Lab04: Support Vector Machines

Handed out: Wednesday, March 29, 2023

Return date: Saturday, April 8, 2023, at the ELEARNING link `Lab04Submit` in the `Lab04` folder.

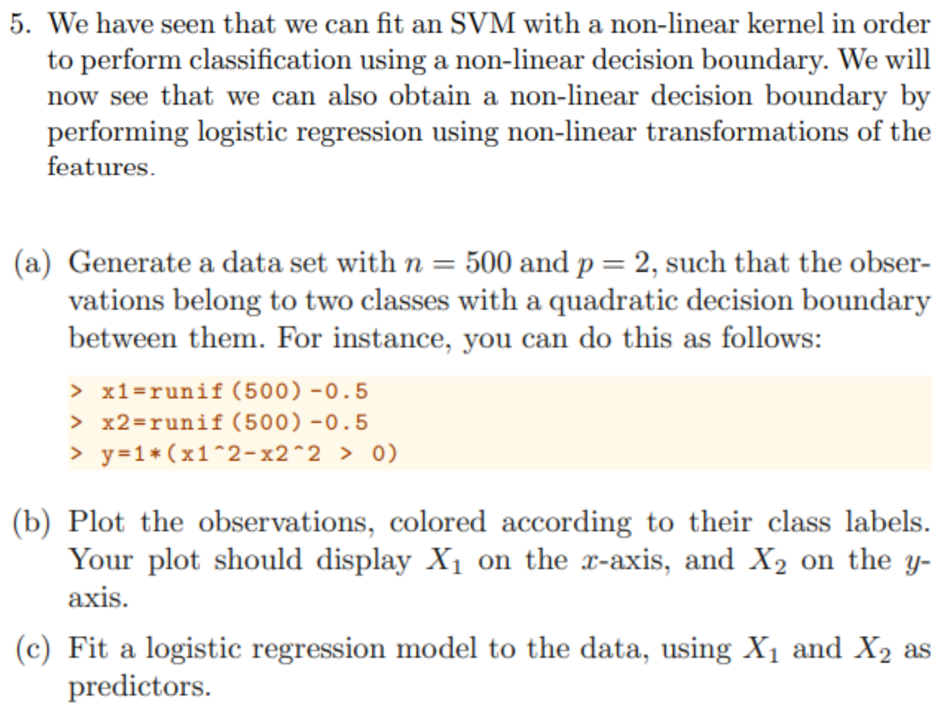
Objectives: Work with different support vector machine algorithms and datasets

Grades: This lab counts 16 % towards your final grade

Format of answer: Your answers (statistical figures and verbal description) should be submitted electronically as Word document. Add a running title with the following information: Lab04, your name and page numbers. Use this document as template: add your answers for each subtask, i.e., 1 (a) etc., in a red color as well as any requested statistical figures. Trial and error answers will lead to a deduction of points. You are expected to hand in professionally formatted answers: use a fixed pitch font, like `Courier New`, for any R code and output.

Support Vector Machines [16 points]

Task 1: You will answer an applied exercise 5 in James et al., 2021. *An Introduction to Statistical Learning with Application in R*. pages 399 and 400. Please follow the sequence of tasks/questions in the exercises. [5 points]



5. We have seen that we can fit an SVM with a non-linear kernel in order to perform classification using a non-linear decision boundary. We will now see that we can also obtain a non-linear decision boundary by performing logistic regression using non-linear transformations of the features.

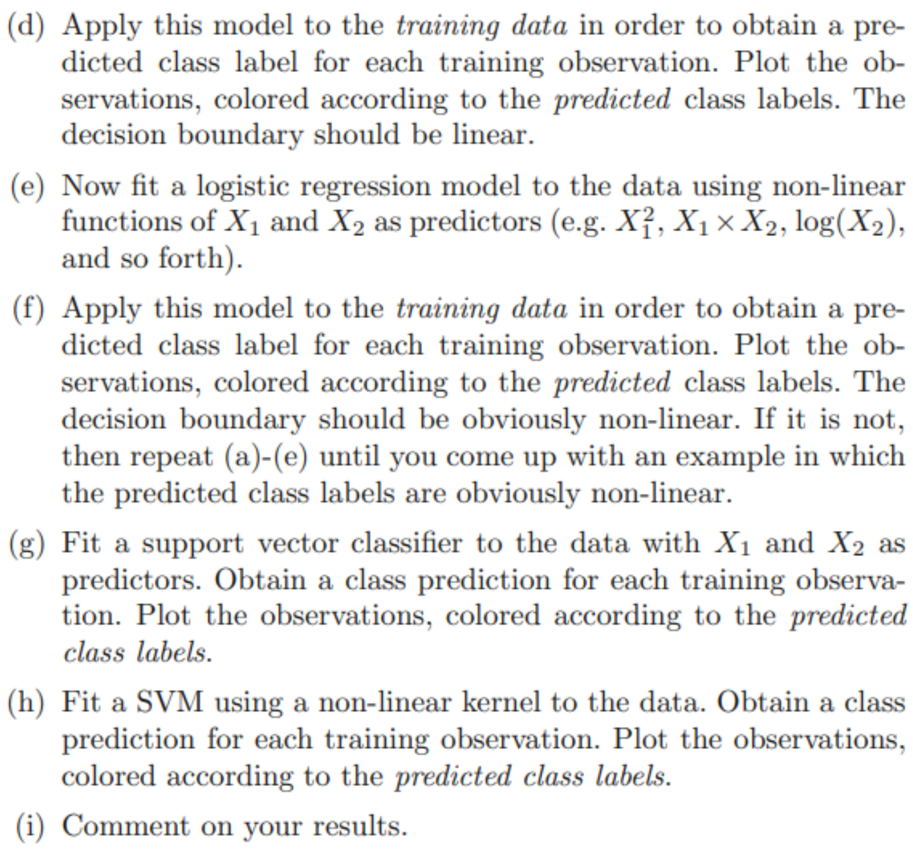
(a) Generate a data set with $n = 500$ and $p = 2$, such that the observations belong to two classes with a quadratic decision boundary between them. For instance, you can do this as follows:

```
> x1=runif(500)-0.5
> x2=runif(500)-0.5
> y=1*(x1^2-x2^2 > 0)
```

(b) Plot the observations, colored according to their class labels. Your plot should display X_1 on the x-axis, and X_2 on the y-axis.

(c) Fit a logistic regression model to the data, using X_1 and X_2 as predictors.

Jim



(d) Apply this model to the *training data* in order to obtain a predicted class label for each training observation. Plot the observations, colored according to the *predicted* class labels. The decision boundary should be linear.

(e) Now fit a logistic regression model to the data using non-linear functions of X_1 and X_2 as predictors (e.g. X_1^2, $X_1 \times X_2$, $\log(X_2)$, and so forth).

(f) Apply this model to the *training data* in order to obtain a predicted class label for each training observation. Plot the observations, colored according to the *predicted* class labels. The decision boundary should be obviously non-linear. If it is not, then repeat (a)-(e) until you come up with an example in which the predicted class labels are obviously non-linear.

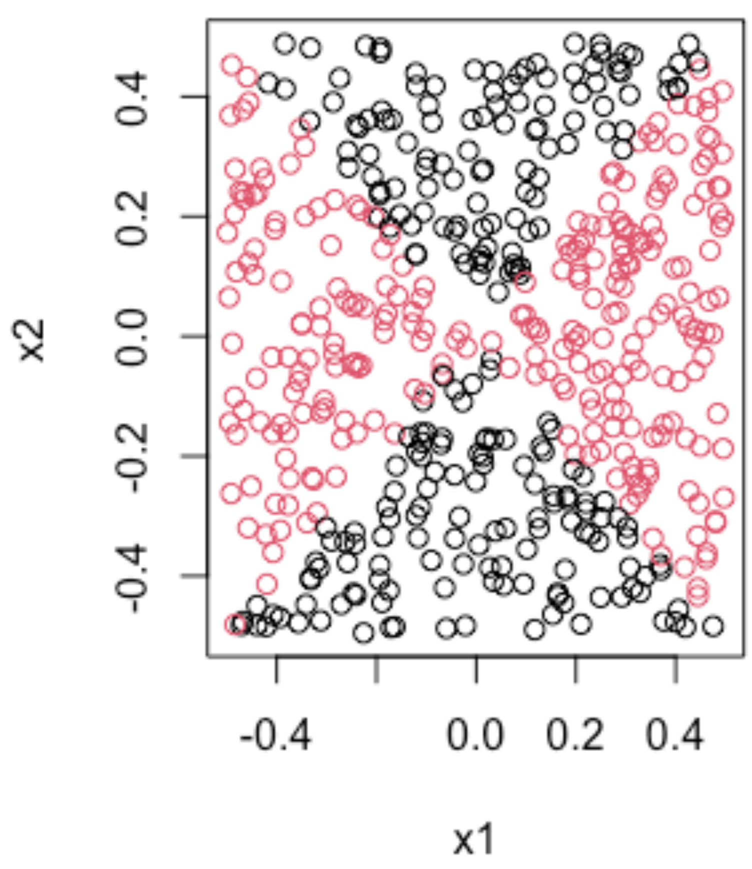
(g) Fit a support vector classifier to the data with X_1 and X_2 as predictors. Obtain a class prediction for each training observation. Plot the observations, colored according to the *predicted class labels*.

(h) Fit a SVM using a non-linear kernel to the data. Obtain a class prediction for each training observation. Plot the observations, colored according to the *predicted class labels*.

(i) Comment on your results.

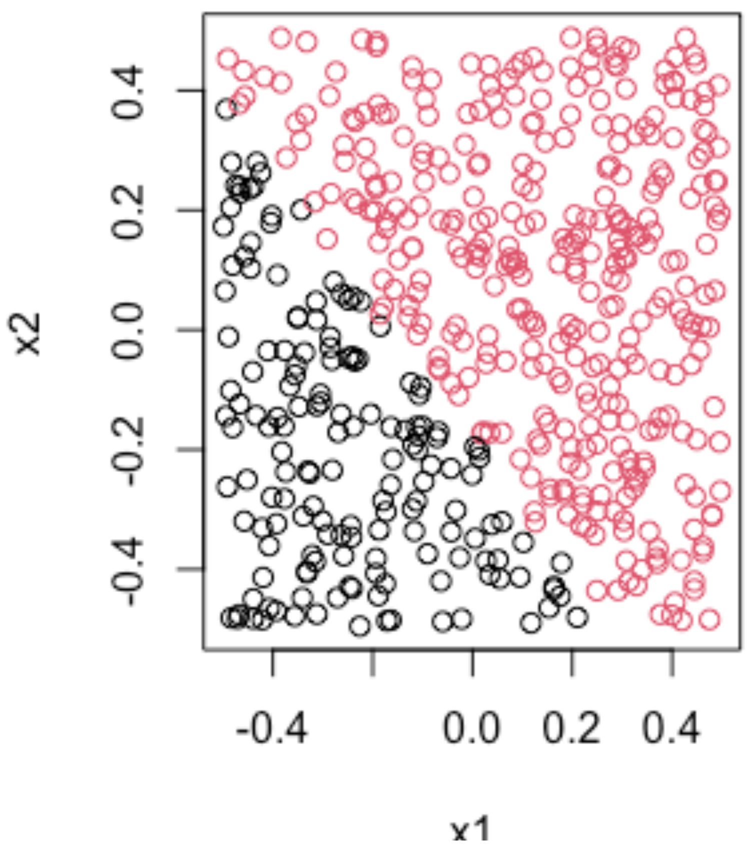
```
rm(list=ls())                       # Clear environment
oldpar <- par()                     # save default graphical
parameters
if (!is.null(dev.list()["RStudioGD"])) # Clear plot window
  dev.off(dev.list()["RStudioGD"])
cat("\014")                         # Clear the Console


#a
set.seed(12345)
x1 <- runif(500) - 0.5
x2 <- runif(500) - 0.5
y <- 1 * (x1^2 - x2^2 > 0)
```

```
#b
plot(x1, x2, col = y + 1)
```

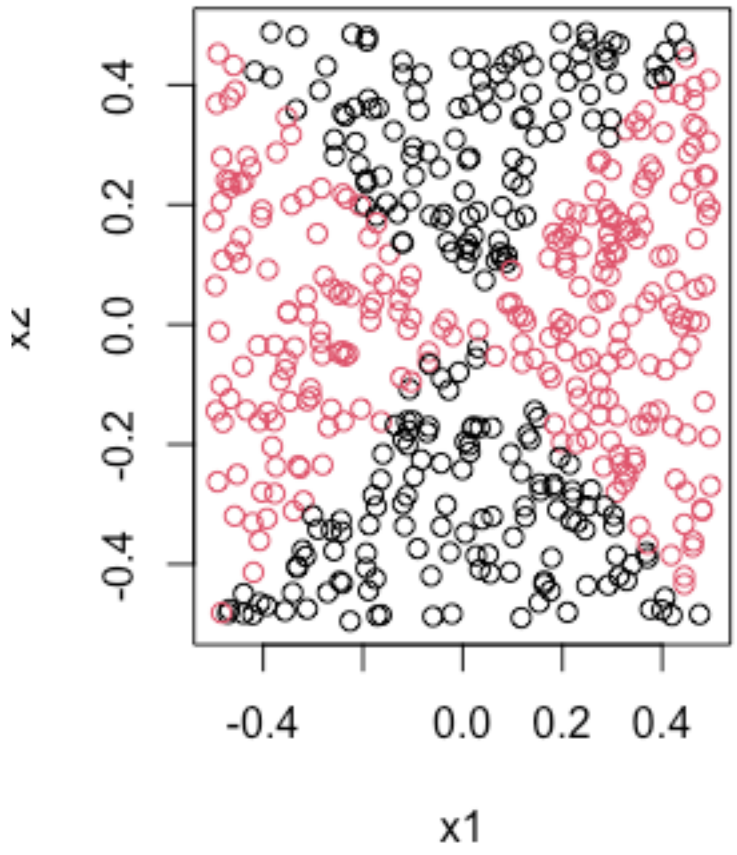


```
#c
#fits a logistic regression model
logit <- glm(as.factor(y) ~ x1 + x2, family = "binomial")

#d
# Make predictions using the logistic regression model
y_pred <- predict(logit, type = "response")

# Convert predicted probabilities to class labels
y_pred_class <- as.numeric(y_pred > 0.5)
```

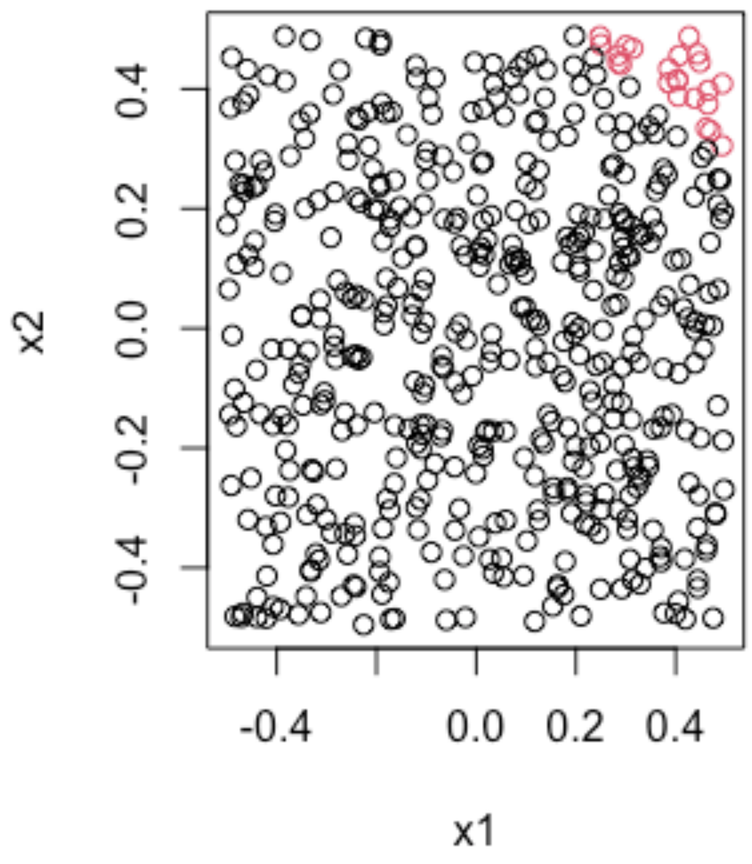
```
plot(x1, x2, col =
 y_pred_class + 1)
```



```
#e
# Create new variables based on x1 and x2
x3 <- x1^2
x4 <- x2^2
x5 <- x1*x2

# Fit a logistic regression model with non-linear terms
logit_nl <- glm(y ~ x1 + x2 + x3 + x4 + x5, family = "binomial")

#f
y_pred_nl <- predict(logit_nl, type = "response")
y_pred_class_nl <- as.numeric(y_pred_nl > 0.5)
plot(x1, x2, col =
y_pred_class_nl + 1)
```

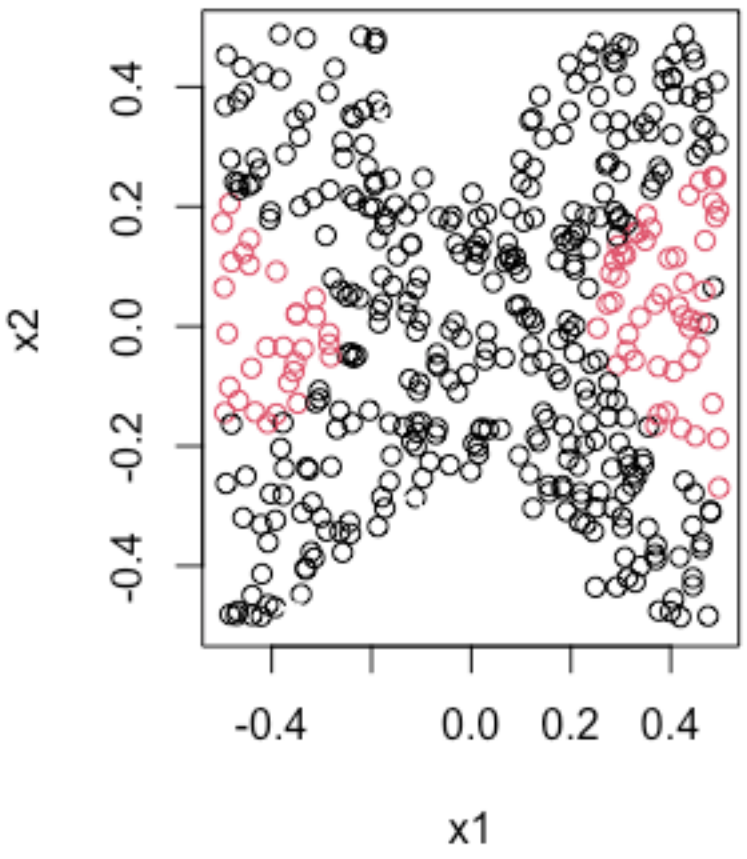


```
#g

??e1071

library(e1071)

svm_linear <- svm(y ~ x1 + x2, data = data.frame(x1, x2, y), kernel =
"linear")


y_pred_svm_linear <- predict(svm_linear)

plot(x1, x2, col = y_pred_svm_linear + 1)
```



Jim

```
#h

svm_nonlinear <- svm(y ~ x1 + x2, data = data.frame(x1, x2, y), kernel
= "radial")

y_pred_svm_nonlinear <- predict(svm_nonlinear)
plot(x1, x2, col = y_pred_svm_nonlinear + 1)
```



(i)

The logistic regression model with a nonlinear transformation of the predictor variables produces a nonlinear decision boundary. Support vector classifiers with linear kernels also produce linear decision boundaries, but support vector machines with nonlinear kernels are able to capture nonlinear patterns in the data and provide better classification boundaries.

Task 2:. For the following tasks continue working with the **`credit.csv`** data set to predict the default probabilities. .[5 points]

[a] Split the data into a stratified training data set with 70% of the observations and a test data set with the remaining 30% of the observations.

```
# Load the required packages

library(caret)

library(dplyr)

library(e1071)


# Read the data

credit <- read.csv("/Users/jimpan/Documents/EPPS 6326/week files/
Weeks06and07/credit.csv")


# Split the data into training (70%) and test (30%) sets

set.seed(123)

trainIndex <- createDataPartition(credit$default, p = 0.7, list =
FALSE, times = 1)

train <- credit[trainIndex, ]

test <- credit[-trainIndex, ]
```

[b] Use a radial kernel support vector classifier. Identify with cross-evaluation the “optimal” cost parameter.

```
# Set up the tuning grid

tuneGrid <- expand.grid(.sigma = c(0.01, 0.1, 1, 5, 10), .C = c(0.1,
1, 5, 10, 100))
```

```
# Set up the train control
control <- trainControl(method = "repeatedcv", number = 10, repeats =
3, classProbs = TRUE, summaryFunction = twoClassSummary)

# Fit the SVM model with radial kernel using cross-validation
svmFit <- train(default ~ ., data = train, method = "svmRadial",
tuneGrid = tuneGrid, trControl = control, preProcess = c("center",
"scale"), metric = "ROC")

# Print the optimal cost parameter
svmFit$bestTune
#  sigma C
#2  0.01 1
```

[c] Evaluate your optimal model with the confusion matrix for the test dataset and the ROC curve including the AUC.

```
library(pROC)

# Predict on the test data using the optimal model
svmPred <- predict(svmFit, newdata = test)

# Convert predicted class labels to factor with levels "No" and "Yes"
svmPred <- factor(svmPred, levels = c("No", "Yes"))

# Convert true class labels to factor with levels "No" and "Yes"
test$default <- factor(test$default, levels = c("No", "Yes"))

svmPred <- factor(ifelse(svmPred == "No", "No", "Yes"), levels =
c("No", "Yes"))
```

```
# Confusion matrix
confusionMatrix(data = svmPred, reference = test$default)

# ROC curve and AUC
rocObj <- roc(test$default, as.numeric(svmPred))
plot(rocObj)
```