32-SVM-RMarkdown

Le Nhat Tung

Contents

1 Giới thiệu về SVM

Support Vector Machine (SVM) là một trong những thuật toán học máy có giám sát (supervised learning) phổ biến nhất. SVM được sử dụng rộng rãi trong các bài toán phân loại (classification) và hồi quy (regression).

1.1 Ý tưởng chính của SVM là gì?

Hãy tưởng tượng bạn có một tập dữ liệu gồm nhiều điểm thuộc hai lớp khác nhau (ví dụ: hoa màu đỏ và hoa màu xanh). SVM cố gắng tìm một đường thẳng (trong không gian 2D) hoặc một siêu phẳng (hyperplane) (trong không gian nhiều chiều) tốt nhất để phân tách hai lớp dữ liệu này.

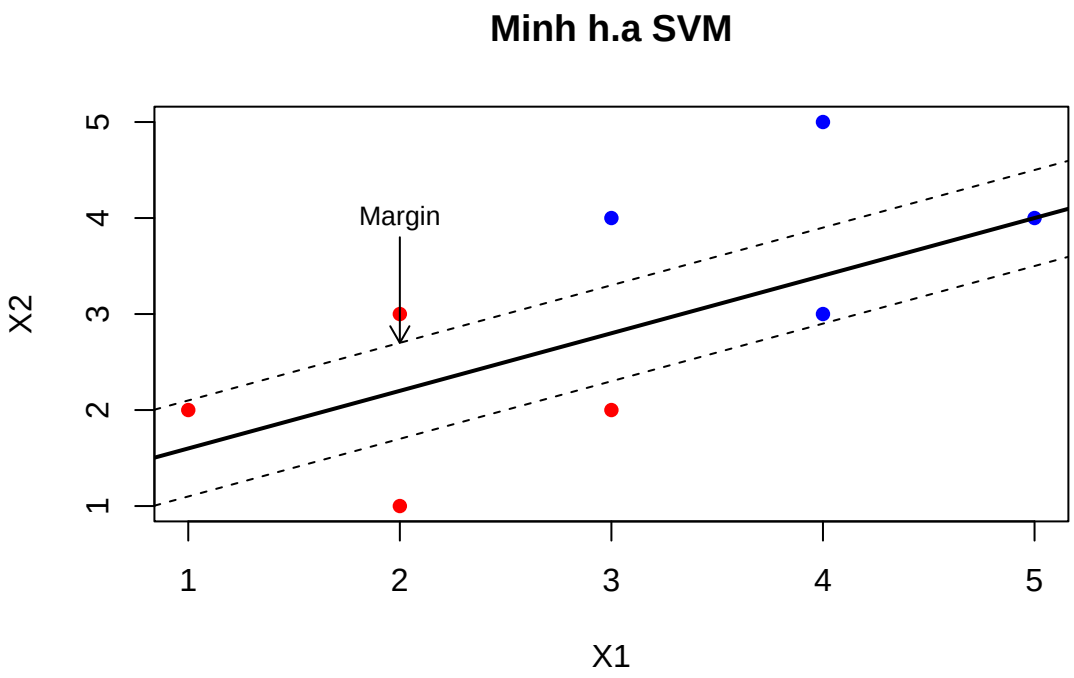


Figure 1: Mô tả SVM

Đường thẳng "tốt nhất" là đường có **lề (margin) lớn nhất** - tức là khoảng cách từ đường đến các điểm dữ liệu gần nhất của mỗi lớp là lớn nhất.

1.1.1 Kernel Trick

Khi dữ liệu không thể phân tách tuyến tính trong không gian ban đầu, SVM sử dụng **kernel trick** để ánh xạ dữ liệu sang không gian có số chiều cao hơn, nơi mà dữ liệu có thể được phân tách tuyến tính.

Một số kernel phổ biến:

- **Linear**: $K(x_i, x_j) = x_i^T x_j$
- **Polynomial**: $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d$
- **RBF (Radial Basis Function)**: $K(x_i, x_j) = exp(-\gamma ||x_i - x_j||^2)$
- **Sigmoid**: $K(x_i, x_j) = tanh(\gamma x_i^T x_j + r)$

2 Mô hình SVM và Ứng dụng với Bộ Dữ liệu Iris

2.1 Cài đặt và nạp các gói cần thiết

```
library(e1071)  # Gói chứa hàm svm
library(caret)  # Dùng cho đánh giá mô hình
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(ggplot2)  # Dùng cho trực quan hóa
```

2.2 Tien xu ly du lieu

2.2.1 Nap du lieu Iris va kiem tra cau truc

```
# Nạp dữ liệu Iris
data(iris)
```

```
# Xem cấu trúc dữ liệu
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
# Xem thống kê mô tả dữ liệu
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

2.2.2 Trực quan hóa dữ liệu

```
# Vẽ biểu đồ phân tán theo cặp đặc trưng Petal.Length và Petal.Width
ggplot(iris, aes(x = Petal.Length, y = Petal.Width, color = Species)) +
  geom_point(size = 3, alpha = 0.8) +
  labs(title = "Phân bố các loài hoa Iris",
       x = "Chiều dài cánh hoa (cm)",
       y = "Chiều rộng cánh hoa (cm)") +
  theme_minimal() +
  theme(legend.position = "bottom",
        plot.title = element_text(hjust = 0.5, face = "bold"),
        text = element_text(size = 12))
```

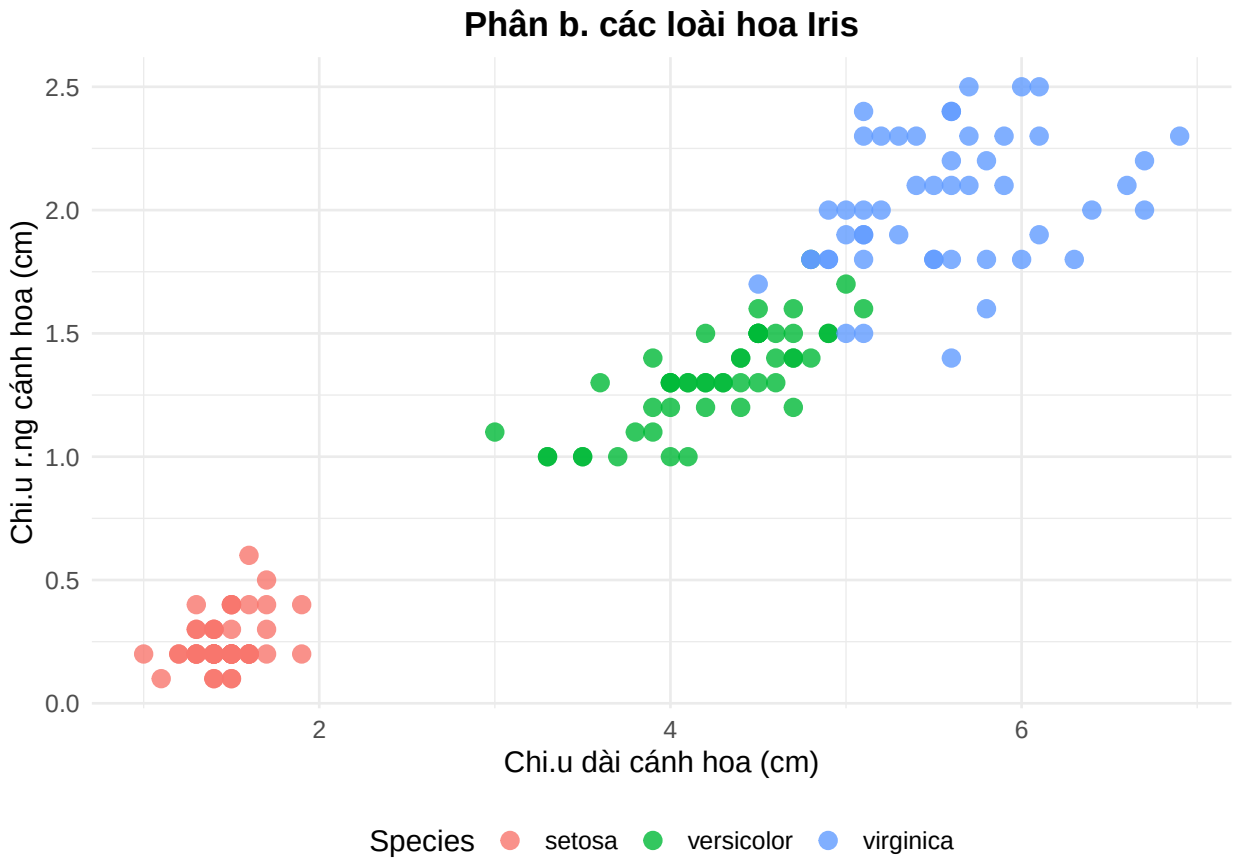


Figure 2: Phân bố các loài hoa Iris dựa trên chiều dài và chiều rộng cánh hoa

```
# Ma trận phân tán (pairs plot)
pairs(iris[1:4],
      main = "Ma trận phân tán dữ liệu Iris",
      pch = 21,
      bg = c("red", "green3", "blue")[unclass(iris$Species)])
legend("bottom",
       legend = levels(iris$Species),
       fill = c("red", "green3", "blue"),
```

```
        horiz = TRUE,
        cex = 0.8)
```

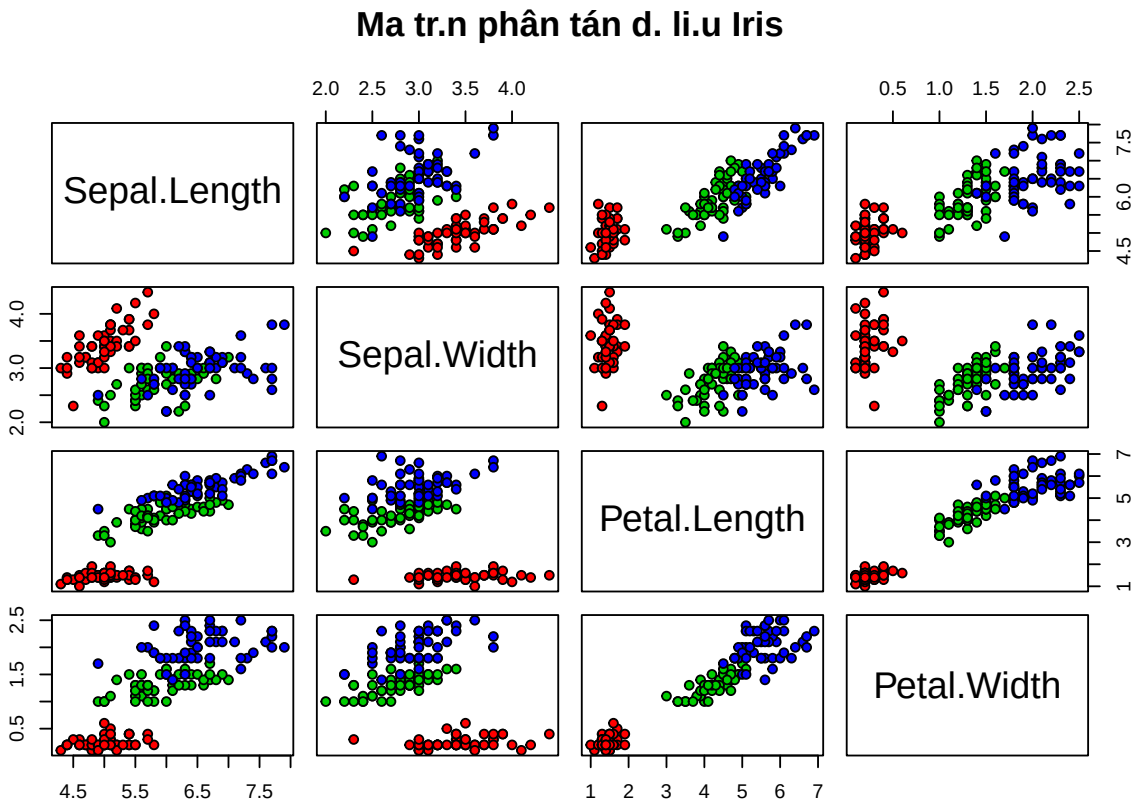


Figure 3: Ma trận phân tán tất cả các đặc trưng của bộ dữ liệu Iris

2.3 Chia dữ liệu thành tập huấn luyện và tập kiểm tra

```
# Đặt seed cho tính tái lập
set.seed(123)

# Tạo chỉ số phân chia
trainIndex <- createDataPartition(iris$Species, p = 0.7, list = FALSE)
trainData <- iris[trainIndex, ]
testData <- iris[-trainIndex, ]

# Kiểm tra kích thước các tập dữ liệu
cat("Kích thước tập huấn luyện:", dim(trainData), "\n")
```

```
## Kích thước tập huấn luyện: 105 5
```

```r
cat("Kích thước tập kiểm tra:", dim(testData), "\n")
```

```
## Kích thước tập kiểm tra: 45 5
```

2.4 Xây dựng mô hình SVM

2.4.1 Mô hình SVM với kernel tuyến tính

2.4.2 Mô hình SVM với kernel tuyến tính

```r
# Xây dựng mô hình SVM với kernel tuyến tính
svm_linear <- svm(Species ~ .,
                  data = trainData,
                  type = "C-classification",
                  kernel = "linear",
                  cost = 1)

# Tóm tắt mô hình
summary(svm_linear)
```

```
##
## Call:
## svm(formula = Species ~ ., data = trainData, type = "C-classification",
##     kernel = "linear", cost = 1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  22
##
##  ( 2 10 10 )
##
##
## Number of Classes:  3
##
## Levels:
##  setosa versicolor virginica
```

2.4.3 Mô hình SVM với kernel RBF

```r
# Xây dựng mô hình SVM với kernel RBF (Radial Basis Function)
svm_radial <- svm(Species ~ .,
                  data = trainData,
                  type = "C-classification",
                  kernel = "radial",
```

```
                cost = 1,
                gamma = 0.5)

# Tóm tắt mô hình
summary(svm_radial)
```

```
##
## Call:
## svm(formula = Species ~ ., data = trainData, type = "C-classification",
##     kernel = "radial", cost = 1, gamma = 0.5)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  45
##
##  ( 10 15 20 )
##
##
## Number of Classes:  3
##
## Levels:
##  setosa versicolor virginica
```

2.5 Đánh giá mô hình

2.5.1 Dự đoán và ma trận nhầm lẫn

```
# Dự đoán với mô hình tuyến tính
pred_linear <- predict(svm_linear, testData)
# Ma trận nhầm lẫn
confusionMatrix(pred_linear, testData$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         15          0         0
##   versicolor      0         15         1
##   virginica       0          0        14
##
## Overall Statistics
##
##                Accuracy : 0.9778
##                  95% CI : (0.8823, 0.9994)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : < 2.2e-16
##
```

```
##                   Kappa : 0.9667
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            1.0000           0.9333
## Specificity                 1.0000            0.9667           1.0000
## Pos Pred Value              1.0000            0.9375           1.0000
## Neg Pred Value              1.0000            1.0000           0.9677
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3333            0.3333           0.3111
## Detection Prevalence        0.3333            0.3556           0.3111
## Balanced Accuracy           1.0000            0.9833           0.9667
```

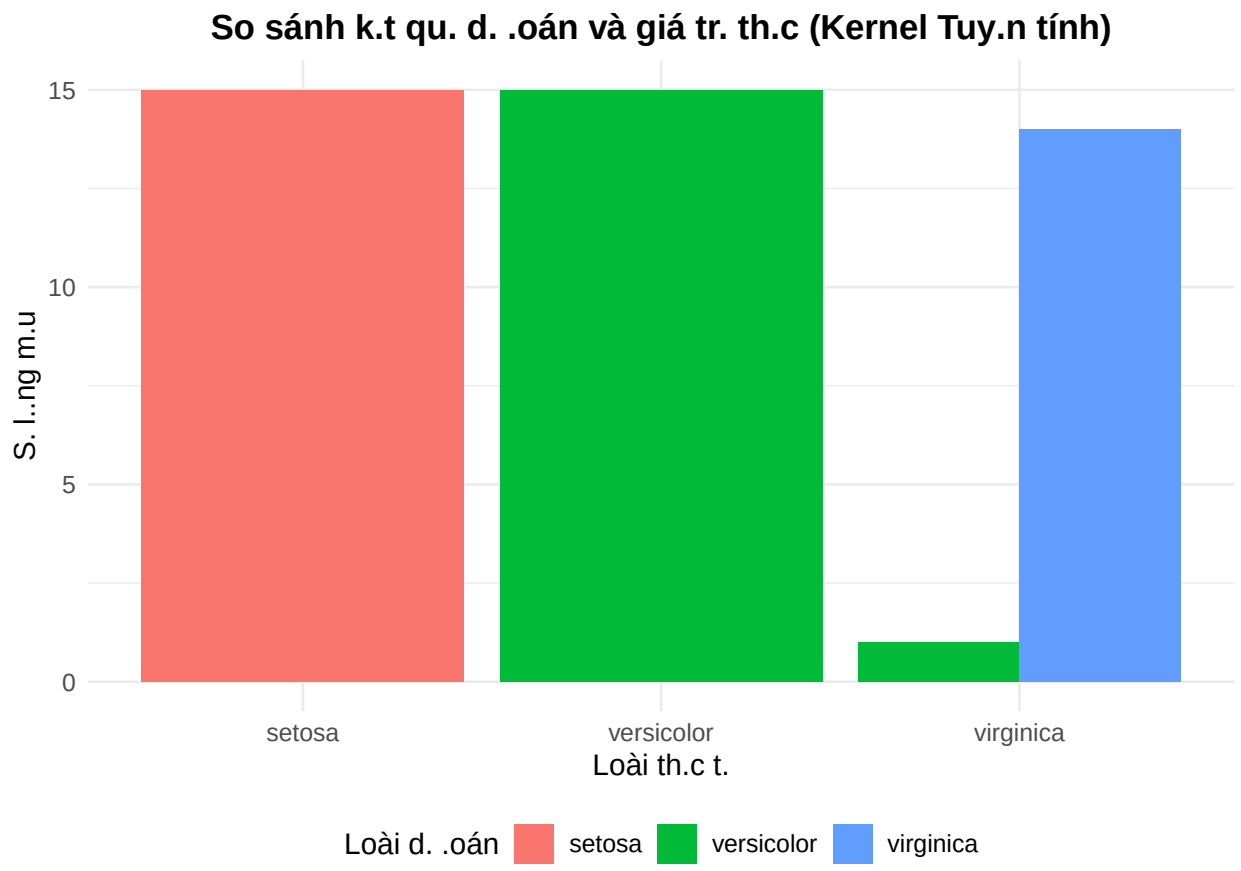
```r
# Dự đoán với mô hình RBF
pred_radial <- predict(svm_radial, testData)
# Ma trận nhầm lẫn
confusionMatrix(pred_radial, testData$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         15          0         0
##   versicolor      0         14         2
##   virginica       0          1        13
##
## Overall Statistics
##
##                Accuracy : 0.9333
##                  95% CI : (0.8173, 0.986)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.9333           0.8667
## Specificity                 1.0000            0.9333           0.9667
## Pos Pred Value              1.0000            0.8750           0.9286
## Neg Pred Value              1.0000            0.9655           0.9355
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3333            0.3111           0.2889
## Detection Prevalence        0.3333            0.3556           0.3111
## Balanced Accuracy           1.0000            0.9333           0.9167
```

2.5.2 Trực quan hóa kết quả phân loại

```r
# Tạo dữ liệu cho biểu đồ
results_df <- data.frame(
  Actual = testData$Species,
  Linear_Pred = pred_linear,
  Radial_Pred = pred_radial
)

# Trực quan hóa kết quả dự đoán của mô hình tuyến tính
ggplot(results_df, aes(x = Actual, fill = Linear_Pred)) +
  geom_bar(position = "dodge") +
  labs(title = "So sánh kết quả dự đoán và giá trị thực (Kernel Tuyến tính)",
       x = "Loài thực tế",
       y = "Số lượng mẫu",
       fill = "Loài dự đoán") +
  theme_minimal() +
  theme(legend.position = "bottom",
        plot.title = element_text(hjust = 0.5, face = "bold"),
        text = element_text(size = 12))
```



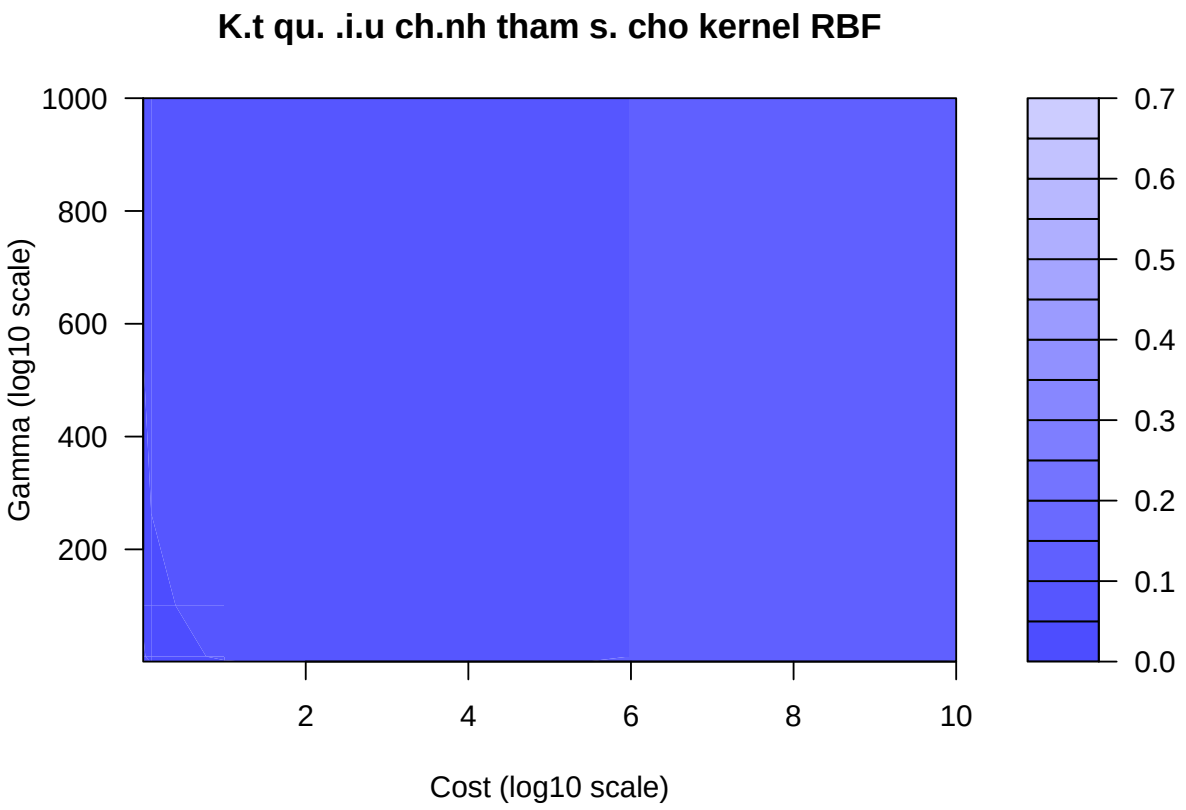
2.6 Điều chỉnh tham số mô hình (Hyperparameter Tuning)

```r
# Tìm tham số tối ưu cho mô hình SVM với kernel RBF
tune_result <- tune.svm(Species ~ .,
                        data = trainData,
                        kernel = "radial",
                        gamma = 10^(-3:1),
                        cost = 10^(0:3))

# Xem kết quả điều chỉnh
print(tune_result)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  gamma cost
##    0.1   10
##
## - best performance: 0.02818182
```

```r
# Trực quan hóa kết quả điều chỉnh tham số
plot(tune_result,
     main = "Kết quả điều chỉnh tham số cho kernel RBF",
     xlab = "Cost (log10 scale)",
     ylab = "Gamma (log10 scale)")
```




```r
# Xây dựng mô hình SVM với tham số tối ưu
svm_tuned <- svm(Species ~ .,
                 data = trainData,
                 type = "C-classification",
                 kernel = "radial",
                 cost = tune_result$best.parameters$cost,
                 gamma = tune_result$best.parameters$gamma)

# Đánh giá mô hình với tham số tối ưu
pred_tuned <- predict(svm_tuned, testData)
confusionMatrix(pred_tuned, testData$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         15          0         0
##   versicolor      0         14         2
##   virginica       0          1        13
##
## Overall Statistics
##
##                Accuracy : 0.9333
##                  95% CI : (0.8173, 0.986)
##     No Information Rate : 0.3333
```

```
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.9333           0.8667
## Specificity                 1.0000            0.9333           0.9667
## Pos Pred Value              1.0000            0.8750           0.9286
## Neg Pred Value              1.0000            0.9655           0.9355
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3333            0.3111           0.2889
## Detection Prevalence        0.3333            0.3556           0.3111
## Balanced Accuracy           1.0000            0.9333           0.9167
```

3 Ưu và nhược điểm của SVM

3.1 Ưu điểm

1. **Hiệu quả trong không gian nhiều chiều**: SVM xử lý tốt khi số lượng chiều lớn hơn số mẫu.
2. **Linh hoạt với các kernel khác nhau**: Có thể áp dụng nhiều kernel khác nhau cho các bài toán khác nhau.
3. **Khả năng tổng quát hóa tốt**: Mô hình SVM thường có khả năng tổng quát hóa tốt để phân loại các mẫu mới.
4. **Mạnh mẽ với dữ liệu ngoại lai (outliers)**: Chỉ sử dụng các support vectors để xác định siêu phẳng quyết định.

3.2 Nhược điểm

1. **Khó khăn với bộ dữ liệu lớn**: SVM không hoạt động hiệu quả với bộ dữ liệu lớn (có nhiều mẫu).
2. **Độ phức tạp về mặt tính toán**: Việc huấn luyện có thể tốn nhiều thời gian cho dữ liệu lớn.
3. **Khó xác định tham số tối ưu**: Việc chọn kernel và các tham số phù hợp có thể phức tạp.
4. **Khó giải thích kết quả**: Các kết quả từ SVM khó giải thích hơn so với các thuật toán đơn giản khác.

4 Tổng kết và kết luận

SVM là một thuật toán mạnh mẽ cho các bài toán phân loại và hồi quy, đặc biệt hiệu quả khi làm việc với dữ liệu có số chiều cao. Thông qua ví dụ với bộ dữ liệu Iris, chúng ta đã thấy SVM có thể đạt được độ chính xác cao trong phân loại các loài hoa.

Trong bài thực hành này, chúng ta đã: - Tìm hiểu về nguyên lý cơ bản của SVM - Áp dụng SVM với các kernel khác nhau (tuyến tính và RBF) - Đánh giá hiệu suất của các mô hình - Điều chỉnh tham số để cải thiện mô hình

Việc lựa chọn kernel phù hợp và điều chỉnh tham số đóng vai trò quan trọng trong việc xây dựng mô hình SVM hiệu quả. Tùy vào đặc điểm của bộ dữ liệu và yêu cầu của bài toán, chúng ta có thể lựa chọn các thiết lập khác nhau cho SVM.

5 Tài liệu tham khảo

1. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273-297.
2. Meyer, D., & Wien, F. H. T. (2015). Support vector machines. *The Interface to libsvm in package e1071*.
3. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning*. Springer.
4. Karatzoglou, A., Meyer, D., & Hornik, K. (2006). Support Vector Machines in R. *Journal of Statistical Software*, 15(9), 1-28.