Lab Report: Lab5 - Computational Statistics

Dhanush Kumar Reddy Narayana Reddy (dhana004), Udaya Shanker Mohanan Nair (udamo524)

2025-03-04

Introduction

Implementation of 2 Assignment questions of Computational Statistics Lab 6.

Contributions

Member: Dhanush Kumar Reddy Narayana Reddy, Liu Id: dhana004, Contribution: Report writing and coding of question 1.

Member: Udaya Shanker Mohanan Nair, Liu Id: udamo524, Contribution: Report writing and coding of question 2.

Question 1

Part A

Notation

- n: Number of observations.
- x_j: The j-th observation ($j = 1, 2, \ldots, n$).
- $K = 3$: Number of components in the mixture model.
- π_k: Mixing proportion for the k-th component ($k = 1, 2, 3$), where $\sum_{k=1}^{3} \pi_k = 1$.
- μ_k: Mean of the k-th component.
- σ_k: Standard deviation of the k-th component.
- γ_{jk}: Responsibility of the k-th component for the j-th observation.

E-Step (Expectation)

Compute the responsibilities γ_{jk}:

$$\gamma_{jk} = \frac{\pi_k \cdot \mathcal{N}(x_j \mid \mu_k, \sigma_k)}{\sum_{l=1}^{3} \pi_l \cdot \mathcal{N}(x_j \mid \mu_l, \sigma_l)}$$

where Probability Density Function (PDF) of Normal Distribution is:

$$\mathcal{N}(x_j \mid \mu_k, \sigma_k) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{(x_j - \mu_k)^2}{2\sigma_k^2}\right)$$

M-Step (Maximization)

Update the parameters π_k, μ_k, and σ_k:

1. Mixing Proportions:

$$\pi_k = \frac{1}{n}\sum_{j=1}^{n} \gamma_{jk}$$

2. Means:

$$\mu_k = \frac{\sum_{j=1}^{n} \gamma_{jk} \cdot x_j}{\sum_{j=1}^{n} \gamma_{jk}}$$

3. Variances:

$$\sigma_k^2 = \frac{\sum_{j=1}^{n} \gamma_{jk} \cdot (x_j - \mu_k)^2}{\sum_{j=1}^{n} \gamma_{jk}}$$

Standard Deviations The standard deviation for component k is computed as:

$$\sigma_k = \sqrt{\frac{\sum_{j=1}^{n} \gamma_{jk} \cdot (x_j - \mu_k)^2}{\sum_{j=1}^{n} \gamma_{jk}}}$$

Part B

Stopping Criterion

The algorithm stops when the change in the parameter vector **pv** becomes smaller than a threshold ϵ. The scale-independent stopping criterion is:

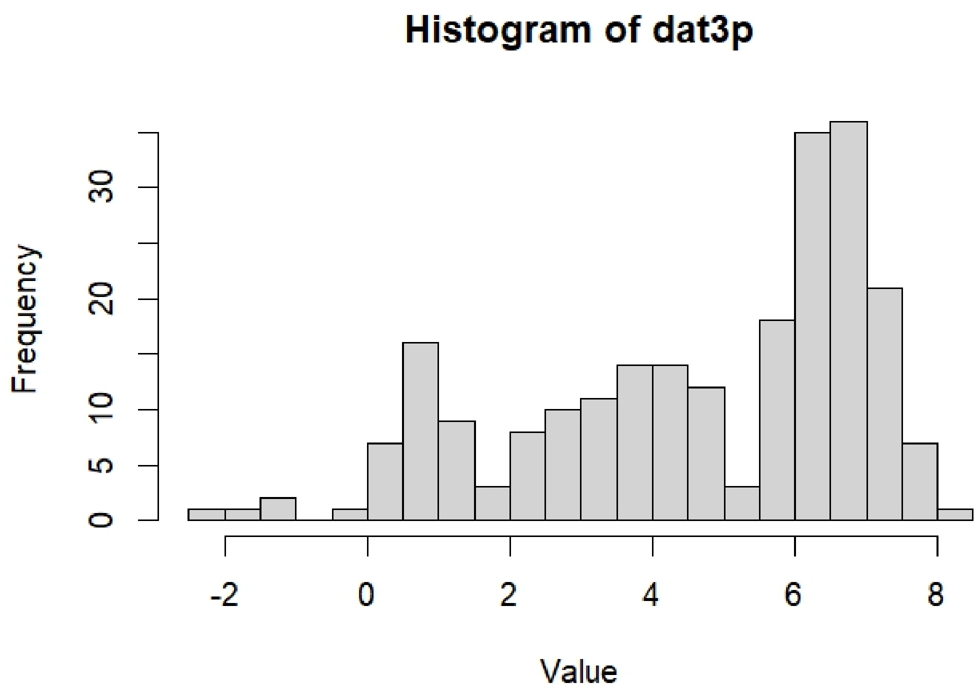
$$cc = \frac{\sqrt{\sum_{i=1}^{9} (pv_i - pv_{i,prev})^2}}{\sqrt{\sum_{i=1}^{9} pv_{i,prev}^2}} < \epsilon$$

Where:

- pv_i: Current value of the i-th parameter.
- $pv_{i,prev}$: Previous value of the i-th parameter.

This stopping Criterion is scale-independent, ensuring consistent behavior across different datasets.It measures relative change, making it sensitive to changes in all parameters, regardless of their magnitude.It is simple and computationally efficient to implement.

Part C



Summary of Model Parameters:

Parameter
Mixing Proportion
Mean
Standard Deviation

The final parameter estimates correspond to the estimated parameters of the three-component normal mixture model, specifically:

Mixing Proportions:

These are the first three values: 0.2268454, 0.2482586, 0.524896. They represent the proportions of the data belonging to each of the three normal distributions.

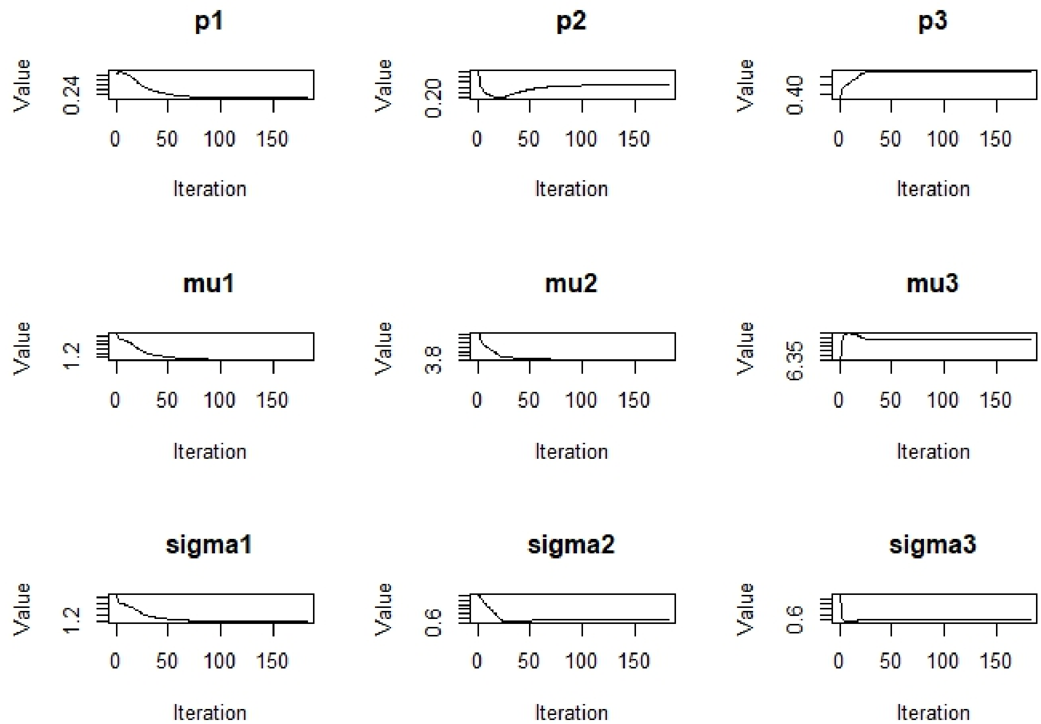
Means:

These are the next three values: 1.072835, 3.831919, 6.581551. They represent the estimated means of the three normal distributions.

Standard Deviations:

These are the last three values: 1.198011, 0.717409, 0.6007124 . They represent the estimated standard deviations of the three normal distributions.

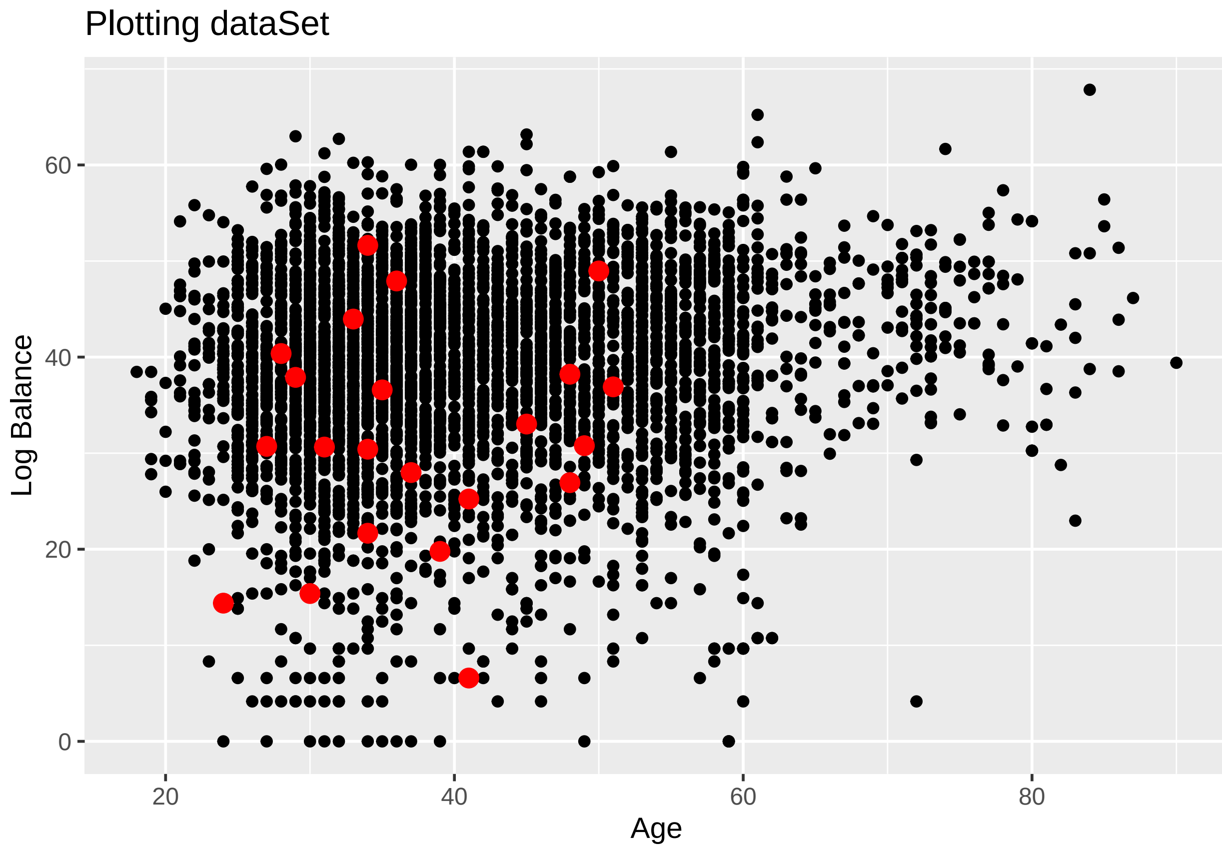
Part D



The plots illustrate the evolution of each parameter estimate across iterations, indicating convergence. Specifically, the estimates for the mixing proportions (p1, p2, p3) stabilize after approximately 50–100 iterations, with only minor fluctuations in p2. The means (mu1, mu2, mu3) also show early fluctuations but generally stabilize around iterations 50–100, with mu3 remaining stable throughout, suggesting it reached convergence early. The standard deviations (sigma1, sigma2, sigma3) demonstrate similar behavior, with sigma1 decreasing but stabilizing after about 50 iterations and sigma2 and sigma3 reaching stability early. Overall, the plots suggest that all parameters have converged, as they exhibit stability and minimal fluctuations after a certain number of iterations.

Question 2

Given a dataset containing 4364 clients, Now we are plotting this data for the fields age and Log Balance of randomly selected 22 clients.

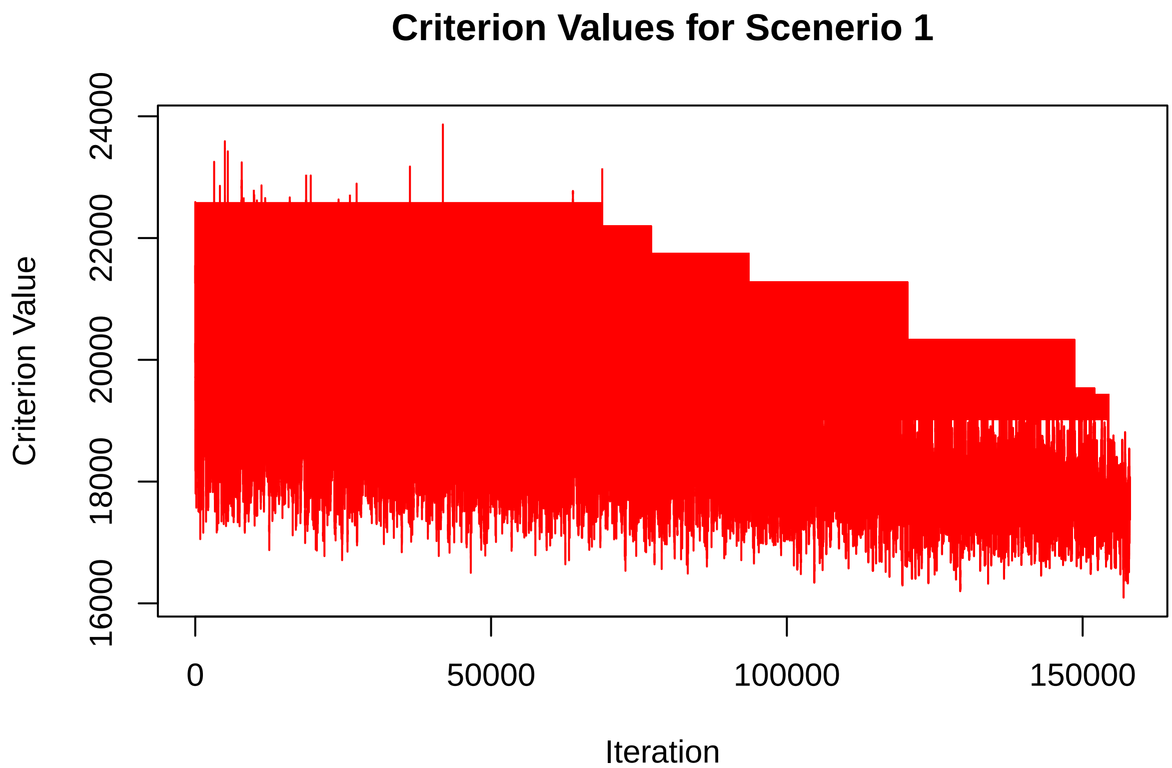


Now we are gone implement simulated annealing algorithm to minimize the criterion function where in each iteration we are exchanging clients from the remaining clients to identify better results.

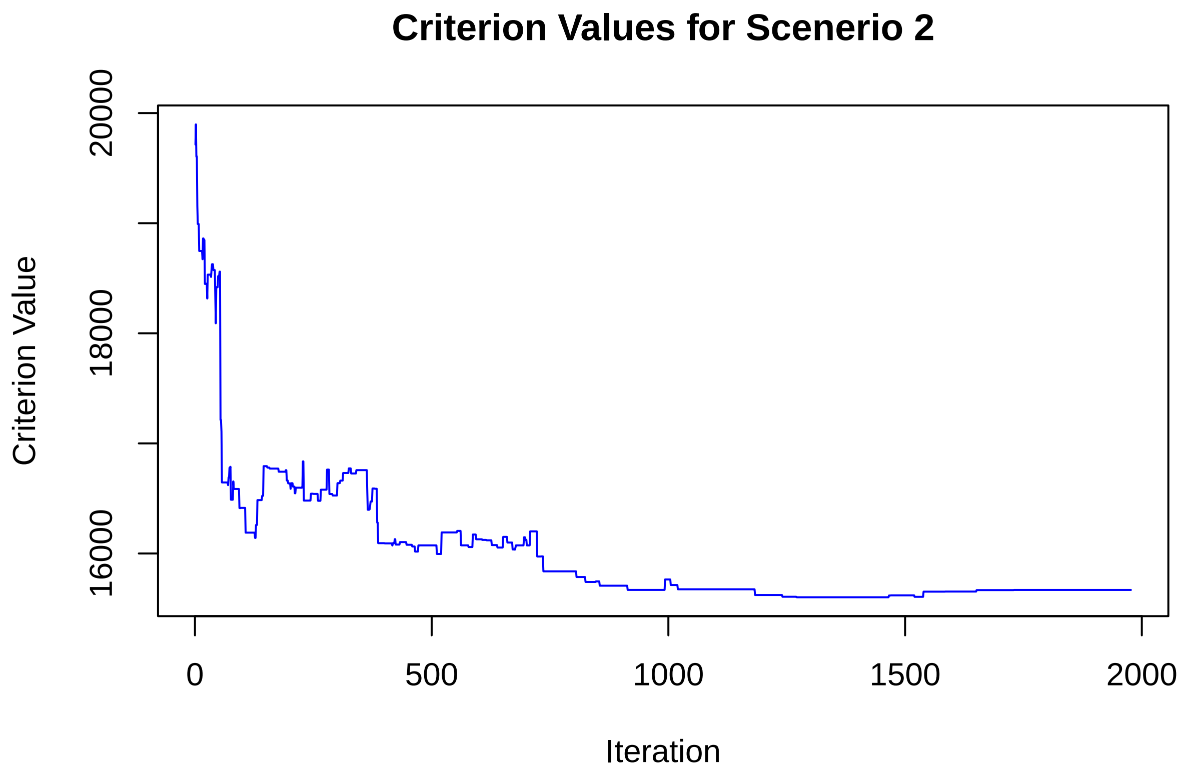
Now we gone to test two scenarios for the function

Scenario1 -> Initial Temperature = 1000 -> Cooling Rate = 0.9 -> beta = 1 -> Number of iteration = 1000

Scenario2 -> Initial Temperature = 100 -> Cooling Rate = 0.7 -> beta = 1.5 -> Number of iteration = 150



Plot shows that the criterion values are having high fluctuations in the start and then give a good solution to the end.



From this plot, it is clear that better starting values when compared with previous plot but it seems to be incomplete thus it is difficult to analyze the plot.

Appendix

Question 1

```
# Question: EM Algorithm
library(ggplot2)

# Part A
# EM Algorithm for 3 components
emalg_3_comp <- function(dat, eps = 0.000001) {
  n <- length(dat)
  pi1 <- rep(NA, n)  # initialize vector for prob. to belong to group 1
  pi2 <- rep(NA, n)  # initialize vector for prob. to belong to group 2
  pi3 <- rep(NA, n)  # initialize vector for prob. to belong to group 3

  # Define reasonable starting values for parameters
  p1 <- 1/3          # starting value for mixing parameter of group 1
  p2 <- 1/3          # starting value for mixing parameter of group 2
  p3 <- 1/3          # starting value for mixing parameter of group 3
  sigma1 <- sd(dat) * 2/3  # starting value for standard deviation in group 1
```

```r
sigma2 <- sigma1         # starting value for standard deviation in group 2
sigma3 <- sigma1         # starting value for standard deviation in group 3
mu1 <- mean(dat) - sigma1  # starting value for mean of group 1
mu2 <- mean(dat)           # starting value for mean of group 2
mu3 <- mean(dat) + sigma1  # starting value for mean of group 3
pv <- c(p1, p2, p3, mu1, mu2, mu3, sigma1, sigma2, sigma3)  # parameter vector

# Store parameter paras
para <- list(p1 = c(), p2 = c(), p3 = c(), mu1 = c(), mu2 = c(), mu3 = c(), sigma1 = c(), sigma2 = c(

cc <- eps + 100  # initialize convergence criterion
while (cc > eps) {
  pv1 <- pv  # save previous parameter vector

  ### E step ###
  for (j in 1:n) {
    pi1_j <- p1 * dnorm(dat[j], mean = mu1, sd = sigma1)
    pi2_j <- p2 * dnorm(dat[j], mean = mu2, sd = sigma2)
    pi3_j <- p3 * dnorm(dat[j], mean = mu3, sd = sigma3)
    total <- pi1_j + pi2_j + pi3_j
    pi1[j] <- pi1_j / total
    pi2[j] <- pi2_j / total
    pi3[j] <- pi3_j / total
  }

  ### M step ###
  p1 <- mean(pi1)
  p2 <- mean(pi2)
  p3 <- mean(pi3)
  mu1 <- sum(pi1 * dat) / (p1 * n)
  mu2 <- sum(pi2 * dat) / (p2 * n)
  mu3 <- sum(pi3 * dat) / (p3 * n)
  sigma1 <- sqrt(sum(pi1 * (dat - mu1)^2) / (p1 * n))
  sigma2 <- sqrt(sum(pi2 * (dat - mu2)^2) / (p2 * n))
  sigma3 <- sqrt(sum(pi3 * (dat - mu3)^2) / (p3 * n))

  pv <- c(p1, p2, p3, mu1, mu2, mu3, sigma1, sigma2, sigma3)

  # Part B
  cc <- sqrt(sum((pv - pv1)^2)) / sqrt(sum(pv1^2))  # scale-independent convergence criterion

  # Storing parameter values
  para$p1 <- c(para$p1, p1)
  para$p2 <- c(para$p2, p2)
  para$p3 <- c(para$p3, p3)
  para$mu1 <- c(para$mu1, mu1)
  para$mu2 <- c(para$mu2, mu2)
  para$mu3 <- c(para$mu3, mu3)
  para$sigma1 <- c(para$sigma1, sigma1)
  para$sigma2 <- c(para$sigma2, sigma2)
  para$sigma3 <- c(para$sigma3, sigma3)
}
```

```r
  # Part D
  par(mfrow = c(3, 3))
  for (param in names(para)) {
    plot(para[[param]], type = "l", main = param, xlab = "Iteration", ylab = "Value")
  }
  pv
}

# Data
load("C:/Users/dhanu/OneDrive/Documents/Computational Stats/Lab 6/threepops.Rdata")

# Part C
# Histogram
hist(dat3p, breaks = 20, main = "Histogram of dat3p", xlab = "Value")

# Fit the mixture model
results <- emalg_3_comp(dat3p)
cat("Final parameter estimates: \n",results)
```

Question 2

```r
#part a
# Load necessary libraries
library(ggplot2)
load("bankdata.Rdata")

set.seed(123)
len <- nrow(bankdata)
dataset <- bankdata[sample(len, 22), ]

ggplot(bankdata, aes(x = age, y = balance)) +
  geom_point() +
  geom_point(data = dataset, color = "red", size = 3) +
  labs(title = "Plotting dataSet", x = "Age", y = "Log Balance")

#part b
source("bankcrit.r")
annealing_fn <- function(data, initial_subset, initial_temperature, cooling_rate, beta, iteration) {
  present_indices <- sample(1:nrow(data), 22)
  current_temp <- initial_temperature
  criterion_values <- numeric()

  while (current_temp > 0.206 * initial_temperature) {
    for (i in 1:iteration) {
      # 1. Generate a new candidate subset by swapping one element
      temp_indices <- present_indices
      select_index <- sample(1:22, 1)
      remaining_indices <- setdiff(1:nrow(data), temp_indices)
      new_index <- sample(remaining_indices, 1)
      temp_indices[select_index] <- new_index
```

```
      # 2. Compute acceptance probability
      crit_value1 <- crit(bankdata, present_indices)
      crit_value2 <- crit(bankdata, temp_indices)
      h_value <- exp((crit_value1 - crit_value2) / current_temp)

      # 3. Accept new subset based on probability
      if (runif(1) < min(h_value, 1)) {
        present_indices <- temp_indices
      }
      criterion_values <- c(criterion_values, crit_value1)
    }
    # 4. Update temperature and iteration parameters
    current_temp <- cooling_rate * current_temp
    iteration <- beta * iteration
  }
  return(list(selected_dataset = data[present_indices, ], criterion_values = criterion_values))
}

#part c

res_1 <- annealing_fn(bankdata, dataset, initial_temperature = 1000,
                      cooling_rate = 0.99, beta = 1, iteration = 1000)
plot(res_1$criterion_values, type = "l", col = "red",
     xlab = "Iteration", ylab = "Criterion Value", main = "Criterion Values for Scenerio 1")

res_2 <- annealing_fn(bankdata, dataset, initial_temperature = 100,
                      cooling_rate = 0.7, beta = 1.5, iteration = 150)
plot(res_2$criterion_values, type = "l", col = "blue",
     xlab = "Iteration", ylab = "Criterion Value", main = "Criterion Values for Scenerio 2")
```