test

Huanyu Chen

2024-04-22

```
# Load necessary libraries
library(mclust)
```

```
## Warning: package 'mclust' was built under R version 4.3.2
```

```
## Package 'mclust' version 6.1
## Type 'citation("mclust")' for citing this R package in publications.
```

```
library(tidyverse)
```

```
## Warning: package 'tidyr' was built under R version 4.3.2
```

```
## -- 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   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     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()
## x purrr::map()    masks mclust::map()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
# Read the data
df <- read.csv("./data/diabetes.csv") |>
  janitor::clean_names() |>
  mutate(across(c(glucose, blood_pressure, skin_thickness, insulin, bmi), ~na_if(.x, 0)))

# Handle missing values
df[is.na(df)] <- 0

# Extract features
X <- df[, c("pregnancies", "glucose", "blood_pressure", "skin_thickness", "insulin", "bmi",
            "diabetes_pedigree_function", "age")]

# Define the EM algorithm function
EM_algorithm <- function(X, num_clusters, max_iter = 100, tol = 1e-6) {
```

```r
  # Initialize parameters
  model <- Mclust(X, G = num_clusters)

  for (iter in 1:max_iter) {
    # E-step
    responsibilities <- matrix(0, nrow = nrow(X), ncol = num_clusters)
    for (k in 1:num_clusters) {
      responsibilities[, k] <- model$z[, k] * model$parameters$pro[k]
    }
    responsibilities <- responsibilities / rowSums(responsibilities)

    # M-step
    model <- Mclust(X, G = num_clusters, z = responsibilities)

    # Calculate log-likelihood and check for convergence
    log_likelihood <- sum(log(apply(model$z * model$parameters$pro, 1, sum)))
    if (iter > 1 && abs(log_likelihood - prev_log_likelihood) < tol) {
      break
    }
    prev_log_likelihood <- log_likelihood
  }

  return(model)
}

# Call the EM algorithm
num_clusters <- 3  # Set the number of clusters
gmm_model <- EM_algorithm(X, num_clusters)

# Get the cluster assignments for each sample
clusters <- predict(gmm_model)

# View the feature patterns of the clusters
cluster_means <- t(apply(gmm_model$parameters$mean, 2, function(x) round(x, 2)))
print(cluster_means)
```

```
##      pregnancies glucose blood_pressure skin_thickness insulin   bmi
## [1,]        4.25  133.59          62.55          25.76  137.72 32.63
## [2,]        3.11  112.17          69.86          28.18   88.94 31.97
## [3,]        4.86  124.39          74.63           0.00    0.00 31.35
##      diabetes_pedigree_function   age
## [1,]                       0.68 38.07
## [2,]                       0.40 28.05
## [3,]                       0.40 38.23
```

```r
# View the mixing weights of the clusters
mixing_weights <- round(gmm_model$parameters$pro, 2)
print(mixing_weights)
```

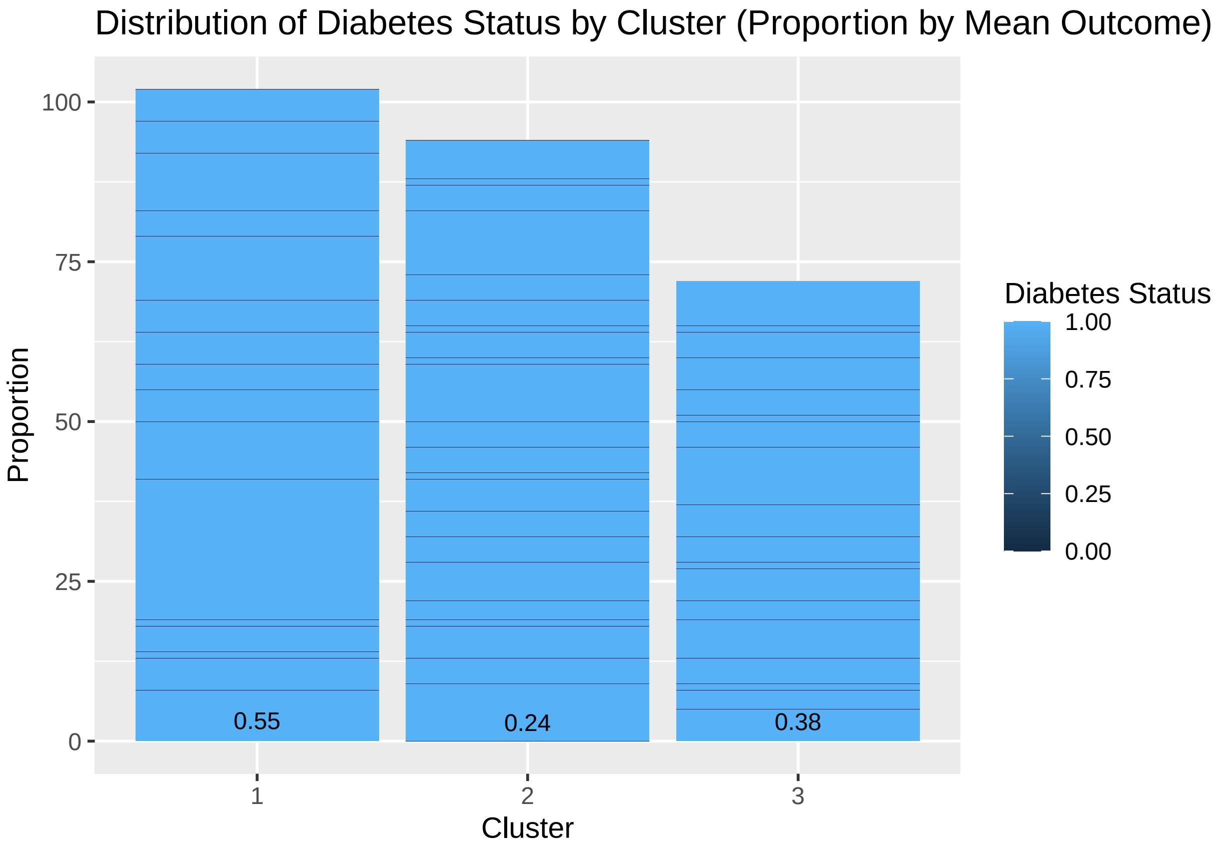
```
## [1] 0.27 0.49 0.25
```

```r
# Further analysis can be performed based on the cluster assignments,
# for example, analyzing the relationship between cluster assignments and diabetes status
```

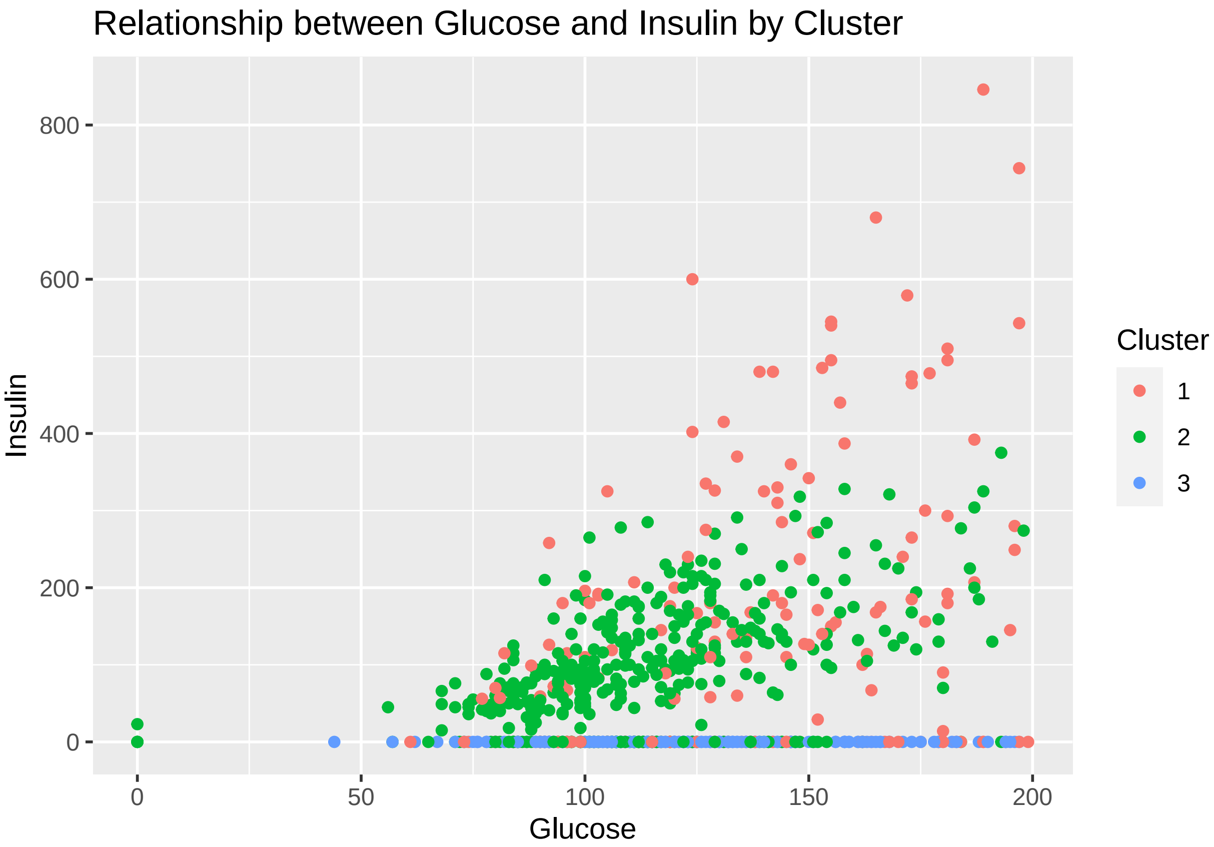
```r
library(ggplot2)

# Ensure the cluster and diabetes_status variables are factors
df$cluster <- factor(clusters$classification)
# Calculate the proportion of outcome (mean) within each cluster
cluster_outcome_mean <- aggregate(outcome ~ cluster, data = df, FUN = mean)

# Plot the distribution of diabetes status within each cluster
ggplot(df, aes(x = cluster, y = outcome, fill = outcome)) +
  geom_bar(stat = "identity") +
  geom_text(data = cluster_outcome_mean, aes(label = sprintf("%.2f", outcome)), vjust = -0.5, color = "
  labs(x = "Cluster", y = "Proportion", fill = "Diabetes Status") +
  ggtitle("Distribution of Diabetes Status by Cluster (Proportion by Mean Outcome)")
```



```r
# Plot the relationship between glucose and insulin, colored by cluster
ggplot(df, aes(x = glucose, y = insulin, color = factor(cluster))) +
  geom_point() +
  labs(x = "Glucose", y = "Insulin", color = "Cluster") +
  ggtitle("Relationship between Glucose and Insulin by Cluster")
```


Relationship between Glucose and Insulin by Cluster
Insulin
800
600
400
200
0
0
50
100
150
200
Glucose
Cluster
● 1
● 2
● 3