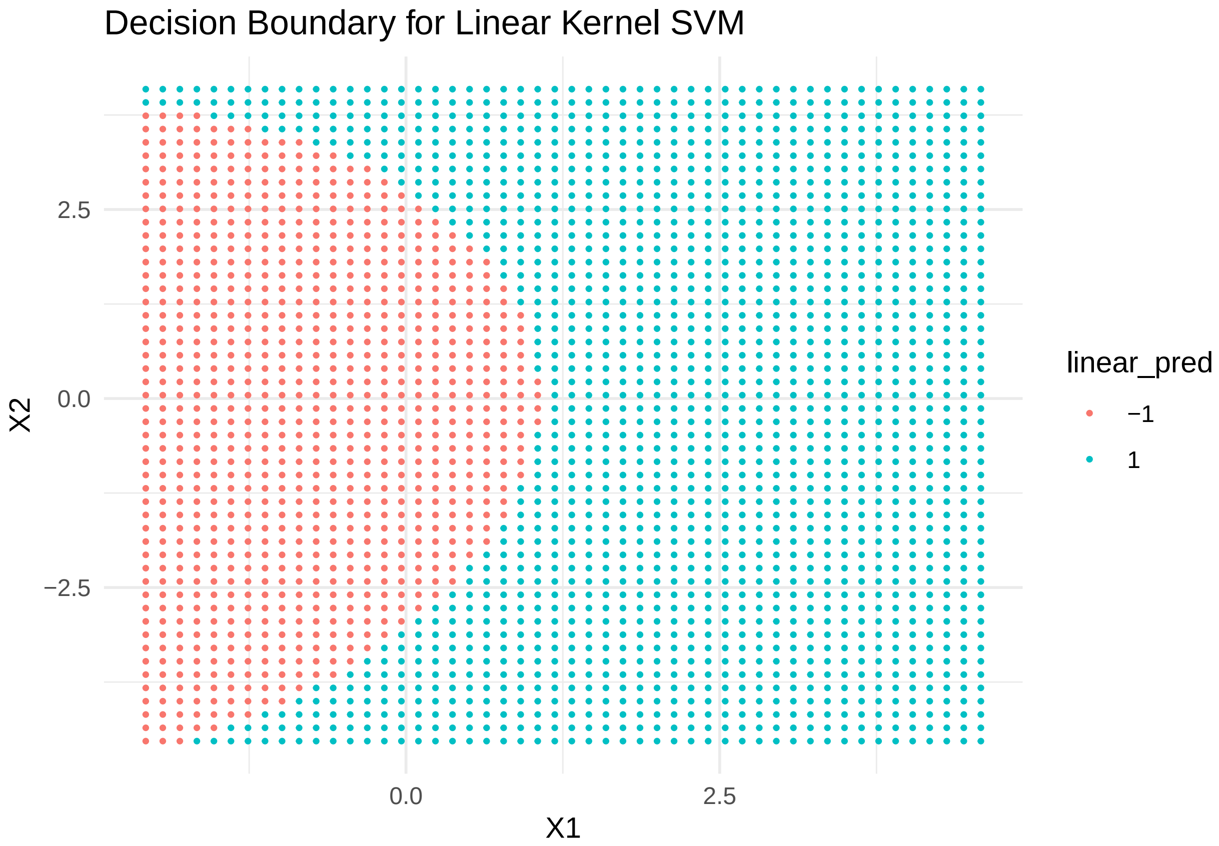
Homework 6

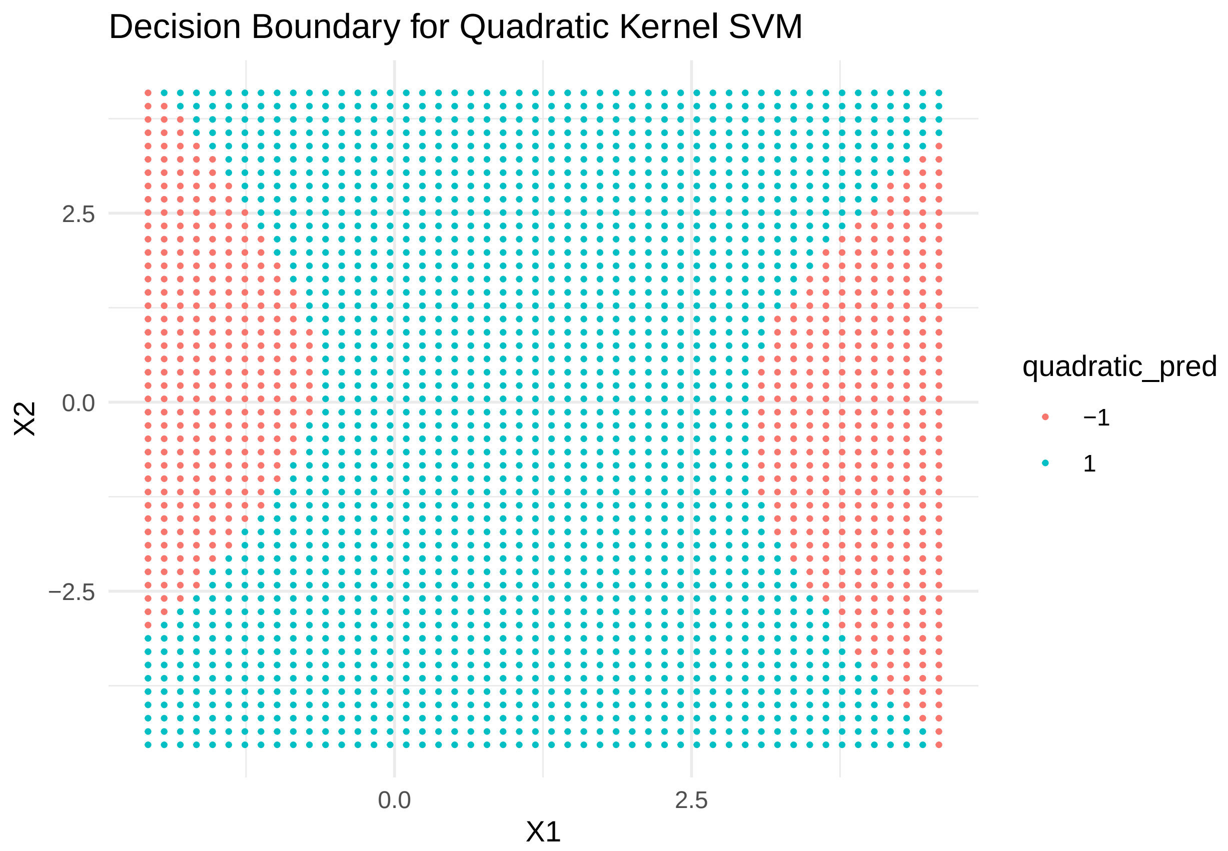
Alex Ojemann

2023-11-02

Problem 1

As shown below, the two kernels do not result in the same decision boundary.





Problem 2

a.

The probability that x1* is not equal to x1 is calculated as the complement of the probability that all n elements in the bootstrap sample are equal to x1.

The probability that a specific sample in the bootstrap is equal to x1 is 1/n because each of the n elements has an equal probability of being selected.

Therefore, the probability that all n samples are equal to x1 is (1/n)ˆn.

Hence, the probability that x1* is not equal to x1 is 1 - (1/n)ˆn.

b.

The probability that x2* is not equal to x1 is calculated as the complement of the probability that the second bootstrap sample is the same as the original sample x1.

Similar to part (a), the probability that a specific sample in the bootstrap is equal to x1 is 1/n because each of the n elements has an equal probability of being selected.

Therefore, the probability that x2* is equal to x1 is (1/n) not 1 - (1/n).

c.

The probability that x1 is not in the bootstrapped dataset is calculated as the complement of the probability that all n elements in the bootstrap sample are equal to x1.

Similar to part (a), the probability that a specific sample in the bootstrap is equal to x1 is 1/n because each of the n elements has an equal probability of being selected.

Therefore, the probability that all n samples are equal to x1 is (1/n)^n.

Hence, the probability that x1 is not in the bootstrapped dataset is (1 - (1/n)$^{n)}$n.

d.

To show that the probability that x1 is not in the bootstrapped dataset converges to e^(-1) as n approaches infinity, we utilize the result from part (c).

The probability that x1 is not in the bootstrapped dataset is given by (1 - (1/n)$^{n)}$n.

As n approaches infinity, (1 - (1/n)^n) approaches e^(-1) according to the limit definition.

Therefore, the probability that x1 is not in the bootstrapped dataset converges to e^(-1) as n tends to infinity.

Appendix

```r
load('HW6.RData')

library(dplyr)


new_dat <- dat %>%
  transmute(sqrt2_X1 = sqrt(2) * X1,
         sqrt2_X2 = sqrt(2) * X2,
         sqrt2_X1X2 = sqrt(2) * X1 * X2,
         X1_squared = X1^2,
         X2_squared = X2^2)

# Load the required libraries
library(kernlab)
library(ggplot2)

# Assuming you have a data frame 'dat' with features X1, X2, and y

# Create a linear kernel SVM
linear_svm <- ksvm(as.matrix(new_dat[, c("sqrt2_X1", "sqrt2_X2","sqrt2_X1X2","X1_squared","X2_squared")]

# Load the required library
library(e1071)

# Assuming 'dat' is your dataset with features X1, X2, and y

# Train the quadratic kernel SVM
quadratic_svm <- svm(y ~ ., data = dat, type = "C-classification", kernel = "polynomial", degree = 2)

# Print the model summary
print(quadratic_svm)
```

```r
# Create a grid of points for plotting decision boundaries
grid <- expand.grid(X1 = seq(min(dat$X1), max(dat$X1), length.out = 50), X2 = seq(min(dat$X2), max(dat$

# Load the required library
library(data.table)

# Create a data table for the grid
grid_data <- data.table(X1 = grid$X1, X2 = grid$X2)

# Calculate the additional columns
grid_data[, `:=` (
  sqrt2_X1 = sqrt(2) * X1,
  sqrt2_X2 = sqrt(2) * X2,
  sqrt2_X1X2 = sqrt(2) * X1 * X2,
  X1_squared = X1 * X1,
  X2_squared = X2 * X2
)]

grid_data[, c("X1", "X2") := NULL]

# Predict on the 5D matrix grid points
grid$linear_pred <- predict(linear_svm, newdata = grid_data)

data2 = as.data.frame(grid)
data2 <- subset(data2, select = -c(linear_pred))


grid$quadratic_pred <- predict(quadratic_svm, newdata = data2)


# Plot the decision boundary for the linear kernel SVM using the original 2D grid points
ggplot(grid, aes(x = X1, y = X2, color = linear_pred)) +
  geom_point(size = 0.5) +
  ggtitle("Decision Boundary for Linear Kernel SVM") +
  theme_minimal()


ggplot(grid, aes(x = X1, y = X2, color = quadratic_pred)) +
  geom_point(size = 0.5) +
  ggtitle("Decision Boundary for Quadratic Kernel SVM") +
  theme_minimal()
```