Giảm chiều dữ liệu sử dụng t-SNE

Le Nhat Tung

Contents

1 Giới thiệu về t-SNE

1.1 Khái niệm

t-Distributed Stochastic Neighbor Embedding (t-SNE) là một kỹ thuật giảm chiều phi tuyến tính được phát triển bởi Laurens van der Maaten và Geoffrey Hinton năm 2008. t-SNE đặc biệt hiệu quả trong việc giảm chiều dữ liệu đa chiều và trực quan hóa dữ liệu phức tạp với độ chính xác cao, bằng cách giữ cấu trúc cục bộ của dữ liệu.

1.2 Ưu điểm của t-SNE

t-SNE có các ưu điểm chính sau:

- **Bảo toàn cấu trúc cục bộ:** t-SNE bảo toàn tốt mối quan hệ giữa các điểm dữ liệu ở khoảng cách gần
- **Hiệu quả với dữ liệu phi tuyến:** Phù hợp với dữ liệu có cấu trúc phức tạp và phi tuyến tính
- **Trực quan hóa dữ liệu đa chiều:** Rất hiệu quả trong việc trực quan hóa dữ liệu nhiều chiều trên không gian 2D hoặc 3D
- **Phát hiện cụm (clusters):** Thường tạo ra các nhóm điểm dữ liệu tương đồng, hữu ích cho việc phân cụm

1.3 Hạn chế của t-SNE

- **Không bảo toàn khoảng cách toàn cục:** t-SNE có thể không thể hiện chính xác khoảng cách giữa các cụm xa nhau
- **Chi phí tính toán cao:** Tốn kém về mặt tính toán với các tập dữ liệu lớn
- **Kết quả phụ thuộc tham số:** Kết quả có thể thay đổi nhiều khi điều chỉnh hyperparameters
- **Không tạo ánh xạ tổng quát:** Không thể áp dụng trực tiếp mô hình t-SNE cho dữ liệu mới

1.4 Nguyên lý hoạt động của t-SNE

t-SNE hoạt động trên nguyên lý chính:

1. **Tính xác suất tương đồng** giữa các cặp điểm dữ liệu trong không gian đa chiều, dựa trên khoảng cách Euclidean và phân phối Gaussian. Xác suất càng cao khi hai điểm càng gần nhau.
2. **Tính xác suất tương đồng** giữa các cặp điểm trong không gian thấp chiều (thường là 2D hoặc 3D), sử dụng phân phối t với 1 bậc tự do (t-distribution). Phân phối t có "đuôi dày" hơn phân phối Gaussian, giúp tránh vấn đề "crowding problem".

3. **Tối thiểu hóa hàm mất mát** (Kullback-Leibler divergence) giữa hai phân phối xác suất trên. Điều này thúc đẩy điểm dữ liệu có xác suất tương đồng cao ở không gian đa chiều cũng sẽ có xác suất tương đồng cao ở không gian thấp chiều.

1.5 Tham số quan trọng của t-SNE

- **Perplexity:** Ảnh hưởng đến số lượng "láng giềng" được xem xét cho mỗi điểm dữ liệu (thường từ 5 đến 50)
- **Số lần lặp (iterations):** Số bước tối ưu hóa (thường từ 250 đến 1000)
- **Learning rate:** Tốc độ học ảnh hưởng đến tốc độ hội tụ
- **Early exaggeration:** Hệ số phóng đại ban đầu giúp tạo các cụm rõ ràng hơn
- **Số chiều đầu ra:** Thường là 2 hoặc 3 để trực quan hóa

2 Ứng dụng t-SNE với bộ dữ liệu mtcars

2.1 Tải thư viện cần thiết

```
library(tidyverse)   # Bộ thư viện chứa nhiều công cụ xử lý dữ liệu
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggplot2)     # Thư viện tạo đồ thị với cú pháp ngữ pháp đồ họa
library(Rtsne)       # Thư viện chứa thuật toán t-SNE trong R
library(factoextra)  # Thư viện cho phân tích đa chiều và trực quan hóa
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(viridis)     # Bảng màu đẹp cho trực quan hóa
```

```
## Loading required package: viridisLite
```

```
library(GGally)      # Mở rộng của ggplot2 cho ma trận tương quan
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

2.2 Hiểu về bộ dữ liệu mtcars

```
data(mtcars)  # Khôi phục lại bộ dữ liệu mặc định
# Xem cấu trúc bộ dữ liệu mtcars
str(mtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

```
# Hiển thị một số dòng đầu tiên
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
# Tóm tắt thống kê
summary(mtcars)
```

```
##       mpg             cyl             disp             hp
##  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0
##  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5
##  Median :19.20   Median :6.000   Median :196.3   Median :123.0
##  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7
##  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0
##  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0
##       drat             wt             qsec             vs
##  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000
##  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000
##  Median :3.695   Median :3.325   Median :17.71   Median :0.0000
##  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375
##  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000
##  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000
##        am              gear            carb
##  Min.   :0.0000   Min.   :3.000   Min.   :1.000
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000
##  Median :0.0000   Median :4.000   Median :2.000
```

```
##  Mean   :0.4062   Mean   :3.688   Mean   :2.812
##  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000
##  Max.   :1.0000   Max.   :5.000   Max.   :8.000
```

Bộ dữ liệu mtcars chứa thông tin về 32 mẫu xe với 11 biến mô tả các đặc điểm kỹ thuật:

- mpg: Miles per gallon (hiệu suất tiêu thụ nhiên liệu)
- cyl: Số xi-lanh
- disp: Dung tích xi-lanh
- hp: Mã lực
- drat: Tỷ số truyền động sau
- wt: Trọng lượng (1000 lbs)
- qsec: Thời gian chạy 1/4 dặm
- vs: Kiểu động cơ (0 = chữ V, 1 = thẳng hàng)
- am: Kiểu hộp số (0 = tự động, 1 = số sàn)
- gear: Số lượng số
- carb: Số lượng bộ chế hòa khí

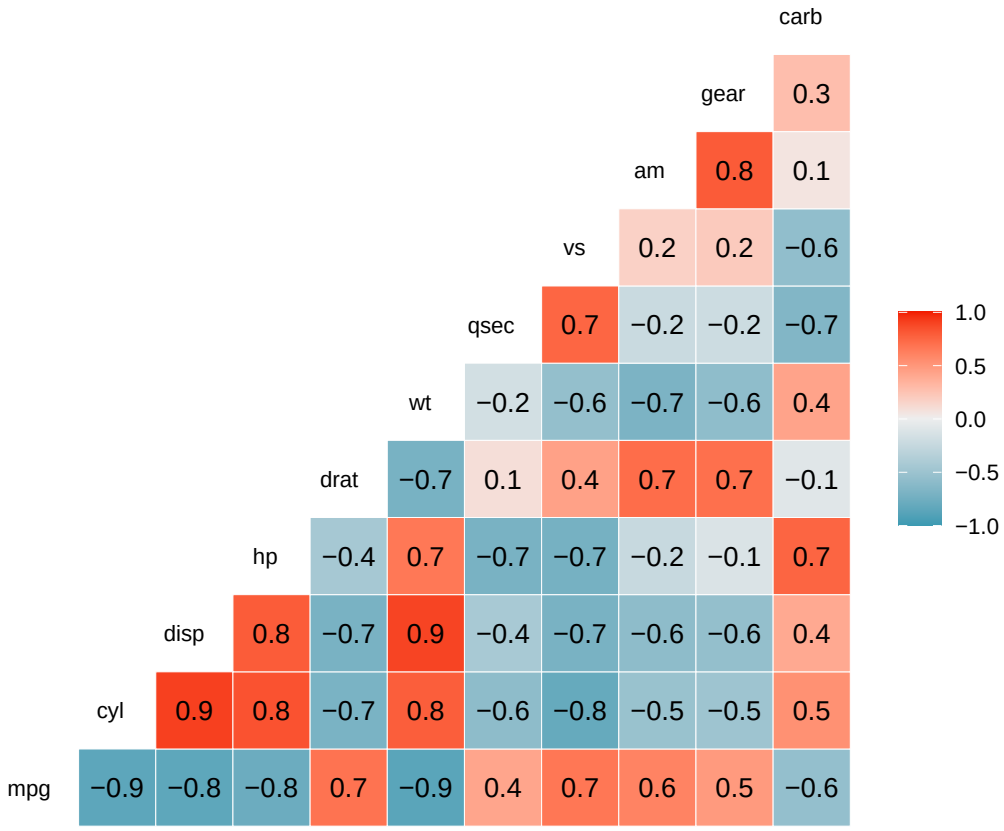
2.3 Phân tích tương quan

Trước khi thực hiện t-SNE, chúng ta hãy xem các biến có tương quan như thế nào:

```r
# Tạo ma trận tương quan
mtcars <-mtcars[sapply(mtcars, is.numeric)]
cor_matrix <- cor(mtcars)
round(cor_matrix, 2)
```

```
##        mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
## mpg   1.00 -0.85 -0.85 -0.78  0.68 -0.87  0.42  0.66  0.60  0.48 -0.55
## cyl  -0.85  1.00  0.90  0.83 -0.70  0.78 -0.59 -0.81 -0.52 -0.49  0.53
## disp -0.85  0.90  1.00  0.79 -0.71  0.89 -0.43 -0.71 -0.59 -0.56  0.39
## hp   -0.78  0.83  0.79  1.00 -0.45  0.66 -0.71 -0.72 -0.24 -0.13  0.75
## drat  0.68 -0.70 -0.71 -0.45  1.00 -0.71  0.09  0.44  0.71  0.70 -0.09
## wt   -0.87  0.78  0.89  0.66 -0.71  1.00 -0.17 -0.55 -0.69 -0.58  0.43
## qsec  0.42 -0.59 -0.43 -0.71  0.09 -0.17  1.00  0.74 -0.23 -0.21 -0.66
## vs    0.66 -0.81 -0.71 -0.72  0.44 -0.55  0.74  1.00  0.17  0.21 -0.57
## am    0.60 -0.52 -0.59 -0.24  0.71 -0.69 -0.23  0.17  1.00  0.79  0.06
## gear  0.48 -0.49 -0.56 -0.13  0.70 -0.58 -0.21  0.21  0.79  1.00  0.27
## carb -0.55  0.53  0.39  0.75 -0.09  0.43 -0.66 -0.57  0.06  0.27  1.00
```

```r
# Visualize correlation matrix
ggcorr(mtcars,
       method = c("everything", "pearson"),
       label = TRUE,
       hjust = 0.75,
       size = 3,
       layout.exp = 2)
```



Qua ma trận tương quan, chúng ta thấy nhiều biến có tương quan mạnh với nhau. Ví dụ:

- cyl, disp, hp và wt có tương quan dương mạnh với nhau
- Các biến trên có tương quan âm mạnh với mpg

Điều này chỉ ra rằng dữ liệu có thể có đa cộng tuyến và phù hợp để áp dụng t-SNE.

2.4 Tiền xử lý dữ liệu

```
# Tạo thêm biến nhóm để phân loại xe
mtcars_labeled <- mtcars %>%
  mutate(
    engine_type = factor(vs, labels = c("V-shaped", "Straight")),
    transmission = factor(am, labels = c("Automatic", "Manual")),
    cylinders = factor(cyl)
  )

# Chuẩn bị dữ liệu số cho t-SNE
mtcars_features <- mtcars %>% select(-vs, -am) # Bỏ biến phân loại nhị phân
# Chuẩn hóa dữ liệu
mtcars_scaled <- scale(mtcars_features)
```

2.5 Thực hiện t-SNE

```
# Kiểm tra điểm trùng lặp
duplicate_check <- duplicated(mtcars_scaled)
cat("Số điểm trùng lặp:", sum(duplicate_check), "\n")
```

```
## Số điểm trùng lặp: 0
```

```
# Loại bỏ điểm trùng lặp (nếu có)
mtcars_unique <- mtcars_scaled[!duplicate_check, ]
# Lấy nhãn tương ứng
row_indices <- which(!duplicate_check)
engine_type_unique <- mtcars_labeled$engine_type[row_indices]
transmission_unique <- mtcars_labeled$transmission[row_indices]
cylinders_unique <- mtcars_labeled$cylinders[row_indices]
car_names_unique <- rownames(mtcars)[row_indices]

# Hiển thị số lượng điểm sau khi loại bỏ trùng lặp
cat("Số điểm dữ liệu sau khi loại bỏ trùng lặp:", nrow(mtcars_unique), "\n")
```

```
## Số điểm dữ liệu sau khi loại bỏ trùng lặp: 32
```

```
# Xác định perplexity phù hợp (không nên quá lớn so với số mẫu)
n_samples <- nrow(mtcars_unique)
perplexity_val <- min(15, floor(n_samples/3))
cat("Perplexity được sử dụng:", perplexity_val, "\n")
```

```
## Perplexity được sử dụng: 10
```

```
set.seed(42) # Đặt seed để kết quả có thể tái tạo

# Thực hiện t-SNE
tsne_result <- Rtsne(mtcars_unique, dims = 2, perplexity = perplexity_val,
                     verbose = TRUE, max_iter = 1000, check_duplicates = FALSE)
```

```
## Performing PCA
## Read the 32 x 9 data matrix successfully!
## Using no_dims = 2, perplexity = 10.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 0.00 seconds (sparsity = 0.966797)!
## Learning embedding...
## Iteration 50: error is 55.223498 (50 iterations in 0.00 seconds)
## Iteration 100: error is 51.873754 (50 iterations in 0.00 seconds)
## Iteration 150: error is 51.776343 (50 iterations in 0.00 seconds)
## Iteration 200: error is 51.345640 (50 iterations in 0.00 seconds)
## Iteration 250: error is 55.322508 (50 iterations in 0.00 seconds)
## Iteration 300: error is 1.422781 (50 iterations in 0.00 seconds)
## Iteration 350: error is 0.944783 (50 iterations in 0.00 seconds)
```

```
## Iteration 400: error is 0.690733 (50 iterations in 0.00 seconds)
## Iteration 450: error is 0.376886 (50 iterations in 0.00 seconds)
## Iteration 500: error is 0.267057 (50 iterations in 0.00 seconds)
## Iteration 550: error is 0.162181 (50 iterations in 0.00 seconds)
## Iteration 600: error is 0.155146 (50 iterations in 0.00 seconds)
## Iteration 650: error is 0.155243 (50 iterations in 0.00 seconds)
## Iteration 700: error is 0.153783 (50 iterations in 0.00 seconds)
## Iteration 750: error is 0.154226 (50 iterations in 0.00 seconds)
## Iteration 800: error is 0.153362 (50 iterations in 0.00 seconds)
## Iteration 850: error is 0.155616 (50 iterations in 0.00 seconds)
## Iteration 900: error is 0.156909 (50 iterations in 0.00 seconds)
## Iteration 950: error is 0.156241 (50 iterations in 0.00 seconds)
## Iteration 1000: error is 0.153901 (50 iterations in 0.00 seconds)
## Fitting performed in 0.01 seconds.
```

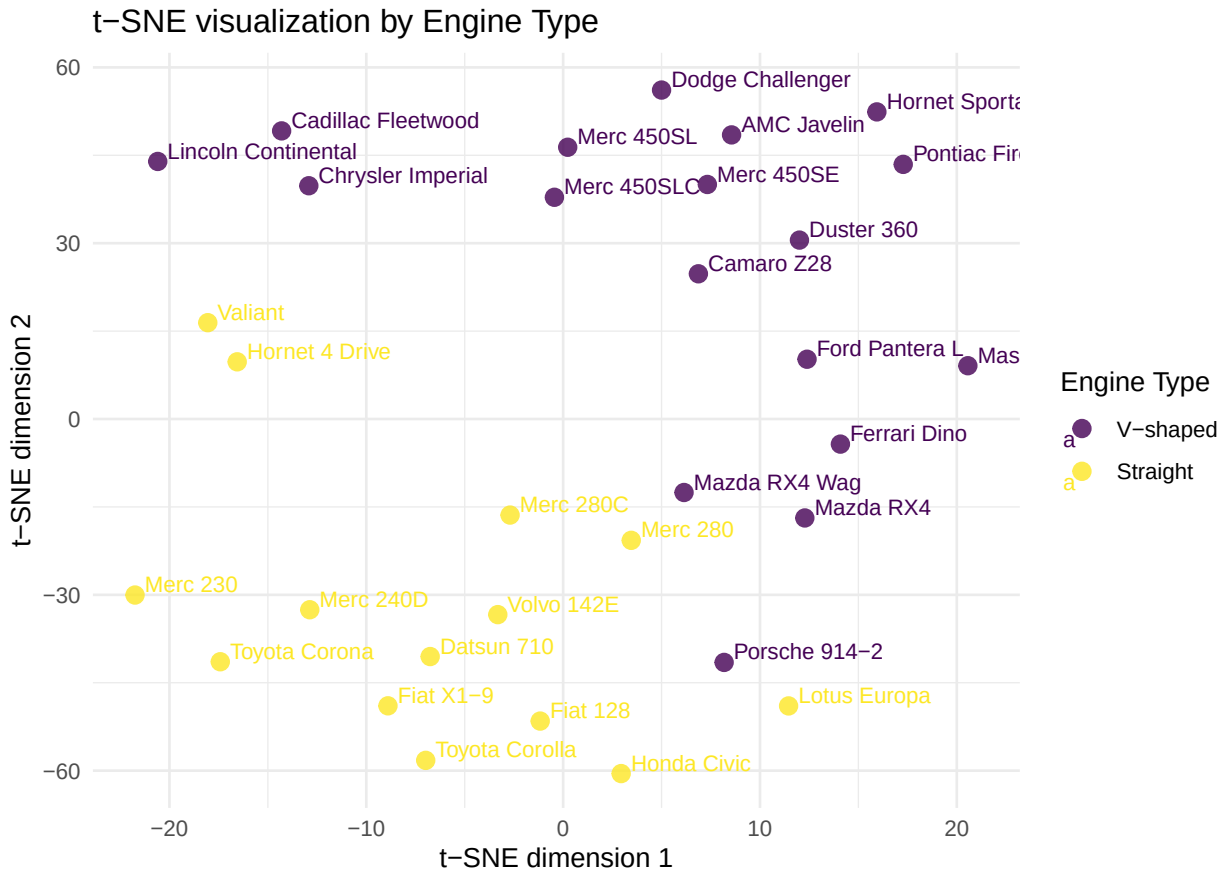
```r
# Tạo dataframe với kết quả t-SNE và nhãn
tsne_df <- data.frame(
  x = tsne_result$Y[, 1],
  y = tsne_result$Y[, 2],
  engine_type = engine_type_unique,
  transmission = transmission_unique,
  cylinders = cylinders_unique,
  car = car_names_unique
)

# Hiển thị kết quả
head(tsne_df)
```

```
##            x          y engine_type transmission cylinders               car
## 1  12.279890 -16.883450    V-shaped       Manual         6         Mazda RX4
## 2   6.142754 -12.539393    V-shaped       Manual         6     Mazda RX4 Wag
## 3  -6.755084 -40.531914    Straight       Manual         4        Datsun 710
## 4 -16.558422   9.758921    Straight    Automatic         6    Hornet 4 Drive
## 5  15.939243  52.403283    V-shaped    Automatic         8 Hornet Sportabout
## 6 -18.052556  16.438206    Straight    Automatic         6           Valiant
```

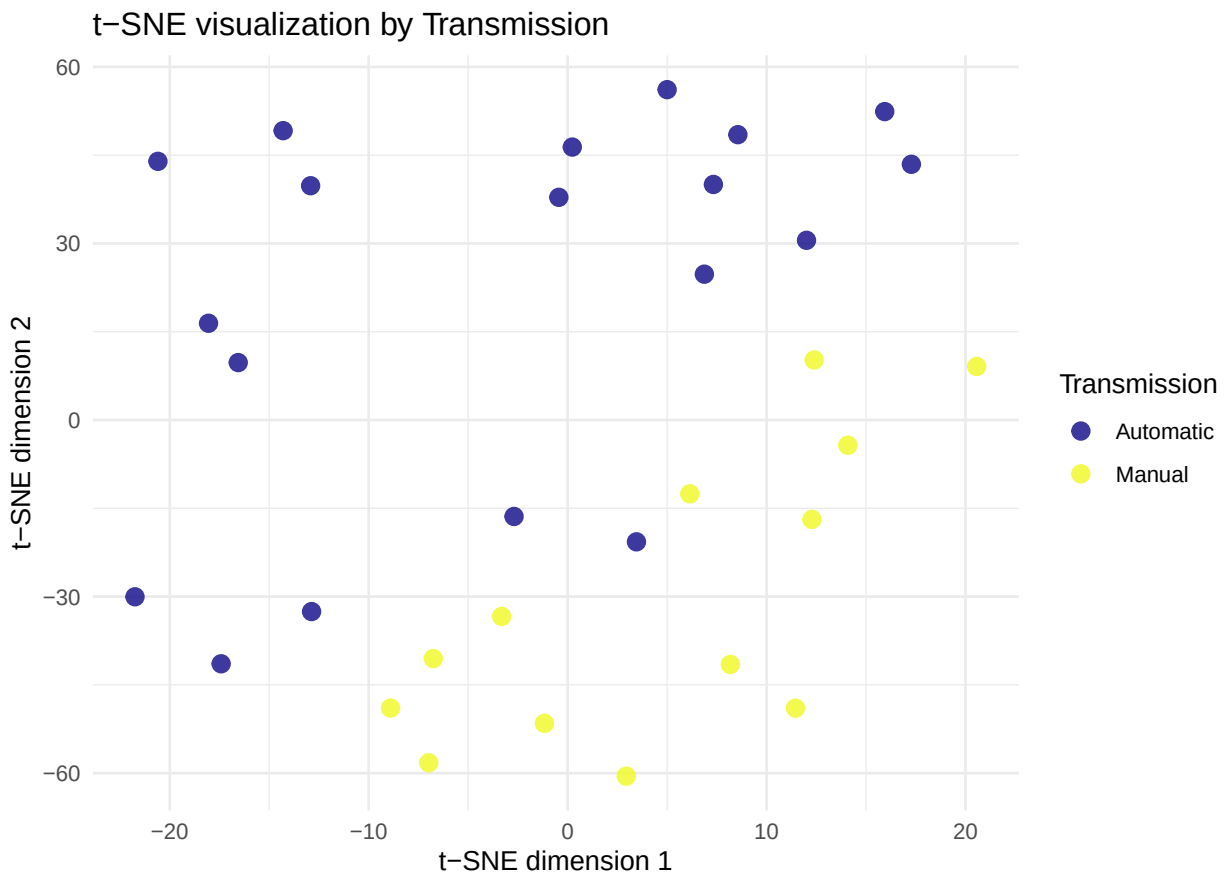
2.6 Trực quan hóa kết quả t-SNE

```r
# Trực quan hóa theo loại động cơ
p1 <- ggplot(tsne_df, aes(x = x, y = y, color = engine_type)) +
  geom_point(size = 3, alpha = 0.8) +
  geom_text(aes(label = car), hjust = 0, vjust = 0, size = 3, nudge_x = 0.5, nudge_y = 0.5, check_overla
  scale_color_viridis_d() +
  labs(title = "t-SNE visualization by Engine Type",
       x = "t-SNE dimension 1",
       y = "t-SNE dimension 2",
       color = "Engine Type") +
  theme_minimal()

# Trực quan hóa theo loại hộp số
p2 <- ggplot(tsne_df, aes(x = x, y = y, color = transmission)) +
```

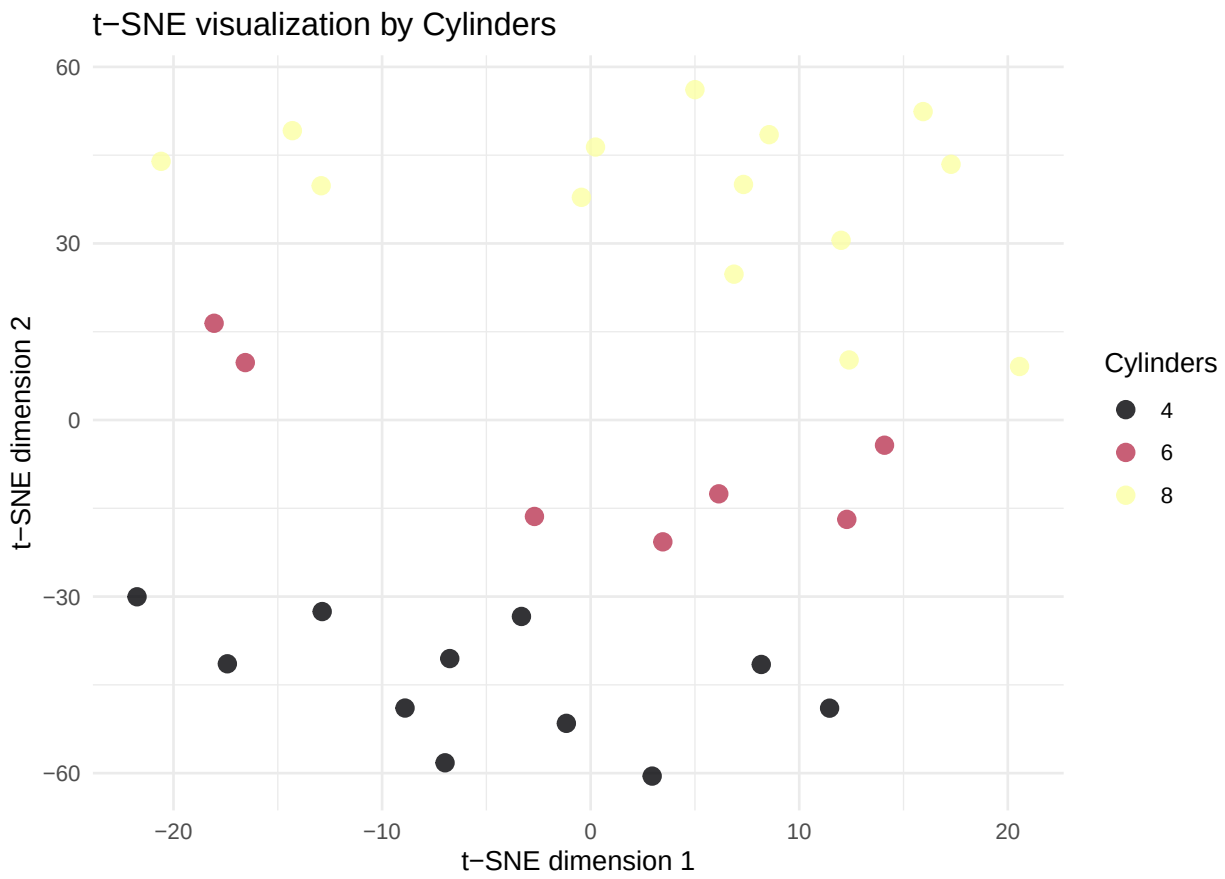
```
  geom_point(size = 3, alpha = 0.8) +
  scale_color_viridis_d(option = "plasma") +
  labs(title = "t-SNE visualization by Transmission",
       x = "t-SNE dimension 1",
       y = "t-SNE dimension 2",
       color = "Transmission") +
  theme_minimal()

# Trực quan hóa theo số xi-lanh
p3 <- ggplot(tsne_df, aes(x = x, y = y, color = cylinders)) +
  geom_point(size = 3, alpha = 0.8) +
  scale_color_viridis_d(option = "inferno") +
  labs(title = "t-SNE visualization by Cylinders",
       x = "t-SNE dimension 1",
       y = "t-SNE dimension 2",
       color = "Cylinders") +
  theme_minimal()

# Hiển thị các biểu đồ
p1
```




```
p2
```


t−SNE visualization by Transmission
60
30
0
−30
−60
t−SNE dimension 2
−20
−10
0
10
20
t−SNE dimension 1
Transmission
Automatic
Manual

```
p3
```

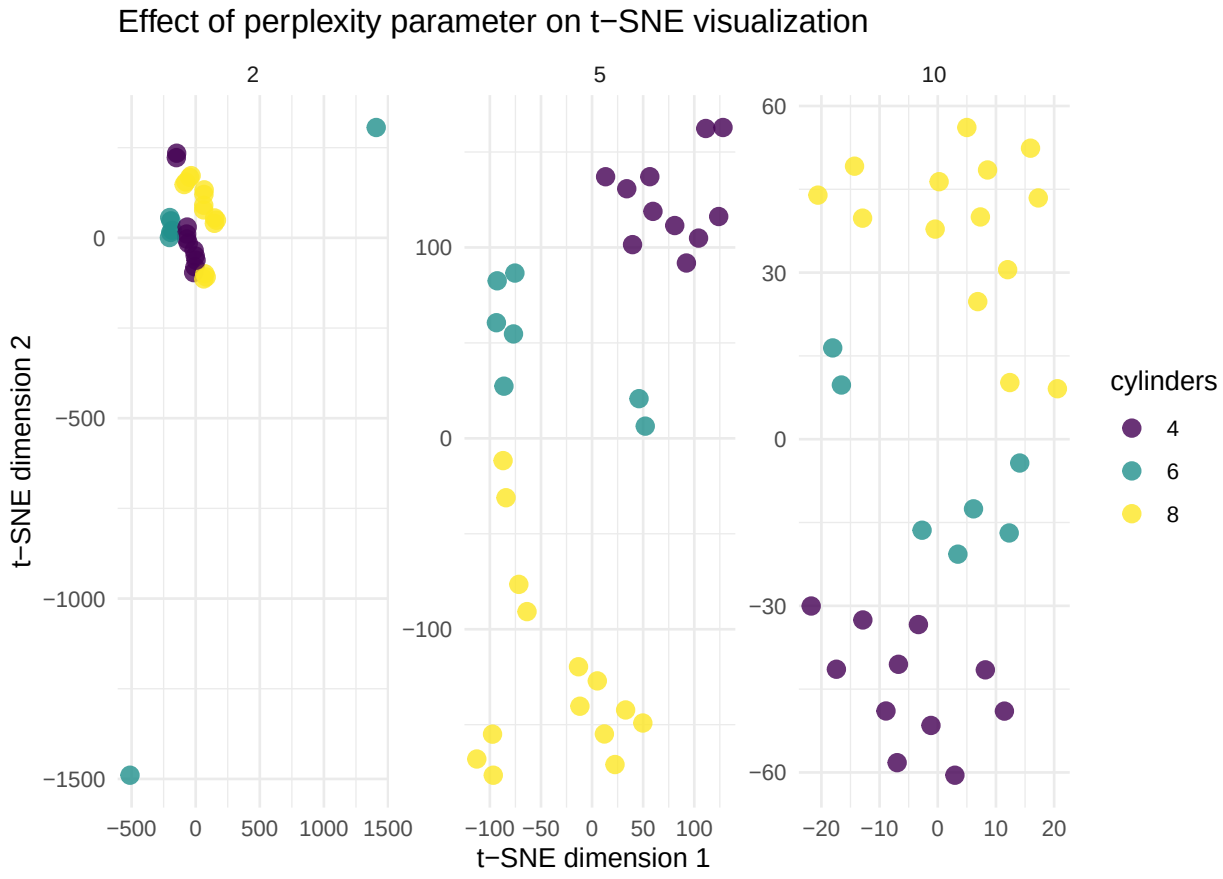


Nhận xét: - t-SNE đã tạo ra một biểu diễn 2D cho dữ liệu mtcars, trong đó các xe có đặc điểm tương tự thường nằm gần nhau - Có sự phân tách khá rõ ràng giữa các loại động cơ, kiểu hộp số và số xi-lanh - Có thể thấy xu hướng các xe có động cơ V-shaped thường nằm gần nhau, tương tự với các xe có số xi-lanh giống nhau

2.7 Khám phá ảnh hưởng của tham số perplexity

```
# Tạo hàm để thực hiện t-SNE với perplexity khác nhau
run_tsne <- function(perplexity_val) {
  set.seed(42)
  # Sử dụng dữ liệu đã loại bỏ trùng lặp
  tsne <- Rtsne(mtcars_unique, dims = 2, perplexity = perplexity_val,
                verbose = FALSE, max_iter = 1000, check_duplicates = FALSE)

  data.frame(
    x = tsne$Y[, 1],
    y = tsne$Y[, 2],
    cylinders = cylinders_unique,
    perplexity = as.factor(perplexity_val)
  )
}

# Thực hiện t-SNE với các giá trị perplexity khác nhau
# Perplexity không nên lớn hơn ~n/3 với n là số điểm dữ liệu
```

```r
max_perplexity <- floor(n_samples/3)

# Chọn các giá trị perplexity phù hợp với kích thước dữ liệu
perplexity_values <- c(2, 5, min(10, max_perplexity), min(15, max_perplexity))
perplexity_values <- unique(perplexity_values) # Loại bỏ giá trị trùng lặp nếu có

tsne_results <- do.call(rbind, lapply(perplexity_values, run_tsne))

# Vẽ biểu đồ so sánh
ggplot(tsne_results, aes(x = x, y = y, color = cylinders)) +
  geom_point(size = 3, alpha = 0.8) +
  scale_color_viridis_d() +
  facet_wrap(~perplexity, scales = "free") +
  labs(title = "Effect of perplexity parameter on t-SNE visualization",
       x = "t-SNE dimension 1",
       y = "t-SNE dimension 2") +
  theme_minimal()
```



Nhận xét về ảnh hưởng của perplexity: - Perplexity thấp (2-5): Tập trung vào cấu trúc cục bộ, có thể tạo ra nhiều cụm nhỏ - Perplexity cao (10-15): Tập trung vào cấu trúc toàn cục, các cụm có xu hướng gộp lại - Với tập dữ liệu nhỏ như mtcars, perplexity không nên quá cao (không quá 1/3 số mẫu)

3 So sánh t-SNE với PCA

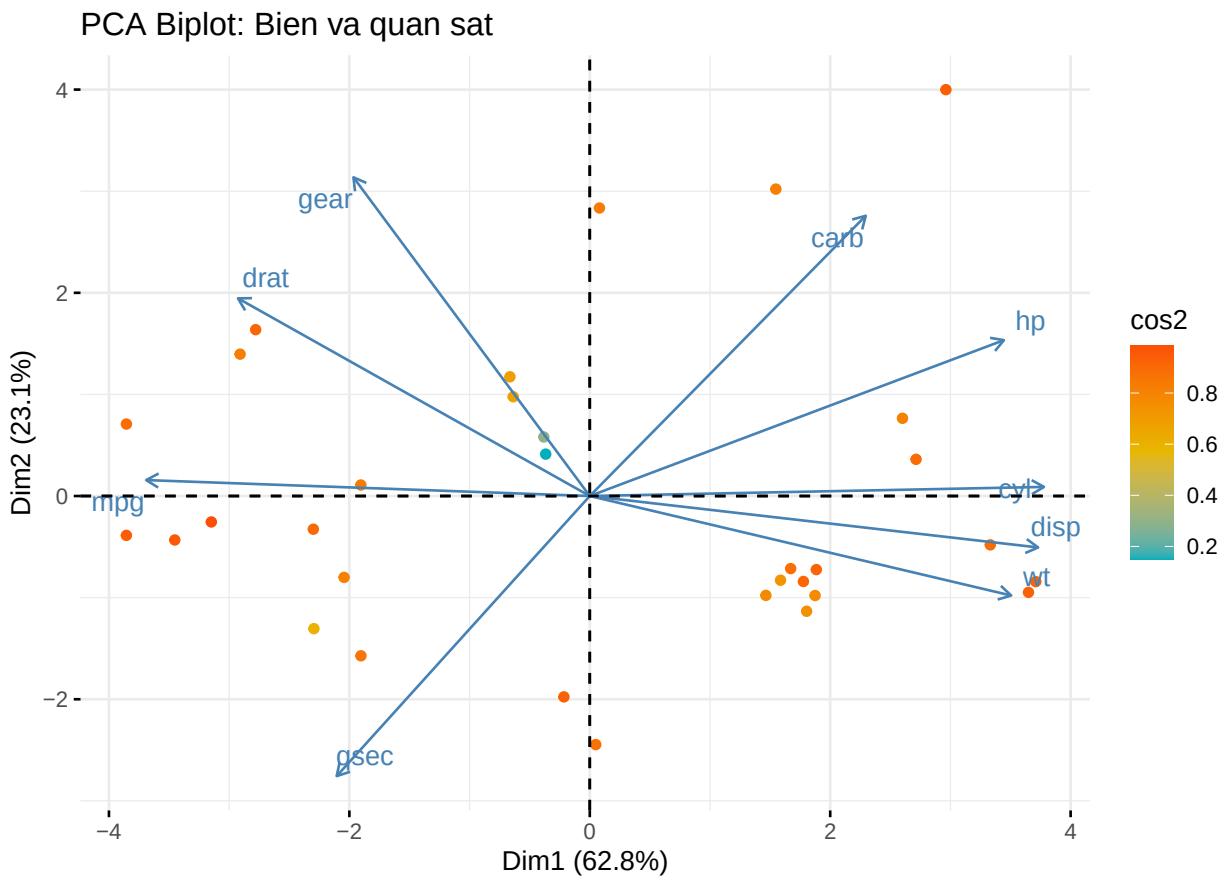
3.1 Thực hiện PCA

```
# Thực hiện PCA trên cùng bộ dữ liệu mtcars
pca_result <- prcomp(mtcars_unique, center = TRUE, scale. = TRUE)

# Tạo dataframe với kết quả PCA và nhãn
pca_df <- data.frame(
  x = pca_result$x[, 1],
  y = pca_result$x[, 2],
  engine_type = engine_type_unique,
  transmission = transmission_unique,
  cylinders = cylinders_unique,
  car = car_names_unique
)

# Xem tóm tắt kết quả PCA
summary(pca_result)
```

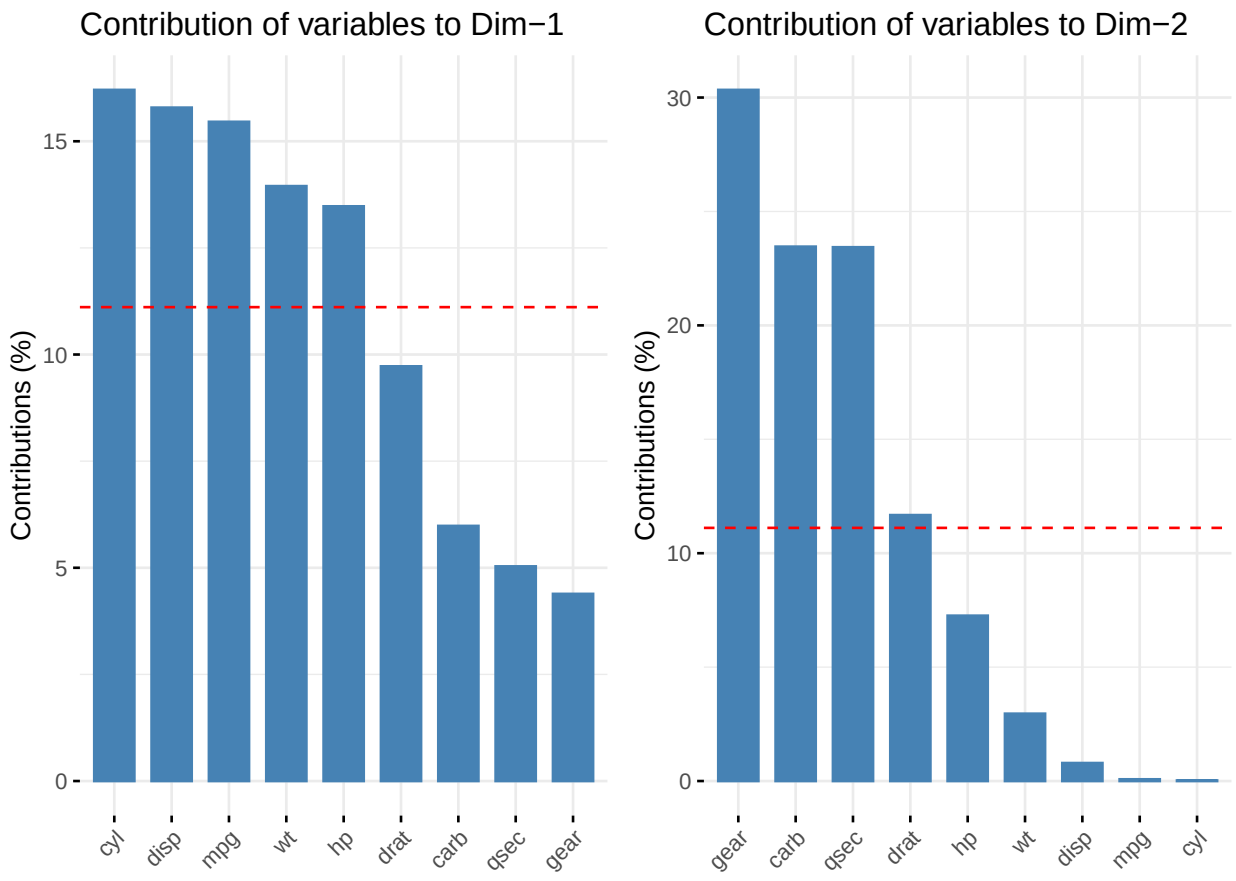
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.3782 1.4429 0.71008 0.51481 0.42797 0.35184 0.32413
## Proportion of Variance 0.6284 0.2313 0.05602 0.02945 0.02035 0.01375 0.01167
## Cumulative Proportion  0.6284 0.8598 0.91581 0.94525 0.96560 0.97936 0.99103
##                           PC8     PC9
## Standard deviation     0.2419 0.14896
## Proportion of Variance 0.0065 0.00247
## Cumulative Proportion  0.9975 1.00000
```

3.2 Phân tích các thành phần chính

```
# Biplot phân tích các biến đóng góp vào PCA
fviz_pca_biplot(pca_result,
                label = "var",
                col.ind = "cos2",
                gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
                repel = TRUE,
                title = "PCA Biplot: Bien va quan sat")
```



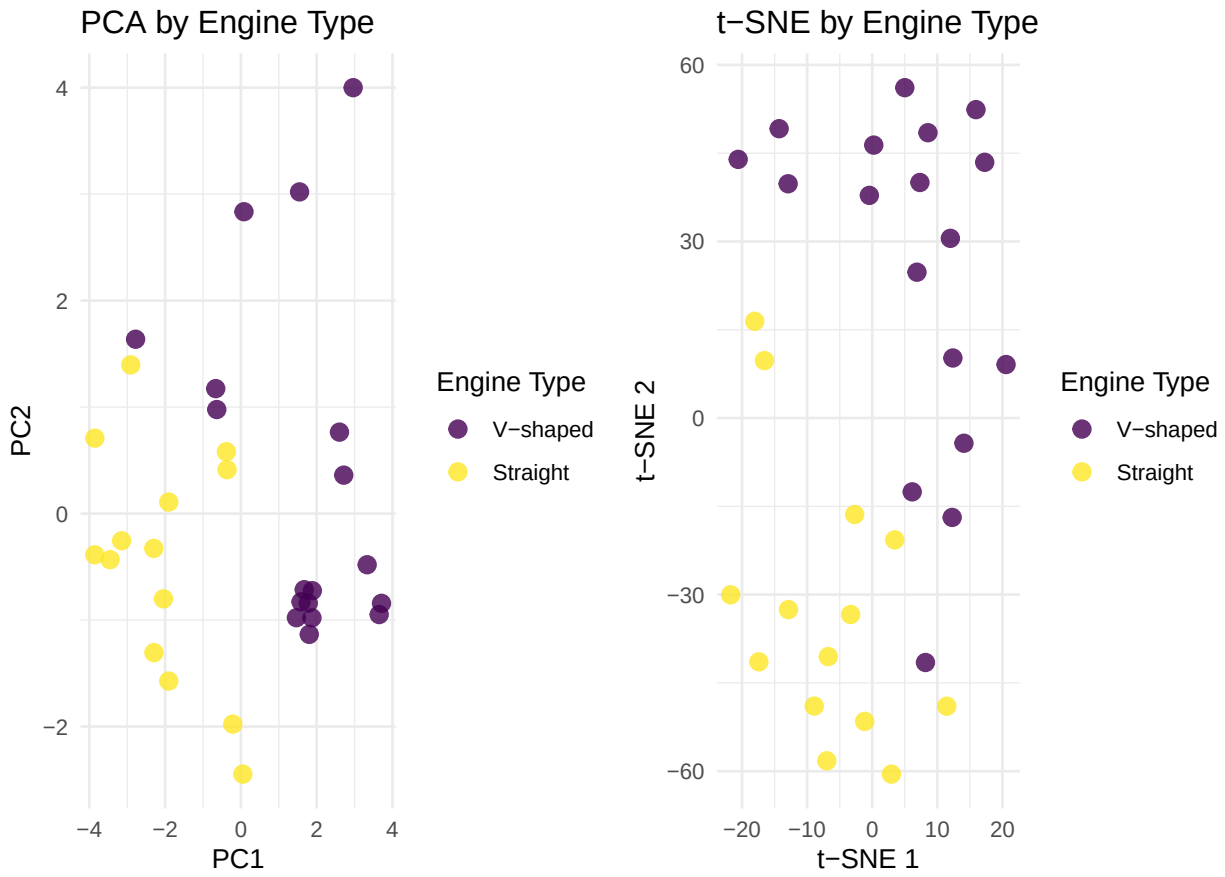

```
# Phân tích đóng góp của các biến vào PC1 và PC2
p1 <- fviz_contrib(pca_result, choice = "var", axes = 1, top = 10)
p2 <- fviz_contrib(pca_result, choice = "var", axes = 2, top = 10)
gridExtra::grid.arrange(p1, p2, ncol = 2)
```



3.3 So sánh trực quan giữa PCA và t-SNE

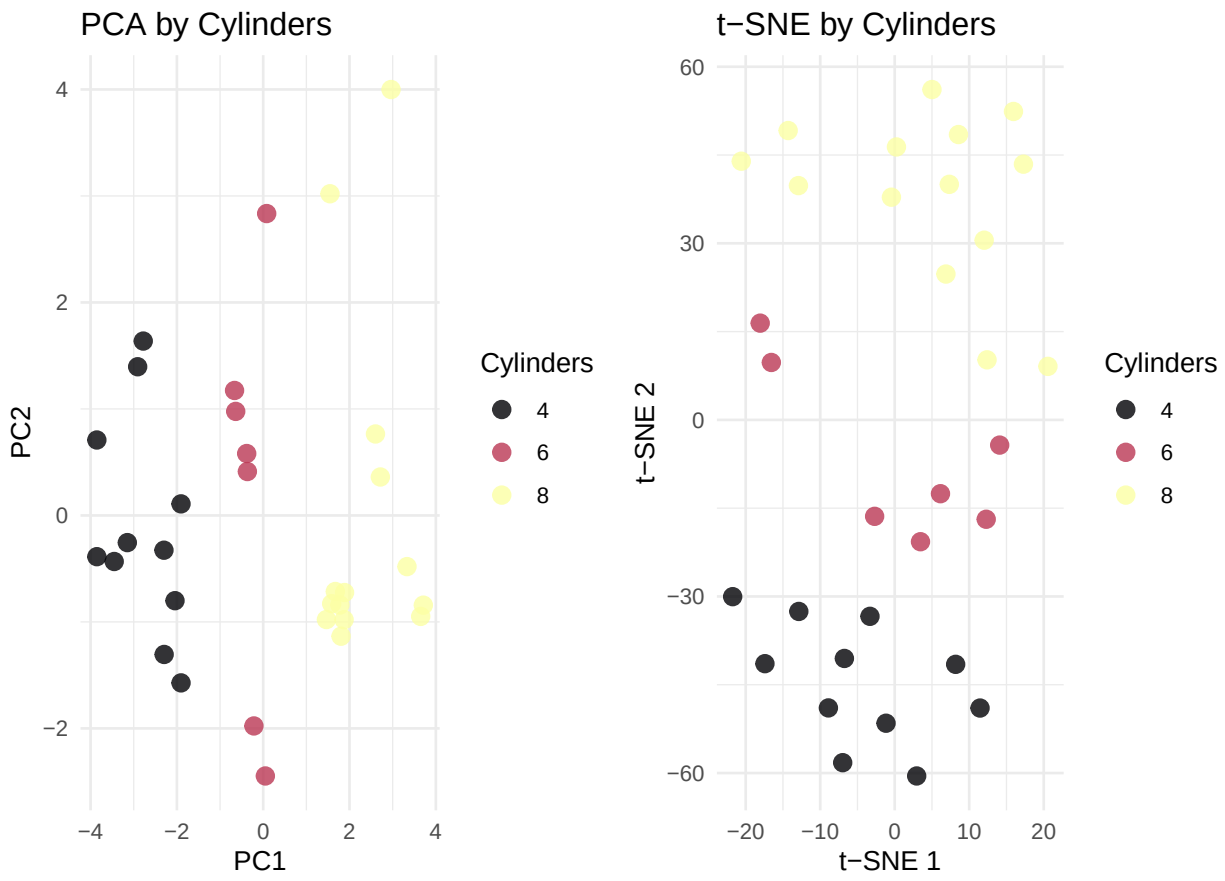
```
# Vẽ biểu đồ PCA theo kiểu động cơ
pca_plot1 <- ggplot(pca_df, aes(x = x, y = y, color = engine_type)) +
  geom_point(size = 3, alpha = 0.8) +
  scale_color_viridis_d() +
  labs(title = "PCA by Engine Type",
       x = "PC1",
       y = "PC2",
       color = "Engine Type") +
  theme_minimal()

# Vẽ biểu đồ t-SNE theo kiểu động cơ
tsne_plot1 <- ggplot(tsne_df, aes(x = x, y = y, color = engine_type)) +
  geom_point(size = 3, alpha = 0.8) +
  scale_color_viridis_d() +
  labs(title = "t-SNE by Engine Type",
       x = "t-SNE 1",
       y = "t-SNE 2",
       color = "Engine Type") +
  theme_minimal()
```

```
# Hiển thị so sánh theo kiểu động cơ
gridExtra::grid.arrange(pca_plot1, tsne_plot1, ncol = 2)
```




```
# Vẽ biểu đồ PCA theo số xi-lanh
pca_plot2 <- ggplot(pca_df, aes(x = x, y = y, color = cylinders)) +
  geom_point(size = 3, alpha = 0.8) +
  scale_color_viridis_d(option = "inferno") +
  labs(title = "PCA by Cylinders",
       x = "PC1",
       y = "PC2",
       color = "Cylinders") +
  theme_minimal()

# Vẽ biểu đồ t-SNE theo số xi-lanh
tsne_plot2 <- ggplot(tsne_df, aes(x = x, y = y, color = cylinders)) +
  geom_point(size = 3, alpha = 0.8) +
  scale_color_viridis_d(option = "inferno") +
  labs(title = "t-SNE by Cylinders",
       x = "t-SNE 1",
       y = "t-SNE 2",
       color = "Cylinders") +
  theme_minimal()

# Hiển thị so sánh theo số xi-lanh
```

```
gridExtra::grid.arrange(pca_plot2, tsne_plot2, ncol = 2)
```



Nhận xét khi so sánh PCA và t-SNE trên bộ dữ liệu mtcars:

- PCA là phương pháp tuyến tính tối đa hóa phương sai, cho phép chúng ta hiểu được đóng góp của các biến gốc
- t-SNE là phương pháp phi tuyến tính tập trung vào bảo toàn cấu trúc cục bộ
- Trong PCA, chúng ta thấy PC1 giải thích khoảng 60% phương sai, và chủ yếu liên quan đến các biến về kích thước động cơ (disp, cyl) và hiệu suất (hp, mpg)
- t-SNE có xu hướng tạo ra các cụm rõ ràng hơn, đặc biệt cho các nhóm xe có số xi-lanh giống nhau
- Cả hai phương pháp đều cho thấy sự phân tách giữa các nhóm xe, nhưng t-SNE có thể làm nổi bật các mẫu cục bộ mà PCA có thể bỏ qua
- PCA có ưu điểm là nhanh hơn, có thể áp dụng cho dữ liệu mới, và giúp hiểu cấu trúc dữ liệu gốc thông qua loadings
- t-SNE thường tốt hơn cho mục đích trực quan hóa và khám phá dữ liệu, nhưng không thể hiện được đóng góp của các biến gốc

số viết tay).

3.4 Phân tích bộ dữ liệu USArrests với t-SNE

```
# Tải bộ dữ liệu USArrests
data(USArrests)
```

```r
# Xem cấu trúc và thông tin cơ bản
str(USArrests)
```

```
## 'data.frame':    50 obs. of  4 variables:
##  $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
##  $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
##  $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
##  $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...
```

```r
head(USArrests)
```

```
##            Murder Assault UrbanPop Rape
## Alabama      13.2     236       58 21.2
## Alaska       10.0     263       48 44.5
## Arizona       8.1     294       80 31.0
## Arkansas      8.8     190       50 19.5
## California    9.0     276       91 40.6
## Colorado      7.9     204       78 38.7
```

```r
# Tóm tắt thống kê
summary(USArrests)
```

```
##      Murder          Assault         UrbanPop          Rape
##  Min.   : 0.800   Min.   : 45.0   Min.   :32.00   Min.   : 7.30
##  1st Qu.: 4.075   1st Qu.:109.0   1st Qu.:54.50   1st Qu.:15.07
##  Median : 7.250   Median :159.0   Median :66.00   Median :20.10
##  Mean   : 7.788   Mean   :170.8   Mean   :65.54   Mean   :21.23
##  3rd Qu.:11.250   3rd Qu.:249.0   3rd Qu.:77.75   3rd Qu.:26.18
##  Max.   :17.400   Max.   :337.0   Max.   :91.00   Max.   :46.00
```

Bộ dữ liệu USArrests chứa thông tin về tỷ lệ tội phạm cho 50 bang của Hoa Kỳ vào năm 1973 với các biến: - Murder: Số vụ giết người trên 100,000 dân - Assault: Số vụ hành hung trên 100,000 dân - UrbanPop: Tỷ lệ dân số đô thị (%) - Rape: Số vụ hiếp dâm trên 100,000 dân

```r
# Chuẩn bị dữ liệu
state_names <- rownames(USArrests)
arrests_data <- USArrests

# Chuẩn hóa dữ liệu
arrests_scaled <- scale(arrests_data)

# Kiểm tra điểm trùng lặp
arrests_duplicates <- duplicated(arrests_scaled)
cat("Số bang trùng lặp:", sum(arrests_duplicates), "\n")
```

```
## Số bang trùng lặp: 0
```

```r
# Nếu có điểm trùng lặp, loại bỏ chúng
arrests_unique <- arrests_scaled[!arrests_duplicates, ]
state_names_unique <- state_names[!arrests_duplicates]
```

```r
# Thực hiện t-SNE
set.seed(42)
# Chọn perplexity phù hợp (không quá lớn so với số mẫu)
perplexity_val <- min(15, floor(nrow(arrests_unique)/3))
arrests_tsne <- Rtsne(arrests_unique, dims = 2, perplexity = perplexity_val,
                      verbose = TRUE, max_iter = 1000, check_duplicates = FALSE)
```

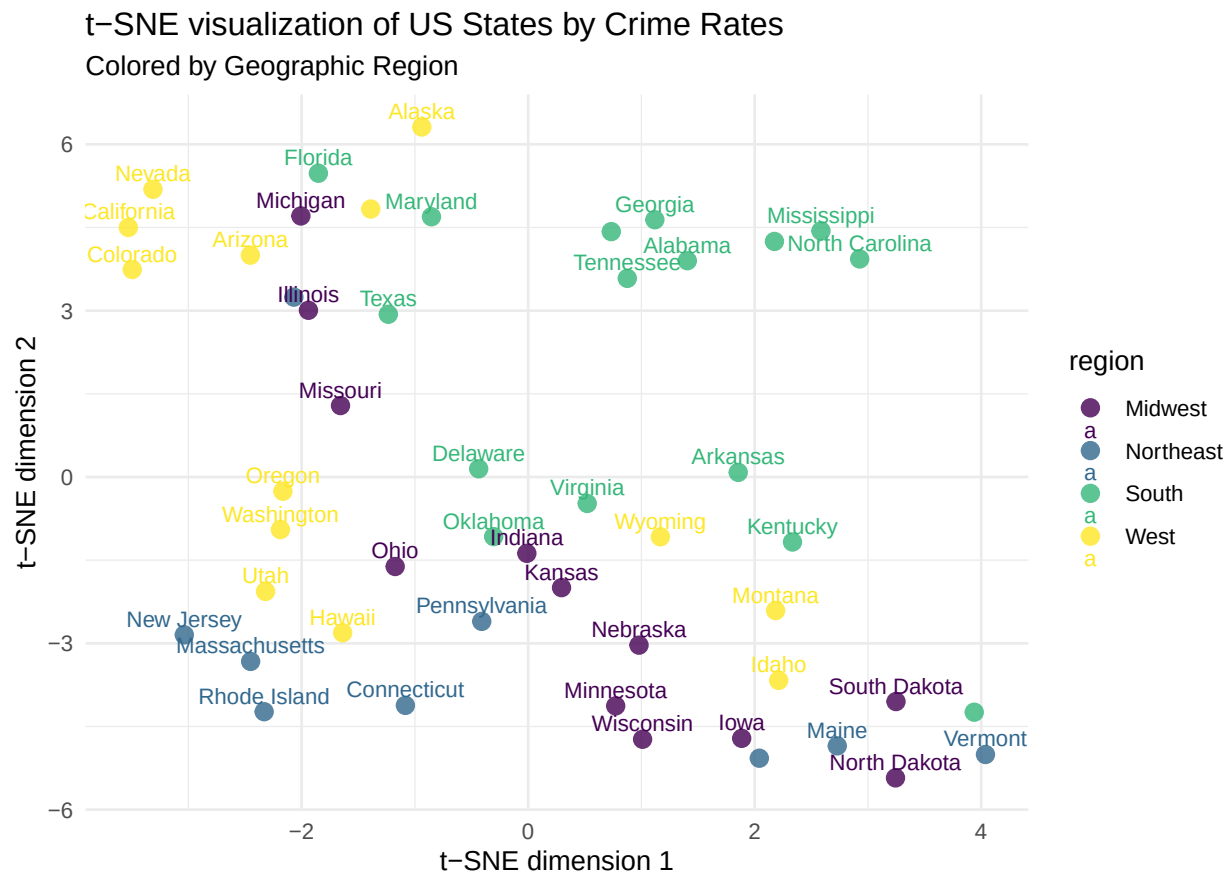
```
## Performing PCA
## Read the 50 x 4 data matrix successfully!
## Using no_dims = 2, perplexity = 15.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 0.00 seconds (sparsity = 0.964800)!
## Learning embedding...
## Iteration 50: error is 59.209301 (50 iterations in 0.00 seconds)
## Iteration 100: error is 49.562359 (50 iterations in 0.00 seconds)
## Iteration 150: error is 54.147123 (50 iterations in 0.00 seconds)
## Iteration 200: error is 51.703217 (50 iterations in 0.00 seconds)
## Iteration 250: error is 54.216927 (50 iterations in 0.00 seconds)
## Iteration 300: error is 1.298946 (50 iterations in 0.00 seconds)
## Iteration 350: error is 0.904706 (50 iterations in 0.00 seconds)
## Iteration 400: error is 0.645064 (50 iterations in 0.00 seconds)
## Iteration 450: error is 0.295438 (50 iterations in 0.00 seconds)
## Iteration 500: error is 0.162382 (50 iterations in 0.00 seconds)
## Iteration 550: error is 0.144645 (50 iterations in 0.00 seconds)
## Iteration 600: error is 0.145044 (50 iterations in 0.00 seconds)
## Iteration 650: error is 0.145380 (50 iterations in 0.00 seconds)
## Iteration 700: error is 0.146756 (50 iterations in 0.00 seconds)
## Iteration 750: error is 0.147871 (50 iterations in 0.00 seconds)
## Iteration 800: error is 0.149949 (50 iterations in 0.00 seconds)
## Iteration 850: error is 0.147925 (50 iterations in 0.00 seconds)
## Iteration 900: error is 0.146904 (50 iterations in 0.00 seconds)
## Iteration 950: error is 0.147425 (50 iterations in 0.00 seconds)
## Iteration 1000: error is 0.148824 (50 iterations in 0.00 seconds)
## Fitting performed in 0.02 seconds.
```

```r
# Tạo dataframe với kết quả
arrests_tsne_df <- data.frame(
  x = arrests_tsne$Y[, 1],
  y = arrests_tsne$Y[, 2],
  state = state_names_unique
)

# Kết hợp dữ liệu gốc với kết quả t-SNE
arrests_tsne_df <- cbind(arrests_tsne_df, arrests_data[!arrests_duplicates, ])

# Thêm thông tin vùng địa lý
# Phân chia các bang theo vùng (chỉ là ví dụ, có thể điều chỉnh)
northeast <- c("Maine", "New Hampshire", "Vermont", "Massachusetts", "Rhode Island",
               "Connecticut", "New York", "New Jersey", "Pennsylvania")
midwest <- c("Ohio", "Indiana", "Illinois", "Michigan", "Wisconsin",
             "Minnesota", "Iowa", "Missouri", "North Dakota", "South Dakota",
```

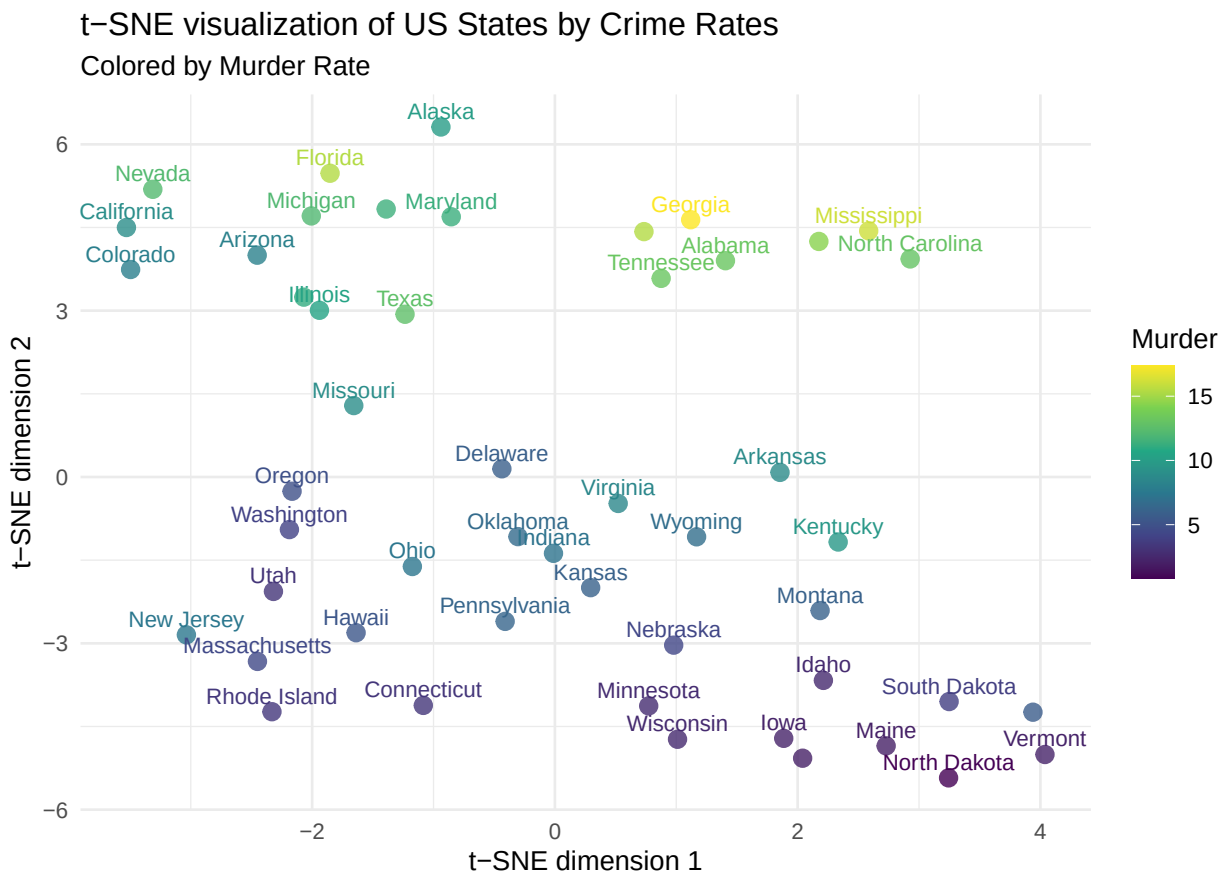
```
             "Nebraska", "Kansas")
south <- c("Delaware", "Maryland", "Virginia", "West Virginia", "North Carolina",
           "South Carolina", "Georgia", "Florida", "Kentucky", "Tennessee",
           "Alabama", "Mississippi", "Arkansas", "Louisiana", "Oklahoma", "Texas")
west <- c("Montana", "Idaho", "Wyoming", "Colorado", "New Mexico", "Arizona", "Utah",
          "Nevada", "Washington", "Oregon", "California", "Alaska", "Hawaii")

# Thêm cột vùng
arrests_tsne_df$region <- NA
arrests_tsne_df$region[arrests_tsne_df$state %in% northeast] <- "Northeast"
arrests_tsne_df$region[arrests_tsne_df$state %in% midwest] <- "Midwest"
arrests_tsne_df$region[arrests_tsne_df$state %in% south] <- "South"
arrests_tsne_df$region[arrests_tsne_df$state %in% west] <- "West"
arrests_tsne_df$region <- factor(arrests_tsne_df$region)
```

3.5 Trực quan hóa kết quả t-SNE trên USArrests

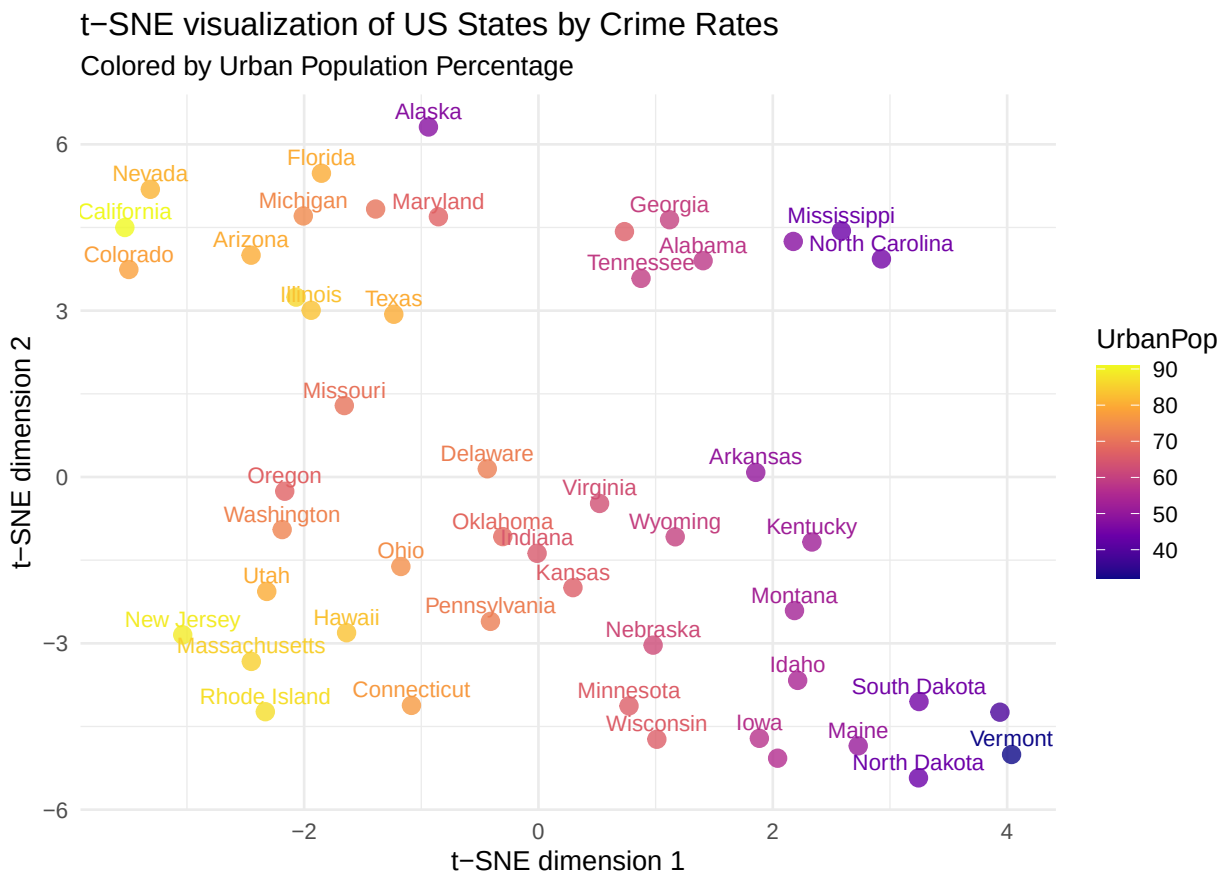
```
# Vẽ biểu đồ t-SNE cho USArrests theo vùng địa lý
p1 <- ggplot(arrests_tsne_df, aes(x = x, y = y, color = region)) +
  geom_point(size = 3, alpha = 0.8) +
  geom_text(aes(label = state), vjust = -0.5, size = 3, check_overlap = TRUE) +
  scale_color_viridis_d() +
  labs(title = "t-SNE visualization of US States by Crime Rates",
       subtitle = "Colored by Geographic Region",
       x = "t-SNE dimension 1",
       y = "t-SNE dimension 2") +
  theme_minimal()

# Vẽ biểu đồ theo tỷ lệ Murder
p2 <- ggplot(arrests_tsne_df, aes(x = x, y = y, color = Murder)) +
  geom_point(size = 3, alpha = 0.8) +
  geom_text(aes(label = state), vjust = -0.5, size = 3, check_overlap = TRUE) +
  scale_color_viridis_c() +
  labs(title = "t-SNE visualization of US States by Crime Rates",
       subtitle = "Colored by Murder Rate",
       x = "t-SNE dimension 1",
       y = "t-SNE dimension 2") +
  theme_minimal()

# Hiển thị các biểu đồ
p1
```


t−SNE visualization of US States by Crime Rates
Colored by Geographic Region
Alaska
Florida
Nevada
Michigan
Maryland
Georgia
Mississippi
California
Arizona
Alabama
North Carolina
Colorado
Tennessee
Illinois
Texas
Missouri
Delaware
Arkansas
Oregon
Virginia
Washington
Oklahoma
Wyoming
Kentucky
Indiana
Ohio
Kansas
Utah
Montana
Pennsylvania
New Jersey
Hawaii
Nebraska
Massachusetts
Idaho
Rhode Island
Connecticut
Minnesota
South Dakota
Wisconsin
Iowa
Maine
Vermont
North Dakota
6
3
0
−3
−6
−2
0
2
4
t−SNE dimension 1
t−SNE dimension 2
region
Midwest
Northeast
South
West

```
p2
```




```r
# Vẽ biểu đồ theo tỷ lệ đô thị hóa
p3 <- ggplot(arrests_tsne_df, aes(x = x, y = y, color = UrbanPop)) +
  geom_point(size = 3, alpha = 0.8) +
  geom_text(aes(label = state), vjust = -0.5, size = 3, check_overlap = TRUE) +
  scale_color_viridis_c(option = "plasma") +
  labs(title = "t-SNE visualization of US States by Crime Rates",
       subtitle = "Colored by Urban Population Percentage",
       x = "t-SNE dimension 1",
       y = "t-SNE dimension 2") +
  theme_minimal()

p3
```



Nhận xét: - t-SNE đã tạo ra biểu diễn 2D của các bang dựa trên mẫu tội phạm và tỷ lệ đô thị hóa - Có thể thấy các bang trong cùng một vùng địa lý thường có xu hướng nằm gần nhau trên không gian t-SNE - Các bang có tỷ lệ tội phạm tương tự nhau được nhóm lại với nhau - Biểu đồ màu theo tỷ lệ Murder cho thấy các bang với tỷ lệ giết người cao (màu sáng) thường tập trung ở một khu vực - Tỷ lệ đô thị hóa cũng cho thấy một mẫu thú vị trong kết quả t-SNE

4 Giảm chiều dữ liệu với t-SNE cho bài toán thực tế

4.1 Ứng dụng t-SNE trong phân tích dữ liệu lớn

Trong phần này, chúng ta sẽ thảo luận về cách t-SNE có thể được áp dụng trong các bài toán thực tế với dữ liệu lớn.

4.1.1 Quy trình xử lý dữ liệu lớn với t-SNE

Khi làm việc với dữ liệu có số chiều cao và kích thước lớn, quy trình tiếp cận thường là:

1. **Làm sạch và chuẩn hóa dữ liệu**
   - Xử lý giá trị thiếu, loại bỏ ngoại lai
   - Chuẩn hóa các biến số
2. **Giảm kích thước dữ liệu (nếu cần)**
   - Với dữ liệu rất lớn (hàng triệu mẫu), có thể lấy mẫu ngẫu nhiên
   - Với dữ liệu có nhiều chiều (hàng nghìn biến), có thể dùng PCA trước để giảm xuống 50-100 chiều

3. **Áp dụng t-SNE**
    - Chọn perplexity phù hợp với kích thước dữ liệu
    - Thử nghiệm với nhiều giá trị tham số
    - Đánh giá tính ổn định của kết quả
4. **Giải thích và ứng dụng kết quả**
    - Kết hợp với các phương pháp phân cụm
    - Sử dụng để trực quan hóa và khám phá dữ liệu

4.1.2 Ví dụ về các bài toán thực tế

1. **Phân tích dữ liệu gene và protein**
    - Giảm chiều dữ liệu biểu hiện gene (thường có hàng nghìn gene)
    - Phát hiện nhóm tế bào có chức năng tương tự
2. **Phân tích văn bản và xử lý ngôn ngữ tự nhiên**
    - Trực quan hóa không gian embedding của từ hoặc văn bản
    - Phát hiện các chủ đề tương tự
3. **Phát hiện gian lận và bất thường**
    - Giảm chiều các đặc trưng giao dịch
    - Xác định các mẫu giao dịch bất thường

4.2 Các khuyến nghị khi sử dụng t-SNE

1. **Tiền xử lý dữ liệu:**
    - Chuẩn hóa dữ liệu trước khi áp dụng t-SNE
    - Loại bỏ các biến không liên quan hoặc nhiễu
    - Với dữ liệu lớn, cân nhắc giảm chiều bằng PCA trước
2. **Điều chỉnh tham số:**
    - **Perplexity**: Thử nghiệm với nhiều giá trị (5-50), không vượt quá 1/3 số mẫu
    - **Số lần lặp**: Tăng số lần lặp (500-2000) nếu kết quả chưa hội tụ
    - **Learning rate**: Điều chỉnh nếu kết quả không ổn định
3. **Giải thích kết quả một cách thận trọng:**
    - Chỉ giải thích về khoảng cách tương đối giữa các điểm lân cận
    - Không nên giải thích về kích thước của các cụm
    - Khoảng cách giữa các cụm xa nhau không mang nhiều ý nghĩa
4. **Đánh giá tính ổn định:**
    - Thực hiện t-SNE nhiều lần với các seed khác nhau
    - So sánh kết quả giữa các lần chạy

4.3 So sánh với các phương pháp giảm chiều khác

Phương pháp	Tuyến tính	Bảo toàn cấu trúc	Tốc độ	Khả năng mở rộng	Dễ giải thích	Ứng dụng chính
PCA	Có	Toàn cục	Nhanh	Tốt	Cao	Giảm chiều, xử lý đa cộng tuyến
t-SNE	Không	Cục bộ	Chậm	Kém	Thấp	Trực quan hóa, phát hiện cụm
UMAP	Không	Cục bộ & Toàn cục	Nhanh hơn t-SNE	Tốt hơn t-SNE	Trung bình	Trực quan hóa, phân cụm
LDA	Có	Phân biệt lớp	Nhanh	Tốt	Cao	Phân loại có giám sát
MDS	Tùy thuộc	Toàn cục	Trung bình	Trung bình	Trung bình	Trực quan hóa, phân tích tương đồng
Autoencoder	Không	Tùy thuộc	Chậm (khi huấn luyện)	Tốt	Thấp	Giảm chiều phi tuyến, phát hiện bất thường

5 Một số lưu ý khi sử dụng t-SNE

5.1 Hướng dẫn thực hành

1. **Xử lý dữ liệu đầu vào:**
    - Chuẩn hóa dữ liệu trước khi áp dụng t-SNE
    - Loại bỏ các điểm trùng lặp (t-SNE yêu cầu các điểm là duy nhất)
    - Với tập dữ liệu lớn, có thể cân nhắc giảm chiều bằng PCA trước khi áp dụng t-SNE
2. **Điều chỉnh tham số:**
    - **Perplexity**: Thường nên từ 5-50, không vượt quá 1/3 số lượng mẫu. Perplexity có thể hiểu như "số lượng láng giềng hiệu quả" mà mỗi điểm nên xem xét.
    - **Số lần lặp (iterations)**: Thường cần 500-1000 lần lặp để hội tụ, có thể tăng thêm nếu cần.
    - **Learning rate**: Mặc định thường hoạt động tốt, nhưng có thể điều chỉnh để tránh tối ưu cục bộ.
3. **Giải thích kết quả:**
    - t-SNE chỉ bảo toàn khoảng cách tương đối giữa các điểm lân cận
    - Kích thước của các cụm không mang ý nghĩa thống kê
    - Khoảng cách giữa các cụm không nên được giải thích trực tiếp
    - Để kiểm tra mức độ phù hợp của kết quả, nên thử với các giá trị perplexity khác nhau

5.2 Mở rộng sang UMAP

UMAP (Uniform Manifold Approximation and Projection) là một phương pháp giảm chiều phi tuyến tính mới hơn t-SNE, với một số ưu điểm:

- Nhanh hơn t-SNE đáng kể
- Bảo toàn tốt hơn cả cấu trúc cục bộ và toàn cục
- Ít nhạy cảm với tham số hơn

- Có thể áp dụng cho dữ liệu mới (ánh xạ tổng quát)

```
# Cài đặt và sử dụng UMAP (Code chỉ để minh họa)
# install.packages("umap")
# library(umap)
# umap_result <- umap(mtcars_scaled)
```

6 Kết luận

t-SNE là một công cụ mạnh mẽ cho việc trực quan hóa dữ liệu đa chiều, đặc biệt phù hợp khi cần phát hiện các mẫu và cụm trong dữ liệu. Phương pháp này cho phép chúng ta:

1. **Trực quan hóa dữ liệu nhiều chiều** trong không gian 2D hoặc 3D một cách hiệu quả
2. **Phát hiện cấu trúc cục bộ** và mẫu phức tạp trong dữ liệu
3. **Khám phá các nhóm tự nhiên** mà các phương pháp tuyến tính như PCA có thể bỏ qua

Tuy nhiên, cần sử dụng t-SNE cẩn thận và hiểu rõ các hạn chế của nó:

1. **Không bảo toàn khoảng cách toàn cục**
2. **Chi phí tính toán cao** với các tập dữ liệu lớn
3. **Kết quả phụ thuộc vào tham số**
4. **Không tạo ra ánh xạ tổng quát** cho dữ liệu mới

Trong thực tế, t-SNE thường được sử dụng kết hợp với các kỹ thuật khác như PCA để đạt được hiệu quả tốt nhất. Đối với các ứng dụng đòi hỏi hiệu suất cao hoặc bảo toàn cấu trúc toàn cục tốt hơn, có thể cân nhắc sử dụng các phương pháp mới hơn như UMAP.

7 Tài liệu tham khảo

1. Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(11).
2. Wattenberg, M., Viégas, F., & Johnson, I. (2016). How to use t-SNE effectively. Distill, 1(10), e2.
3. Linderman, G. C., & Steinerberger, S. (2019). Clustering with t-SNE, provably. SIAM Journal on Mathematics of Data Science, 1(2), 313-332.
4. McInnes, L., Healy, J., & Melville, J. (2018). UMAP: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426.
5. https://distill.pub/2016/misread-tsne/