35-BaiTapTinhDiem

Bùi Minh Huy

Contents

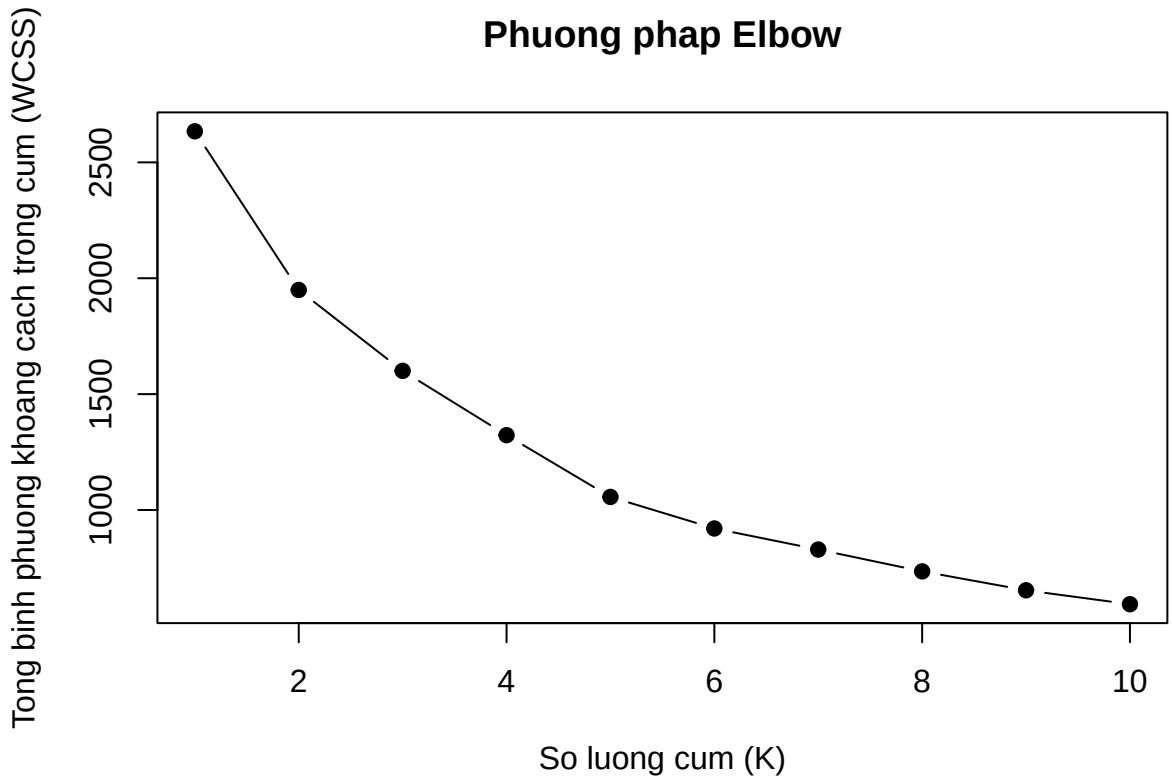
K-mean

xác định k trong K-mean

1. Elbow

```
data <- read.csv('/Users/huy/Documents/phân tích và trực quan hoá dữ liệu/Lab-10/Wholesale customers da
data_numeric <- data[, 3:8]
data_scale <- scale(data_numeric)
wcss <- numeric(10)
```

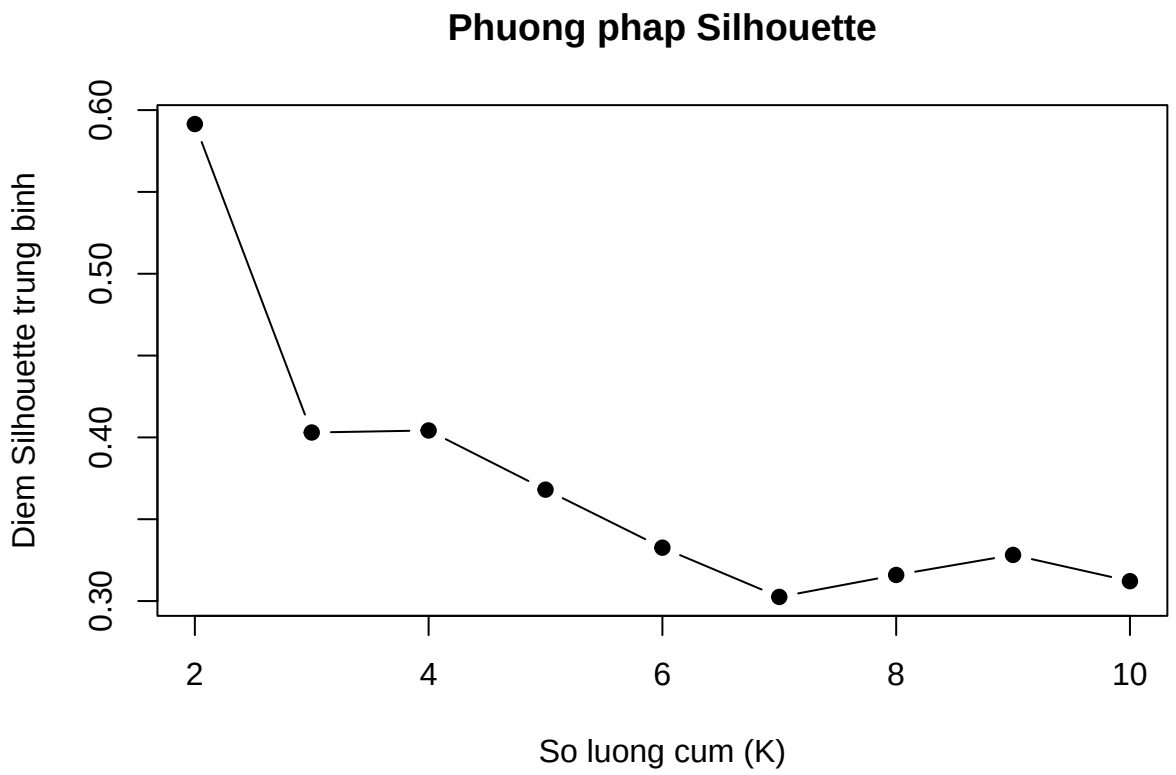
```r
for(i in 1:10) {
  kmeans_model <- kmeans(data_scale, centers=i, nstart=25)
  wcss[i]  <- kmeans_model$tot.withinss
}

# Vẽ biểu đồ Elbow
plot(1:10, wcss, type = "b", pch = 19,
     xlab = "So luong cum (K)",
     ylab = "Tong binh phuong khoang cach trong cum (WCSS)",
     main = "Phuong phap Elbow")
```



2. Silhouette

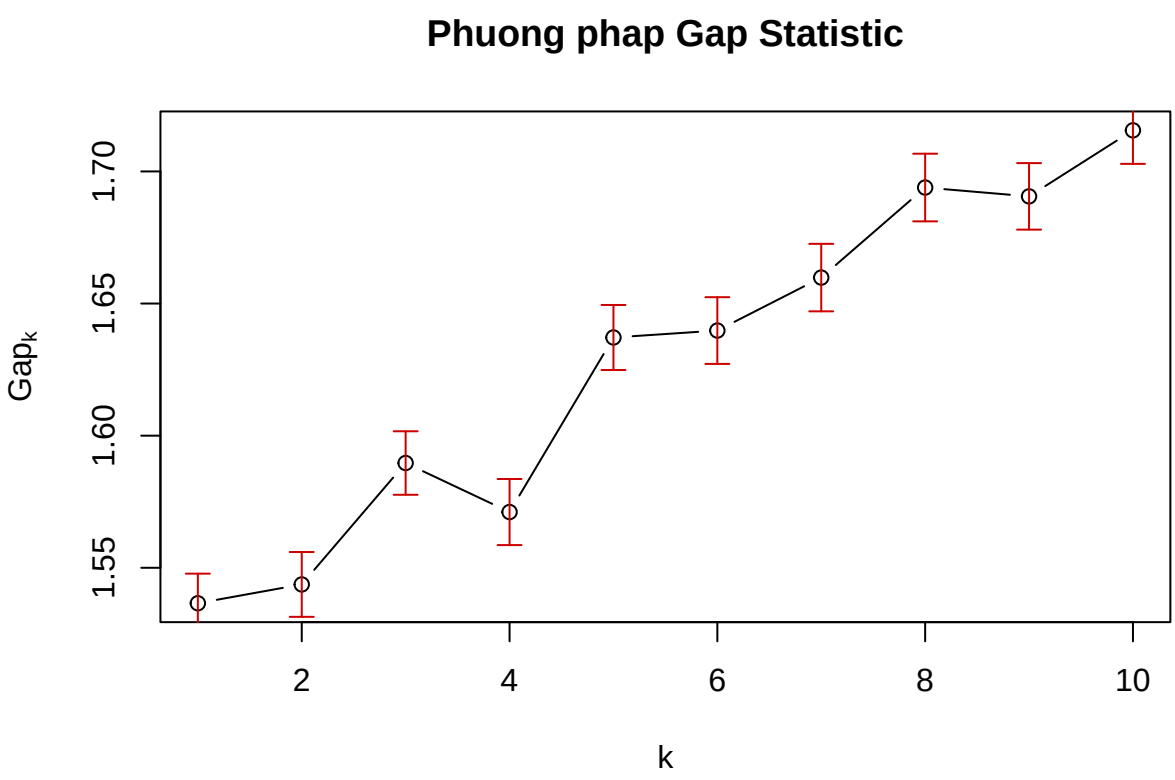
```r
# Tính điểm Silhouette trung bình cho các giá trị K từ 2 đến 10
avg_sil <- numeric(9)
for(k in 2:10) {
  km <- kmeans(data_scale, centers = k, nstart = 25)
  ss <- silhouette(km$cluster, dist(data_scale))
  avg_sil[k-1] <- mean(ss[, 3])
}

# Vẽ biểu đồ Silhouette
plot(2:10, avg_sil, type = "b", pch = 19,
```

```
     xlab = "So luong cum (K)",
     ylab = "Diem Silhouette trung binh",
     main = "Phuong phap Silhouette")
```



3. Gap Statistic

```
set.seed(123)
gap_stat <- clusGap(data_scale, FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
plot(gap_stat, main = "Phuong phap Gap Statistic")
```



4. NbClust

```
# Sử dụng NbClust để đánh giá nhiều phương pháp
# Chú ý: Kết quả này có thể mất thời gian để tính toán
library(NbClust)
nb <- NbClust(data_scale, distance = "euclidean", min.nc = 2,
              max.nc = 10, method = "kmeans")
```

```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##


## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 6 proposed 2 as the best number of clusters
```

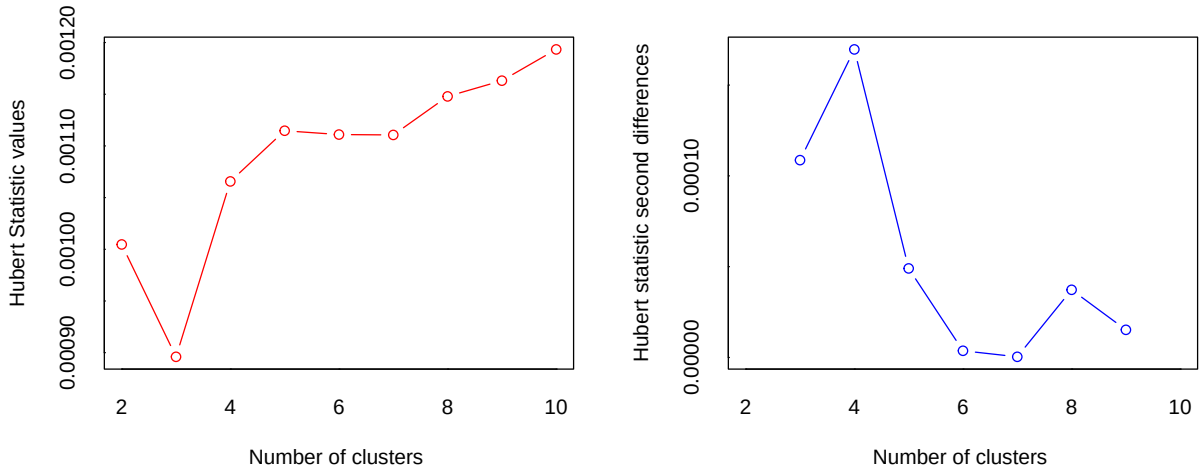


Figure 1: So sanh nhieu phuong phap xac dinh K toi uu

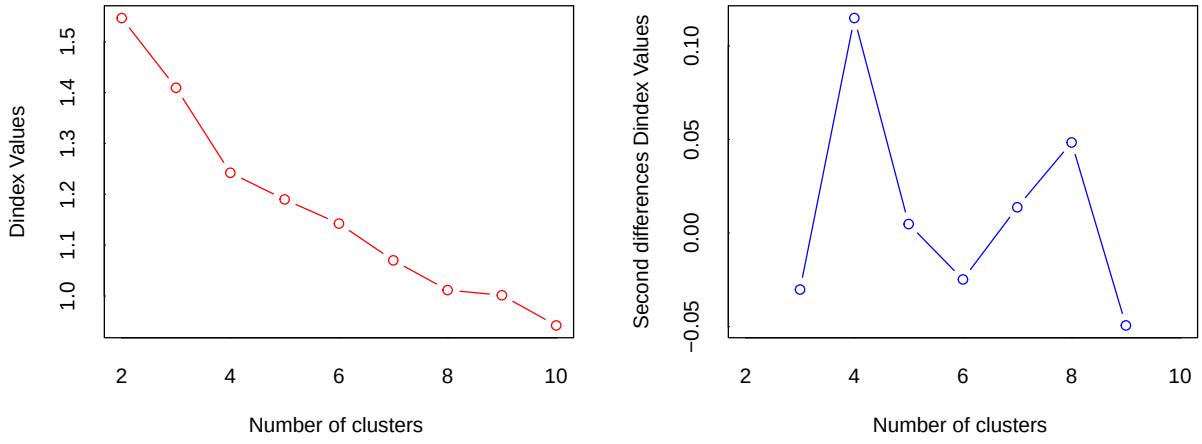


Figure 2: So sanh nhieu phuong phap xac dinh K toi uu

```
## * 4 proposed 3 as the best number of clusters
## * 5 proposed 4 as the best number of clusters
## * 2 proposed 5 as the best number of clusters
## * 2 proposed 6 as the best number of clusters
## * 1 proposed 7 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 3 proposed 10 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

```r
# Hiển thị biểu đồ tần suất của các K được đề xuất
barplot(table(nb$Best.n[1,]),
        xlab = "So luong cum",
        ylab = "So phuong phap de xuat",
        main = "So luong cum duoc de xuat boi 30 chi so")
```

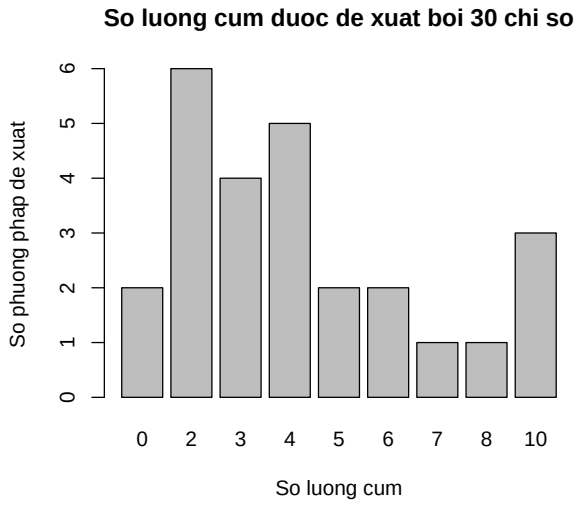


Figure 3: So sanh nhieu phuong phap xac dinh K toi uu

5. Đánh giá kết quả phân cụm với k = 3

```r
library(factoextra)
# Áp dụng K-Means với K = 3
km_result <- kmeans(data_scale, centers = 3, nstart = 100)

# Tính và vẽ biểu đồ Silhouette
sil <- silhouette(km_result$cluster, dist(data_scale))
fviz_silhouette(sil, print.summary = TRUE)
```

```
##   cluster size ave.sil.width
## 1       1   44          0.20
## 2       2    3         -0.08
## 3       3  393          0.59
```

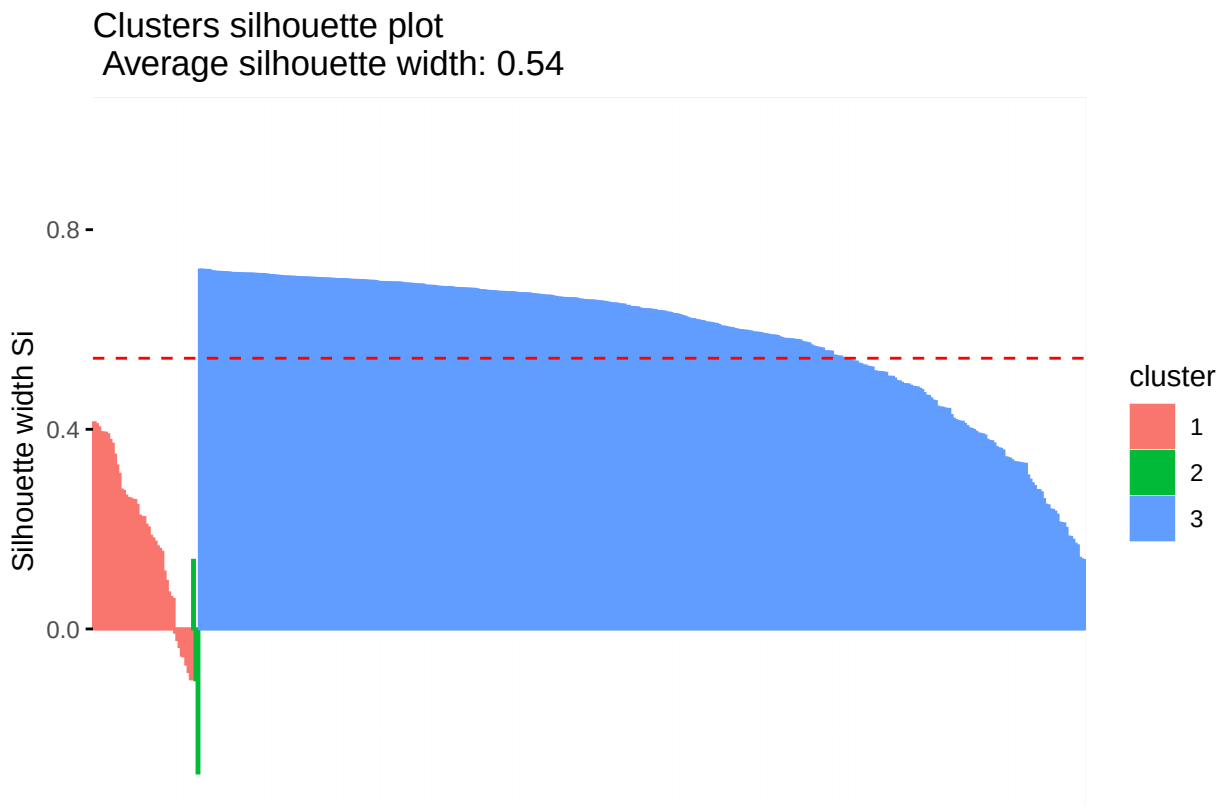
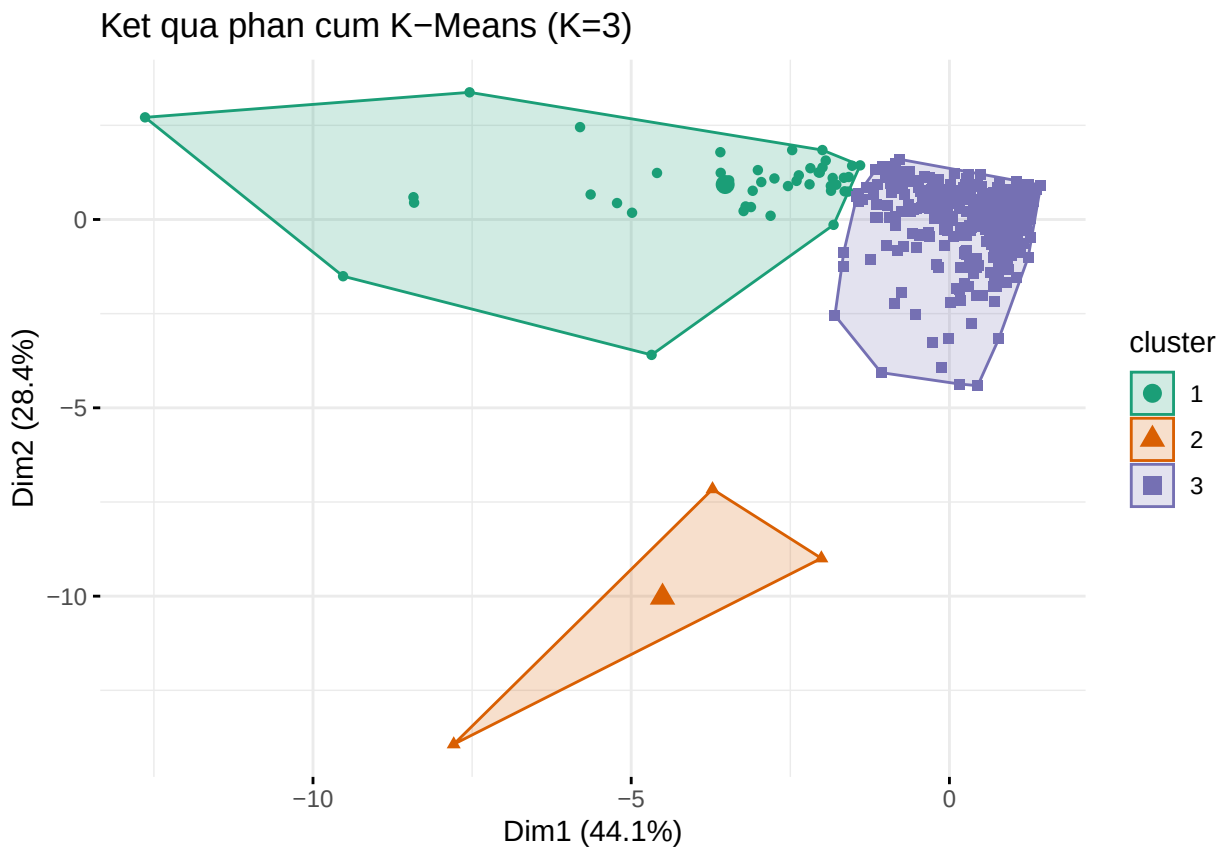


Figure 4: Bieu do Silhouette cho ket qua phan cum

6.Trực quan kết quả phân cụm với k = 3

```
fviz_cluster(km_result, data = data_scale,
             palette = c("#1B9E77", "#D95F02", "#7570B3"),
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_minimal(),
             main = "Ket qua phan cum K-Means (K=3)")
```



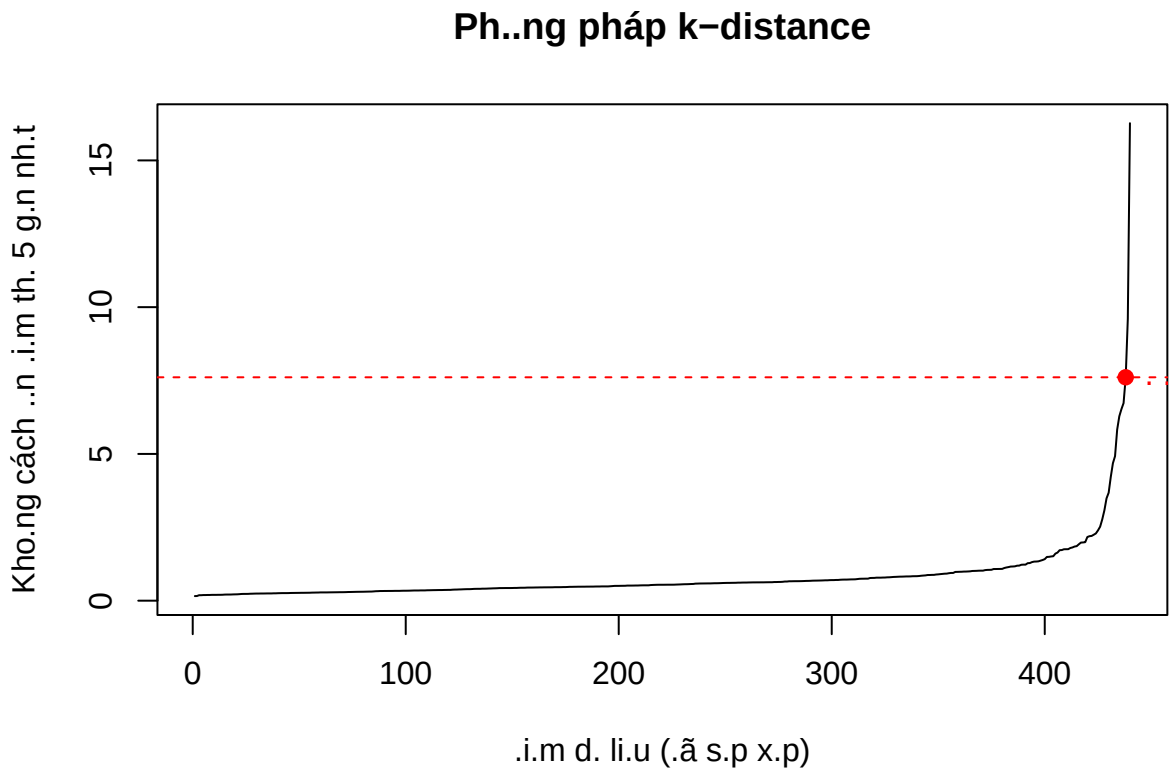
DBSCAN

xác định eps

1. Phương pháp k-distance graph

```
library(dbscan)
k <- 5
knn_dists <- kNNdist(data_scale, k = k)
# Sắp xếp khoảng cách và vẽ đồ thị
eps_candidates <- sort(knn_dists)
plot(eps_candidates, type = "l",
     xlab = "Điểm dữ liệu (đã sắp xếp)",
     ylab = paste("Khoảng cách đến điểm thứ", k, "gần nhất"),
     main = "Phương pháp k-distance")

# Tìm điểm gẩy (có thể bằng thuật toán hoặc quan sát)
# Ví dụ đơn giản: tìm điểm có độ cong lớn
eps_diff <- diff(eps_candidates, differences = 2)
eps_index <- which.max(eps_diff)
eps_value <- eps_candidates[eps_index]
# Đánh dấu điểm gẩy
points(eps_index, eps_candidates[eps_index], col = "red", pch = 19)
```

```r
text(eps_index, eps_candidates[eps_index],
     labels = paste("  ", round(eps_candidates[eps_index], 2)),
     pos = 4, col = "red")
# Vẽ đường thẳng tại điểm gẩy
abline(h = eps_candidates[eps_index], col = "red", lty = 2)
```




```r
# Tạo lưới các tham số
eps_values <- c(1.2,1.4 ,1.6,1.8)
minPts_values <- c(3, 4, 5)

# Tạo lưới plot: 3 hàng (minPts), 5 cột (eps)
par(mfrow = c(length(minPts_values), length(eps_values)))
par(mar = c(4, 4, 2, 1))  # Thu nhỏ lề cho rõ plot

# Áp dụng DBSCAN với các tổ hợp tham số
for (minPts in minPts_values) {
  for (eps in eps_values) {
    db <- dbscan::dbscan(data_scale, eps = eps, minPts = minPts)

    # Vẽ kết quả phân cụm
    plot(data_scale, col = db$cluster + 1, pch = 20, cex = 0.8,
         main = paste("  =", eps, ", MinPts =", minPts),
         xlab = "X", ylab = "Y")

    # Ghi chú số cụm và nhiễu
```

```
    n_clusters <- max(db$cluster)
    n_noise <- sum(db$cluster == 0)
    legend("topright",
           legend = c(paste("Cụm:", n_clusters),
                      paste("Nhiễu:", n_noise)),
           cex = 0.7, bty = "n")
  }
}
```

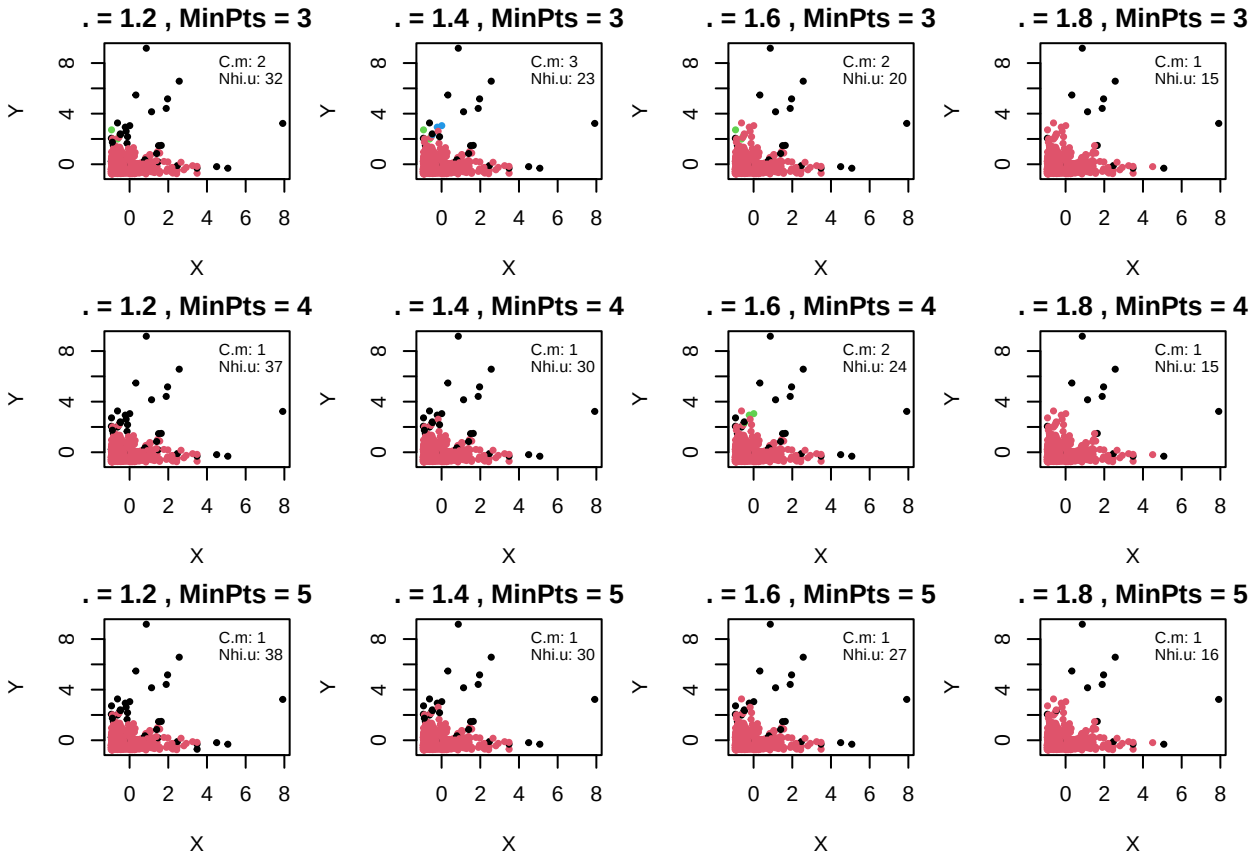


Figure 5: So sanh ket qua voi cac tham so khac nhau

2. Dùng Silhouette cho DBSCAN

```
# Áp dụng DBSCAN
library(dbscan)
db_result <- dbscan::dbscan(data_scale, eps = 1.6, minPts = 4)

# Tính chỉ số Silhouette cho các điểm KHÔNG phải nhiễu
non_noise <- which(db_result$cluster > 0)

# Kiểm tra xem có nhiều hơn 1 cụm thực sự không
if (length(unique(db_result$cluster[non_noise])) > 1) {
```

```r
  # Tính silhouette
  sil <- cluster::silhouette(db_result$cluster[non_noise],
                             dist(data_scale[non_noise, ]))

  # Vẽ biểu đồ Silhouette
  plot(sil, main = "Silhouette plot cho DBSCAN",
       col = as.numeric(factor(db_result$cluster[non_noise])))

  # Trung bình Silhouette
  avg_sil <- mean(sil[, 3])
  cat("Điểm Silhouette trung bình:", round(avg_sil, 3), "\n")

} else {
  cat("Không thể tính Silhouette: chỉ tìm thấy 1 cụm (không tính nhiễu)\n")
}
```

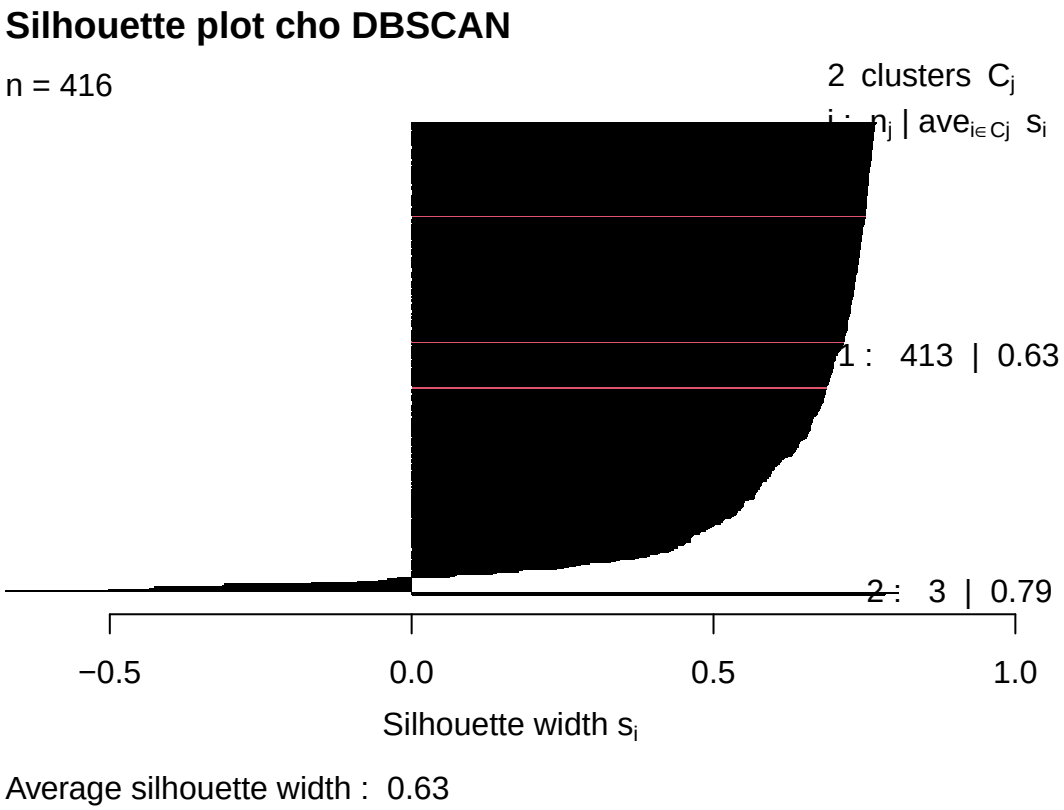


Figure 6: Chỉ số Silhouette cho DBSCAN

```
## Điểm Silhouette trung bình: 0.633
```

3. Tỷ lệ nhiễu (Noise Ratio)

```
# Tính tỷ lệ nhiễu
noise_ratio <- sum(db_result$cluster == 0) / length(db_result$cluster)
cat("Tỷ lệ nhiễu:", noise_ratio, "\n")
```

```
## Tỷ lệ nhiễu: 0.05454545
```

4. Mật độ các cụm (Cluster Density)

```
# Tính mật độ các cụm
cluster_density <- numeric(max(db_result$cluster))
for (i in 1:max(db_result$cluster)) {
  cluster_points <- data_scale[db_result$cluster == i, ]
  # Ước lượng thể tích bằng tích của phạm vi trên mỗi chiều
  volume <- prod(apply(cluster_points, 2, function(x) diff(range(x))))
  # Mật độ = số điểm / thể tích
  cluster_density[i] <- nrow(cluster_points) / volume
}

# Hiển thị mật độ các cụm
cat("Mật độ các cụm:", cluster_density, "\n")
```

```
## Mật độ các cụm: 0.1505339 112.591
```

5. Davies-Bouldin Index và Calinski-Harabasz Index

```
library(clusterCrit)
library(fpc)
# Tính Calinski-Harabasz Index (chỉ cho các điểm không phải nhiễu)
if (length(unique(db_result$cluster[non_noise])) > 1) {
  ch_index <- calinhara(data_scale[non_noise, ],
                        db_result$cluster[non_noise])
  cat("Calinski-Harabasz Index:", ch_index, "\n")
} else {
  cat("Không thể tính CH Index: chỉ tìm thấy 1 cụm (không tính nhiễu)\n")
}
```

```
## Calinski-Harabasz Index: 37.44201
```

6. Trực quan hóa kết quả DBSCAN

```
# Trực quan hóa kết quả DBSCAN
fviz_cluster(list(data = data_scale, cluster = db_result$cluster),
             palette = c("black", "red", "blue"),
             geom = "point",
```

```
            ellipse = FALSE,
            ggtheme = theme_minimal(),
            main = "Kết quả phân cụm DBSCAN")
```

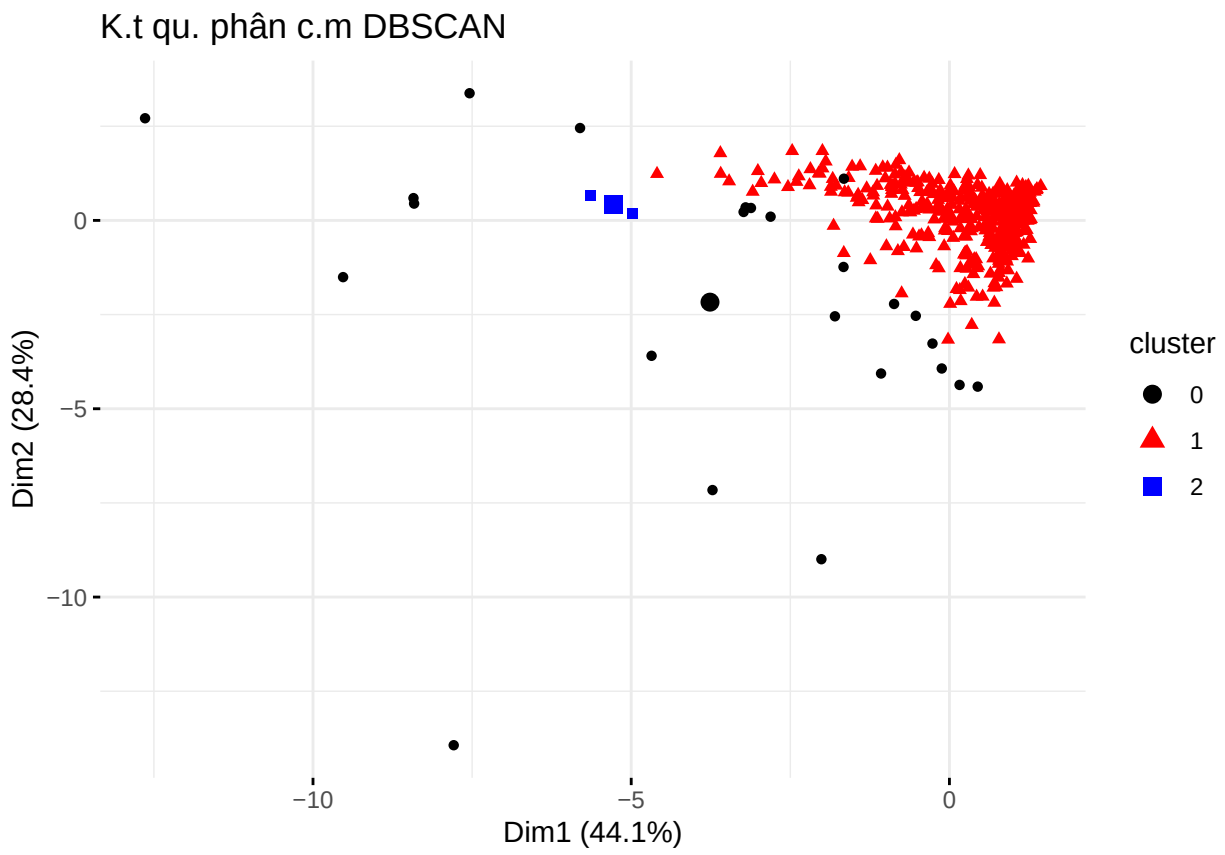


Figure 7: Ket qua phan cum DBSCAN

kết luận

- Về hiệu năng: DBSCAN cho Silhouette trung bình cao hơn → chứng tỏ các điểm trong cụm gần nhau hơn và

- Về trực quan hóa: K-Means tạo cụm tròn đẹp mắt, nhưng DBSCAN phát hiện nhiễu và các cụm có hình dạng

- Về ứng dụng:
  + Nếu dữ liệu sạch, cụm hình tròn → dùng K-Means

  + Nếu dữ liệu có nhiễu, phân bố không đều → nên dùng DBSCAN