R Languages

GVGD: VÕ THỊ HỒNG TUYẾT

Nội dung

1. Giới thiệu ngôn ngữ R
2. Tại sao dùng R
3. R và Google Colaboratory
4. R basic
5. OOP in R
6. R data structures
7. R graphics
8. R statistics
9. Data manipulation in R
10. Machine learning in R

Data manipulation in R

- dplyr
- tidyr
- data.table
- Xử lý chuỗi ký tự
- Xử lý thời gian
- Xử lý dữ liệu thiếu

Data manipulation in R

- dplyr: gói thao tác dữ liệu
  - filter(): chọn các dòng theo điều kiện

```
library(dplyr)
data <- data.frame(x = 1:5, y = c("A", "B", "A", "B", "A"))
filtered_data <- filter(data, x > 2)
filtered_data
```

A data.frame: 3 × 2

x	y
<int>	<chr>
3	A
4	B
5	A

Data manipulation in R

- dplyr: gói thao tác dữ liệu
  - select(): chọn các cột

```
selected_data <- select(data, y)
selected_data
```

A data.frame: 5 × 1

y
<chr>
A
B
A
B
A

Data manipulation in R

- dplyr: gói thao tác dữ liệu
  - mutate(): thêm hoặc biến đổi cột

```
mutated_data <- mutate(data, n = x * 2)
mutated_data
```

A data.frame: 5 × 3

x	y	n
<int>	<chr>	<dbl>
1	A	2
2	B	4
3	A	6
4	B	8
5	A	10

Data manipulation in R

- dplyr: gói thao tác dữ liệu
  - summarise(): tổng hợp dữ liệu

```
summarised_data <- summarise(data, mean_x = mean(x))
summarised_data
```

A data.frame: 1 × 1

mean_x
<dbl>
3

Data manipulation in R

- tidyr: sắp xếp dữ liệu
  - gather(): biến cột thành hàng

```
library(tidyr)
wide_data <- data.frame(id = 1:2, score1 = c(10, 20), score2 = c(30, 40))
wide_data
long_data <- gather(wide_data, key = "score_type", value = "score", score1:score2)
long_data
```

A data.frame: 2 × 3

id <int>	score1 <dbl>	score2 <dbl>
1	10	30
2	20	40

A data.frame: 4 × 3

id <int>	score_type <chr>	score <dbl>
1	score1	10
2	score1	20
1	score2	30
2	score2	40

Data manipulation in R

- tidyr: sắp xếp dữ liệu
  - spread(): biến hàng thành cột

```
spread_data <- spread(long_data, key = "score_type", value = "score")
spread_data
```

A data.frame: 2 × 3

id <int>	score1 <dbl>	score2 <dbl>
1	10	30
2	20	40

Data manipulation in R

- tidyr: sắp xếp dữ liệu
  - unite(): gộp nhiều cột thành 1 cột

```
[12] united_data <- unite(wide_data, "combined", score1, score2, sep = "--")
     united_data
```

A data.frame: 2 × 2

id <int>	combined <chr>
1	10--30
2	20--40

Data manipulation in R

- tidyr: sắp xếp dữ liệu
  - separate(): tách 1 cột thành nhiều cột

```
separated_data <- separate(united_data, combined, into = c("score1", "score2"), sep = "--")
separated_data
```

A data.frame: 2 × 3

id <int>	score1 <chr>	score2 <chr>
1	10	30
2	20	40

Data manipulation in R

- tidyr: sắp xếp dữ liệu
  - arrange(): sắp xếp dữ liệu theo thứ tự

```
data <- data.frame(x = 1:5, y = c("A", "B", "A", "B", "A"))
arranged_data <- arrange(data, desc(x))
arranged_data
```

A data.frame: 5 × 2

x <int>	y <chr>
5	A
4	B
3	A
2	B
1	A

Data manipulation in R

- data.table: cung cấp các thao tác lọc, tổng hợp và thao tác dữ liệu

```
library(data.table)
dt <- data.table(x = 1:5, y = c("A", "B", "A", "B", "A"))
filtered_dt <- dt[x > 2 & x < 5]
filtered_dt
```

A data.table: 2 × 2

x <int>	y <chr>
3	A
4	B

Data manipulation in R

- Xử lý chuỗi ký tự:
  - grep(): tìm giá trị phù hợp với 1 mẫu

```
string_vec <- c("apple", "banana", "pear")
matches <- grep("a", string_vec)
matches
matches_ <- grep("p", string_vec)
matches_
```

```
1 2 3
1 3
```

Data manipulation in R

- Xử lý chuỗi ký tự:
  - sub() hoặc gsub(): tìm và thay thế chuỗi

```
sub_example <- sub("a", "o", "apple")
sub_example

gsub_example <- gsub("a", "o", "banana")
gsub_example
```

```
'opple'
'bonono'
```

Data manipulation in R

- Xử lý thời gian: thư viện lubridate
  - ymd(string_date): tạo chuỗi thời gian
  - year/month/day(string_date): cập nhật năm/tháng/ngày của đối tượng thời gian

```
library(lubridate)
date <- ymd("2025-03-25")
date
year(date) <- 2026
date
month(date) <-04
date
day(date) <- 30
date
```

```
2025-03-25
2026-03-25
2026-04-25
2026-04-30
```

Data manipulation in R

- Xử lý dữ liệu thiếu:
  - is.na(): xác định giá trị thiếu

```
na_vec <- c(1, NA, 3)
na_positions <- is.na(na_vec)
na_positions
```

```
FALSE · TRUE · FALSE
```

Data manipulation in R

- Xử lý dữ liệu thiếu:
  - na.omit(): loại bỏ các hàng

```r
# Tạo một dataframe với các giá trị thiếu
data <- data.frame(
  Name = c("Alice", "Bob", "Charlie", "David"),
  Score = c(85, NA, 95, NA)
)

# Hiển thị dữ liệu gốc
print("Dữ liệu gốc:")
print(data)

# Sử dụng na.omit() để loại bỏ các hàng có giá trị thiếu
clean_data <- na.omit(data)

# Hiển thị dữ liệu sau khi loại bỏ các hàng chứa NA
print("Dữ liệu sau khi loại bỏ hàng chứa NA:")
print(clean_data)
```

```
[1] "Dữ liệu gốc:"
     Name Score
1   Alice    85
2     Bob    NA
3 Charlie    95
4   David    NA
[1] "Dữ liệu sau khi loại bỏ hàng chứa NA:"
     Name Score
1   Alice    85
3 Charlie    95
```

Data manipulation in R

Xử lý dữ liệu thiếu:

- mice(): nằm trong thư viện mice và ước lượng các giá trị thiếu

```r
library(mice)

# Tạo một dataframe mẫu với giá trị thiếu
data <- data.frame(
  Name = c("Alice", "Bob", "Charlie", "David"),
  Age = c(25, NA, 30, NA),
  Score = c(95, 88, NA, 70)
)

# Hiển thị dữ liệu gốc
print("Dữ liệu gốc:")
print(data)

# Sử dụng hàm mice để ước lượng giá trị thiếu
imputed_data <- mice(data, m = 5, method = 'pmm', maxit = 5, seed = 500)

# Xem kết quả ước lượng
print("Tóm tắt ước lượng:")
summary(imputed_data)

# Trích xuất một trong các bộ dữ liệu đã được ước lượng
completed_data <- complete(imputed_data, 1)

# Hiển thị dữ liệu đã xử lý
print("Dữ liệu sau khi xử lý các giá trị thiếu:")
print(completed_data)
```

```
[1] "Dữ liệu gốc:"
     Name Age Score
1   Alice  25    95
2     Bob  NA    88
3 Charlie  30    NA
4   David  NA    70
```

```
[1] "Tóm tắt ước lượng:"
Class: mids
Number of multiple imputations:  5
Imputation methods:
 Name   Age Score
   "" "pmm" "pmm"
PredictorMatrix:
      Name Age Score
Name     0   1     1
Age      0   0     1
Score    0   1     0
Number of logged events:  17
  it im   dep     meth
1  0  0        constant
2  1  1   Age       pmm
3  1  2   Age       pmm
4  1  2 Score       pmm
5  1  3   Age       pmm
6  1  4   Age       pmm
```

```
[1] "Dữ liệu sau khi xử lý các giá trị thiếu:"
     Name Age Score
1   Alice  25    95
2     Bob  30    88
```

Machine learning in R

- Hồi quy tuyến tính (lm):

model <- lm(bien_phu_thuoc ~ bien_doc_lap, data)

predict(model, predict_value)

```
Hours <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
Score <- c(55, 60, 65, 70, 75, 80, 82, 85, 90, 95) + rnorm(10, mean = 0, sd = 5)  # Thêm nhiễu ngẫu nhiên

data <- data.frame(Hours, Score)

# Hiển thị dữ liệu
print("Dữ liệu mẫu:")
print(data)
```

```
[1] "Dữ liệu mẫu:"
   Hours     Score
1      1  49.66088
2      2  58.91013
3      3  59.86998
4      4  66.35554
5      5  71.87480
6      6  71.56653
7      7  86.18894
8      8  85.76687
9      9  84.30932
10    10 101.26907
```

Machine learning in R

- Hồi quy tuyến tính (lm):

model <- lm(bien_phu_thuoc ~ bien_doc_lap, data)

predict(model, predict_value)

```
model <- lm(Score ~ Hours, data = data)

# Hiển thị tóm tắt mô hình
summary(model)
```

```
Call:
lm(formula = Score ~ Hours, data = data)

Residuals:
   Min     1Q Median     3Q    Max
-6.894 -1.220 -0.034  2.423  5.058

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  45.8791     2.7689   16.57 1.78e-07 ***
Hours         5.0360     0.4462   11.29 3.42e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.053 on 8 degrees of freedom
Multiple R-squared:  0.9409,    Adjusted R-squared:  0.9335
F-statistic: 127.4 on 1 and 8 DF,  p-value: 3.418e-06
```

```
# Dự đoán điểm số dựa trên số giờ học mới
new_hours <- data.frame(Hours = c(2.5, 4.5, 8.5))
predictions <- predict(model, new_hours)

# Hiển thị dự đoán
print("Dự đoán điểm số cho số giờ học mới:")
print(predictions)
```

```
[1] "Dự đoán điểm số cho số giờ học mới:"
       1        2        3
58.46913 68.54118 88.68529
```

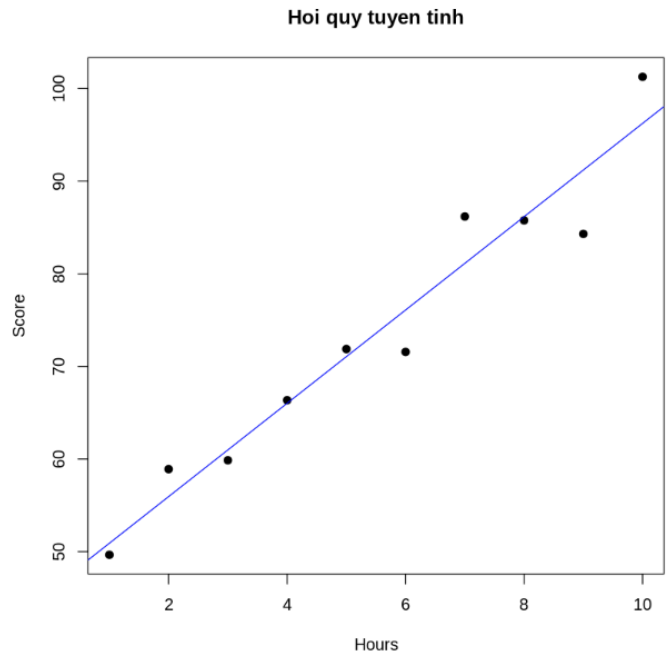
Machine learning in R

- Hồi quy tuyến tính (lm):

model <- lm(bien_phu_thuoc ~ bien_doc_lap, data)

predict(model, predict_value)

```
# Vẽ biểu đồ
plot(data$Hours, data$Score, main = "Hoi quy tuyen tinh", xlab = "Hours", ylab = "Score", pch = 19)
abline(model, col = "blue")  # Thêm đường hồi quy vào biểu đồ
```



Machine learning in R

Rừng ngẫu nhiên (randomforest)

model <- randomForest(bien_phu_thuoc ~ bien_doc_lap, data, ntree = so_luong_cay)

predict(model, test_data)

```
library(randomForest)
# Thông tin học sinh
age <- sample(15:20, 100, replace = TRUE)  # Tuổi ngẫu nhiên từ 15 đến 20
score <- round(runif(100, 50, 100))          # Điểm số ngẫu nhiên từ 50 đến 100
# Nhãn mục tiêu dựa trên điểm số
grade <- ifelse(score > 80, "Good",
                ifelse(score > 60, "Average", "Poor"))

# Tạo dataframe
student_data <- data.frame(Age = age, Score = score, Grade = as.factor(grade))

# Kiểm tra dữ liệu
print(head(student_data))
```

```
randomForest 4.7-1.2

Type rfNews() to see new features/changes/bug fixes.

  Age Score   Grade
1  19    70 Average
2  18    94    Good
3  19    68 Average
4  16    64 Average
5  15    59    Poor
6  15    59    Poor
```

Machine learning in R

Rừng ngẫu nhiên (randomforest)

model <- randomForest(bien_phu_thuoc ~ bien_doc_lap, data, ntree = so_luong_cay)

predict(model, test_data)

```
train_index <- sample(1:nrow(student_data), 0.7 * nrow(student_data))
train_data <- student_data[train_index, ]
test_data <- student_data[-train_index, ]

# Hiển thị số lượng dòng trong từng tập dữ liệu
cat("Số lượng dòng trong tập huấn luyện:", nrow(train_data), "\n")
cat("Số lượng dòng trong tập kiểm tra:", nrow(test_data), "\n")
```

```
Số lượng dòng trong tập huấn luyện: 70
Số lượng dòng trong tập kiểm tra: 30
```

Machine learning in R

Rừng ngẫu nhiên (randomforest)

model <- randomForest(bien_phu_thuoc ~ bien_doc_lap, data, ntree = so_luong_cay)

predict(model, test_data)

```
# Xây dựng mô hình Random Forest
rf_model <- randomForest(Grade ~ Age + Score, data = train_data, ntree = 100)

# Hiển thị thông tin mô hình
print(rf_model)
```

```
Call:
 randomForest(formula = Grade ~ Age + Score, data = train_data,      ntree = 100)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 1

        OOB estimate of  error rate: 2.86%
Confusion matrix:
        Average Good Poor class.error
Average      26    0    0  0.00000000
Good          1   27    0  0.03571429
Poor          1    0   15  0.06250000
```

Machine learning in R

Rừng ngẫu nhiên (randomforest)

model <- randomForest(bien_phu_thuoc ~ bien_doc_lap, data, ntree = so_luong_cay)

predict(model, test_data)

```
# Dự đoán trên tập kiểm tra
predictions <- predict(rf_model, test_data)

# Tạo ma trận nhầm lẫn
confusion_matrix <- table(test_data$Grade, predictions)
print("Ma trận nhầm lẫn:")
print(confusion_matrix)

# Tính toán độ chính xác
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Độ chính xác:", round(accuracy * 100, 2), "%\n")
```

```
[1] "Ma trận nhầm lẫn:"
         predictions
          Average Good Poor
  Average      11    0    0
  Good          0   12    0
  Poor          0    0    7
Độ chính xác: 100 %
```

Machine learning in R

Rừng ngẫu nhiên (randomforest)

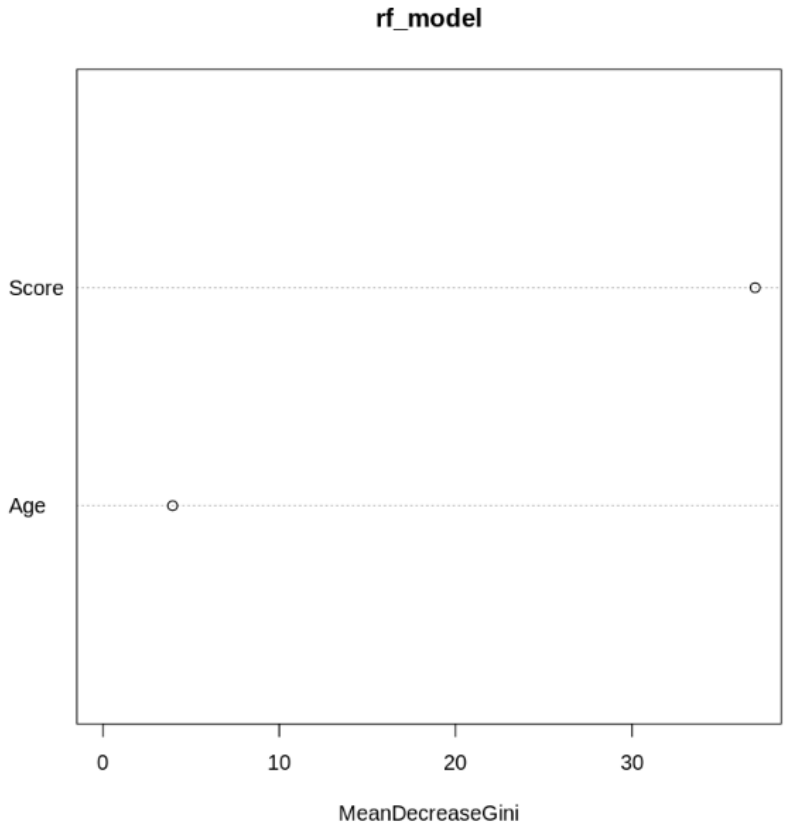
model <- randomForest(bien_phu_thuoc ~ bien_doc_lap, data, ntree = so_luong_cay)

predict(model, test_data)

```
# Hiển thị tầm quan trọng của các đặc trưng
importance_rf <- importance(rf_model)
print("Tầm quan trọng của các đặc trưng:")
print(importance_rf)

# Vẽ biểu đồ tầm quan trọng
varImpPlot(rf_model)
```

```
[1] "Tầm quan trọng của các đặc trưng:"
      MeanDecreaseGini
Age           3.956796
Score        36.983827
```



Machine learning in R

Học máy KNN (class):

predictions <- knn(train = train_features, test = test_features, cl = train_labels, k = k)

```
# Thông tin học sinh
age <- sample(15:20, 100, replace = TRUE)  # Tuổi ngẫu nhiên từ 15 đến 20
score <- round(runif(100, 50, 100))          # Điểm số ngẫu nhiên từ 50 đến 100
# Nhãn mục tiêu dựa trên điểm số
grade <- ifelse(score > 80, "Good",
                ifelse(score > 60, "Average", "Poor"))

# Tạo dataframe
student_data <- data.frame(Age = age, Score = score, Grade = as.factor(grade))

# Kiểm tra dữ liệu
print(head(student_data))
```

```
  Age Score   Grade
1  17    54    Poor
2  17    63 Average
3  17    68 Average
4  16    70 Average
5  17    72 Average
6  17    91    Good
```

Machine learning in R

- Học máy KNN (class):

predictions <- knn(train = train_features, test = test_features, cl = train_labels, k = k)

```
train_index <- sample(1:nrow(student_data), 0.7 * nrow(student_data))
train_data <- student_data[train_index, ]
test_data <- student_data[-train_index, ]

# Hiển thị số lượng dòng trong từng tập dữ liệu
cat("Số lượng dòng trong tập huấn luyện:", nrow(train_data), "\n")
cat("Số lượng dòng trong tập kiểm tra:", nrow(test_data), "\n")
```

```
Số lượng dòng trong tập huấn luyện: 70
Số lượng dòng trong tập kiểm tra: 30
```

Machine learning in R

- Học máy KNN (class):

predictions <- knn(train = train_features, test = test_features, cl = train_labels, k = k)

```
library(class)

# Chọn các đặc điểm để phân loại
train_features <- train_data[, c("Age", "Score")]
test_features <- test_data[, c("Age", "Score")]

# Chọn nhãn mục tiêu cho tập huấn luyện và kiểm tra
train_labels <- train_data$Grade
test_labels <- test_data$Grade

# Thực hiện phân loại KNN
k <- 3  # Số lượng láng giềng
predictions <- knn(train = train_features, test = test_features, cl = train_labels, k = k)

# Hiển thị dự đoán
print("Dự đoán lớp cho tập kiểm tra:")
print(predictions)
```

```
[1] "Dự đoán lớp cho tập kiểm tra:"
 [1] Poor    Good    Poor    Average Average Average Poor    Good    Good
[10] Average Average Good    Average Average Good    Good    Average Poor
[19] Good    Average Good    Average Good    Poor    Poor    Good    Average
[28] Poor    Poor    Good
Levels: Average Good Poor
```

Machine learning in R

- Học máy KNN (class):

predictions <- knn(train = train_features, test = test_features, cl = train_labels, k = k)

```
# Tạo ma trận nhầm lẫn
confusion_matrix <- table(test_labels, predictions)
print("Ma trận nhầm lẫn:")
print(confusion_matrix)

# Tính toán độ chính xác
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Độ chính xác:", round(accuracy * 100, 2), "%\n")
```

```
[1] "Ma trận nhầm lẫn:"
            predictions
test_labels Average Good Poor
    Average      11    0    0
    Good          0   11    0
    Poor          0    0    8
Độ chính xác: 100 %
```

Bài tập 1

- Tạo file csv chứa data như sau:

```r
# Tạo dataframe mẫu
student_data <- data.frame(
  Name = c("Alice", "Bob", "Charlie", "David", "Eva"),
  Age = c(20, 22, 21, 23, 19),
  GPA = c(3.5, 2.8, 3.9, 3.2, 4.0),
  Major = c("Math", "Physics", "Chemistry", "Math", "Biology")
)

# Lưu dataframe thành file CSV
write.csv(student_data, "student_data.csv", row.names = FALSE)
```

- Lọc những sinh viên có GPA lớn hơn 2.5
- Chọn hiển thị các cột `Name`, `Age` và `GPA`
- Thêm cột mới `Age_after_Graduation`, giả sử tính tuổi lúc tốt nghiệp là 21
- Sắp xếp dữ liệu theo GPA giảm dần
- Tính giá trị trung bình của GPA theo chuyên ngành

Bài tập 2

- **Tạo dataframe** với thông tin về giá nhà, diện tích, số phòng ngủ và số phòng tắm.

	Area	Bedrooms	Bathrooms	Price	Price_Grade
1	134	2	3	143146	Low
2	74	5	1	116617	Low
3	105	5	1	157862	Medium
4	75	3	2	98230	Low
5	244	4	2	261287	Medium
6	283	4	2	293128	Medium

- **Hồi quy tuyến tính**: Dự đoán giá nhà dựa trên diện tích và số phòng ngủ.
- **Random Forest**: Phân loại dựa trên giá nhà thành các nhóm (Thấp, Trung bình, Cao).
- **KNN**: Phân loại giá nhà với cùng tập dữ liệu và so sánh độ chính xác của các mô hình.