))  
contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)  
points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, '#ed5e26', '#91b368'))  
points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'red3', 'green4'))

