Unsupervised ML

# Libraries

library(tidyverse)

── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
✔ dplyr 1.1.4 ✔ readr 2.1.5  
✔ forcats 1.0.0 ✔ stringr 1.5.1  
✔ ggplot2 3.5.1 ✔ tibble 3.2.1  
✔ lubridate 1.9.3 ✔ tidyr 1.3.1  
✔ purrr 1.0.2   
── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
✖ dplyr::filter() masks stats::filter()  
✖ dplyr::lag() masks stats::lag()  
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(GGally)

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

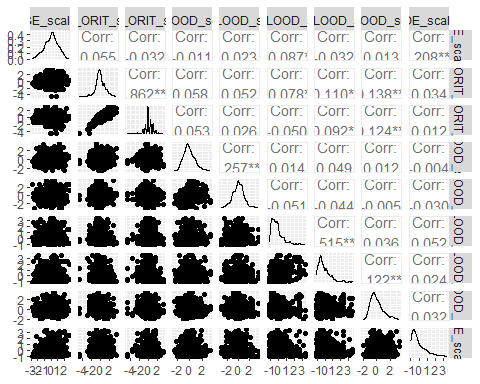
# Read data

data\_to\_model <- readRDS(file = "processed\_data/data\_to\_model.rds")  
  
data\_to\_model |>   
 select\_if(is.numeric) |>   
 summary()

ID AGE S\_AD\_ORIT D\_AD\_ORIT   
 Min. : 2.0 Min. :30.00 Min. : 0.0 Min. : 0.0   
 1st Qu.: 552.5 1st Qu.:54.00 1st Qu.:120.0 1st Qu.: 80.0   
 Median : 901.0 Median :63.00 Median :130.0 Median : 80.0   
 Mean : 907.4 Mean :61.93 Mean :135.4 Mean : 83.6   
 3rd Qu.:1255.8 3rd Qu.:70.00 3rd Qu.:150.0 3rd Qu.: 90.0   
 Max. :1699.0 Max. :92.00 Max. :240.0 Max. :140.0   
 K\_BLOOD NA\_BLOOD ALT\_BLOOD AST\_BLOOD   
 Min. :2.300 Min. :117.0 Min. :0.0300 Min. :0.0400   
 1st Qu.:3.700 1st Qu.:132.0 1st Qu.:0.2300 1st Qu.:0.1500   
 Median :4.000 Median :136.0 Median :0.3800 Median :0.2200   
 Mean :4.132 Mean :136.2 Mean :0.4471 Mean :0.2516   
 3rd Qu.:4.500 3rd Qu.:140.0 3rd Qu.:0.5200 3rd Qu.:0.3000   
 Max. :6.800 Max. :159.0 Max. :1.8100 Max. :0.9800   
 L\_BLOOD ROE AGE\_scaled.V1 S\_AD\_ORIT\_scaled.V1  
 Min. : 2.000 Min. : 2.00 Min. :-2.7959370 Min. :-4.485726   
 1st Qu.: 6.500 1st Qu.: 5.00 1st Qu.:-0.6790141 1st Qu.:-0.497978   
 Median : 8.000 Median :10.00 Median : 0.1148320 Median :-0.165666   
 Mean : 8.712 Mean :13.11 Mean : 0.0208516 Mean : 0.012161   
 3rd Qu.:10.400 3rd Qu.:18.00 3rd Qu.: 0.7322679 3rd Qu.: 0.498959   
 Max. :20.500 Max. :53.00 Max. : 2.6727806 Max. : 3.489770   
 D\_AD\_ORIT\_scaled.V1 K\_BLOOD\_scaled.V1 NA\_BLOOD\_scaled.V1   
 Min. :-4.787634 Min. :-2.498009 Min. :-2.989837   
 1st Qu.:-0.179603 1st Qu.:-0.649894 1st Qu.:-0.689592   
 Median :-0.179603 Median :-0.253870 Median :-0.076193   
 Mean : 0.027484 Mean :-0.079273 Mean :-0.049722   
 3rd Qu.: 0.396401 3rd Qu.: 0.406171 3rd Qu.: 0.537206   
 Max. : 3.276421 Max. : 3.442359 Max. : 3.450851   
 ALT\_BLOOD\_scaled.V1 AST\_BLOOD\_scaled.V1 L\_BLOOD\_scaled.V1   
 Min. :-1.167211 Min. :-1.104292 Min. :-1.991748   
 1st Qu.:-0.651990 1st Qu.:-0.566084 1st Qu.:-0.671166   
 Median :-0.265574 Median :-0.223588 Median :-0.230972   
 Mean :-0.092760 Mean :-0.068999 Mean :-0.022093   
 3rd Qu.: 0.095081 3rd Qu.: 0.167836 3rd Qu.: 0.473338   
 Max. : 3.418257 Max. : 3.494941 Max. : 3.437310   
 ROE\_scaled.V1   
 Min. :-1.002119   
 1st Qu.:-0.739266   
 Median :-0.301178   
 Mean :-0.028729   
 3rd Qu.: 0.399763   
 Max. : 3.466381

data\_to\_model |>   
 select(ends\_with("scaled")) |>   
 as.matrix() -> numerics\_scaled

numerics\_scaled |>   
 ggpairs()

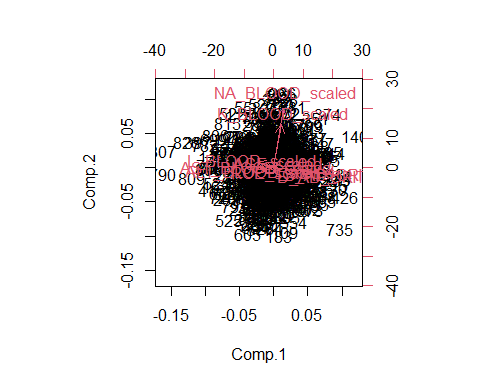


# Principal components analysis (PCA)

# Perform PCA using princomp  
pca\_results <- princomp(numerics\_scaled)  
summary(pca\_results)

Importance of components:  
 Comp.1 Comp.2 Comp.3 Comp.4 Comp.5  
Standard deviation 1.3561971 1.0853991 1.0765832 1.0232343 0.9002594  
Proportion of Variance 0.2345209 0.1502155 0.1477852 0.1335015 0.1033406  
Cumulative Proportion 0.2345209 0.3847364 0.5325216 0.6660231 0.7693638  
 Comp.6 Comp.7 Comp.8 Comp.9  
Standard deviation 0.83100444 0.81803154 0.56865267 0.35453481  
Proportion of Variance 0.08805268 0.08532493 0.04123159 0.01602705  
Cumulative Proportion 0.85741643 0.94274136 0.98397295 1.00000000

biplot(pca\_results)



## Explained variance and scree plot

To decide how many principal components to retain, you can use several criteria. Some commonly used methods:

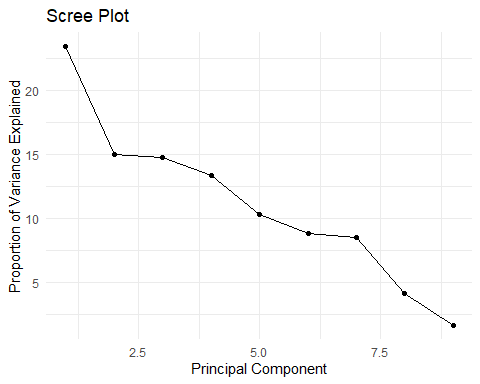
### Elbow Method

Look for an “elbow” in the scree plot, where the explained variance starts to level off. The principal components to the left of this elbow are typically considered important.

# Calculate eigenvalues  
eigenvalues <- pca\_results$sdev^2  
explained\_variance <- eigenvalues / sum(eigenvalues) \* 100  
explained\_variance

Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8   
23.452088 15.021551 14.778524 13.350147 10.334065 8.805268 8.532493 4.123159   
 Comp.9   
 1.602705

# Create a data frame for plotting  
scree\_data <- data.frame(  
 Principal\_Component = seq\_along(explained\_variance),  
 explained\_variance = explained\_variance  
)  
  
# Create the scree plot using ggplot2  
ggplot(scree\_data, aes(x = Principal\_Component, y = explained\_variance)) +  
 geom\_line() +  
 geom\_point() +  
 labs(title = "Scree Plot",  
 x = "Principal Component",  
 y = "Proportion of Variance Explained") +  
 theme\_minimal()



7 seems ok with the scree plot

### Kaiser Criterion

Keep all principal components with eigenvalues (variance explained) greater than 1. This criterion is based on the idea that a principal component should explain at least as much variance as one of the original variables.

# Print eigenvalues  
print(eigenvalues)

Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8   
1.8392707 1.1780912 1.1590314 1.0470085 0.8104670 0.6905684 0.6691756 0.3233659   
 Comp.9   
0.1256949

We would use 4

### Proportion of Variance Explained

Choose enough components to explain a high cumulative proportion of the variance (e.g., 80% or 90%).

cumsum(explained\_variance)

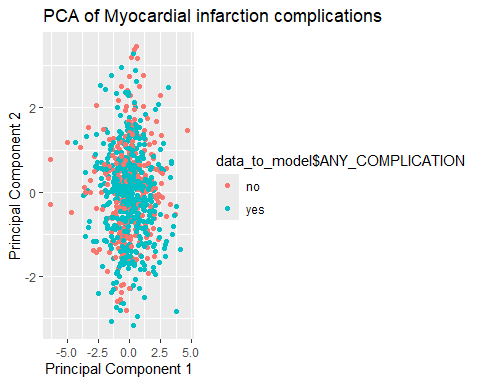
Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8   
 23.45209 38.47364 53.25216 66.60231 76.93638 85.74164 94.27414 98.39729   
 Comp.9   
100.00000

6 or 7

## Ploting the scores

num\_scores\_retained <- 7  
pc\_scores <- pca\_results$scores[, 1:num\_scores\_retained]  
# Select PCs based on explained variance (optional)

# Plot the first two principal components  
pc\_scores |>   
 as.data.frame() |>   
 ggplot( aes(x = Comp.1, y = Comp.2)) +  
 geom\_point(aes(color = data\_to\_model$ANY\_COMPLICATION)) +  
 labs(  
 title = "PCA of Myocardial infarction complications",  
 x = "Principal Component 1",  
 y = "Principal Component 2"  
 )



# Principal components clustering

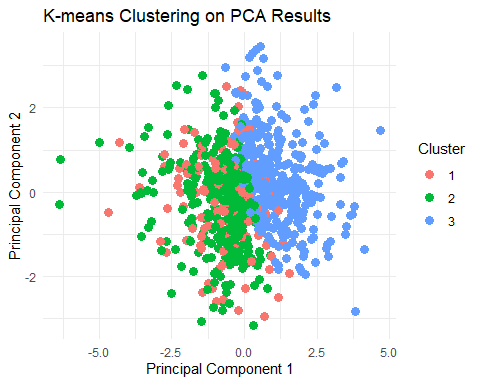
Principal components clustering is simply clustering the data based on M ≤ p principal components. Remeber we do not have to, and that we can use the original data. Here we are first reducing the data with PCA and then applying the clustering technique.

## K-means algorithm

# Set the number of clusters  
num\_clusters <- 3 # Example: 3 clusters  
  
# Perform k-means clustering  
set.seed(123) # Set seed for reproducibility  
kmeans\_result <- kmeans(pc\_scores, centers = num\_clusters)  
  
# Print k-means result  
print(kmeans\_result)

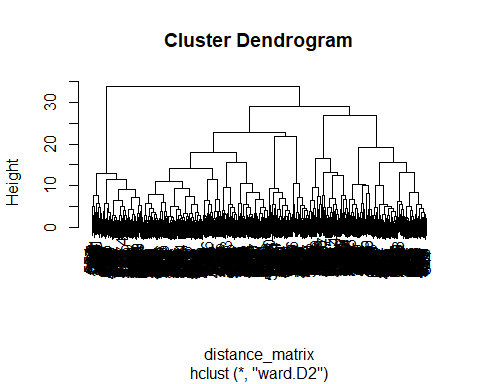
K-means clustering with 3 clusters of sizes 150, 348, 342  
  
Cluster means:  
 Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6  
1 -0.6730506 -0.1780095 -0.7478080 1.37490450 0.464558890 0.14856410  
2 -0.8376720 -0.2617708 0.1263797 -0.52244020 -0.207787823 -0.02597803  
3 1.1475657 0.3444376 0.1993891 -0.07142247 0.007679325 -0.03872591  
 Comp.7  
1 0.057371563  
2 0.004653474  
3 -0.029898080  
  
Clustering vector:  
 [1] 2 2 1 1 3 3 3 2 3 2 1 3 1 1 3 2 2 3 2 2 3 2 2 1 3 1 3 2 1 2 1 2 2 2 3 2 3  
 [38] 2 1 3 1 2 2 2 3 2 3 2 2 3 3 3 3 2 1 3 2 2 2 3 3 2 3 3 2 3 2 2 2 3 2 3 1 1  
 [75] 3 2 2 3 1 1 3 2 1 3 1 2 1 3 3 3 3 3 1 2 2 3 3 1 1 2 1 3 3 2 1 2 3 2 1 3 1  
[112] 3 3 2 2 1 2 3 3 3 1 3 2 3 2 3 2 3 2 3 2 3 3 3 3 2 2 3 3 3 1 1 3 3 1 2 1 3  
[149] 3 3 3 3 3 3 2 2 3 1 1 2 2 2 1 2 2 1 2 1 1 3 2 3 2 3 1 2 2 2 3 1 3 2 2 3 3  
[186] 3 3 2 1 3 2 3 2 1 2 3 3 3 3 3 2 3 3 3 3 2 3 2 3 2 3 3 1 3 2 2 1 3 2 2 2 2  
[223] 2 3 1 1 3 1 1 3 2 3 1 2 2 2 1 2 3 3 3 2 3 1 2 2 1 3 3 3 3 2 1 1 1 2 1 3 3  
[260] 2 2 2 1 1 2 3 1 2 1 3 2 2 3 3 1 2 1 3 1 3 1 1 2 2 2 2 3 3 3 2 3 3 1 1 3 2  
[297] 2 2 1 3 3 1 3 2 2 2 3 1 2 2 1 1 3 1 1 3 3 2 3 3 3 3 3 3 3 1 3 3 3 3 3 2 3  
[334] 2 2 2 3 2 3 2 3 3 3 2 3 2 3 1 2 2 3 1 1 3 2 2 2 3 3 3 3 2 3 3 3 2 2 3 2 3  
[371] 2 3 3 3 2 3 2 3 2 3 3 1 3 2 3 3 3 3 2 1 3 2 3 3 3 2 1 2 3 3 2 2 3 2 2 2 2  
[408] 2 2 3 2 2 1 3 1 2 3 2 2 2 3 3 3 3 2 3 2 3 2 3 2 3 2 1 3 3 2 2 1 1 3 1 3 3  
[445] 3 2 2 3 3 3 1 3 3 2 2 3 2 3 2 2 3 3 3 3 2 2 3 3 3 1 3 2 3 2 2 2 2 2 2 1 1  
[482] 2 2 3 2 2 2 3 2 3 1 2 1 2 1 2 3 2 3 3 3 3 1 2 2 3 2 2 2 3 2 1 1 2 2 2 3 1  
[519] 2 2 2 2 3 2 2 3 3 3 3 3 2 3 3 3 2 1 3 1 2 3 2 3 2 2 2 2 3 2 2 1 3 1 2 1 2  
[556] 3 3 2 1 2 2 3 1 1 1 3 2 1 3 3 2 2 3 3 3 2 2 2 1 3 3 3 1 2 3 3 3 2 3 2 3 1  
[593] 2 3 1 2 2 3 3 3 3 3 2 3 2 3 3 1 2 2 1 3 3 1 1 3 2 1 2 2 2 2 2 3 2 3 2 2 3  
[630] 3 2 3 2 1 2 3 2 2 3 1 2 3 2 3 2 3 1 3 1 3 2 3 3 1 3 3 2 3 2 2 2 2 3 2 3 3  
[667] 2 2 1 3 2 2 2 1 3 3 3 3 2 2 2 2 2 3 2 2 2 2 2 3 3 2 2 2 1 3 3 3 2 3 3 2 2  
[704] 2 3 3 2 2 2 3 2 3 2 3 3 2 3 3 2 2 3 2 3 2 2 1 3 2 2 1 2 3 2 3 3 1 1 2 2 1  
[741] 1 2 2 2 2 1 3 3 3 2 3 1 2 3 1 1 3 3 2 2 2 3 3 2 2 3 2 3 2 2 1 3 1 2 3 2 1  
[778] 2 2 3 3 2 2 2 2 3 2 1 1 2 2 2 2 1 1 1 1 2 3 2 2 3 2 2 2 2 2 2 1 3 3 1 2 2  
[815] 2 2 3 3 2 2 3 3 1 3 2 2 3 2 3 2 2 3 3 1 1 2 2 3 1 2  
  
Within cluster sum of squares by cluster:  
[1] 1038.154 1963.475 1841.708  
 (between\_SS / total\_SS = 22.0 %)  
  
Available components:  
  
[1] "cluster" "centers" "totss" "withinss" "tot.withinss"  
[6] "betweenss" "size" "iter" "ifault"

# Add cluster assignments to the principal components data frame  
pca\_data <- as.data.frame(pc\_scores)  
clusters\_kmeans <- factor(kmeans\_result$cluster)  
pca\_data$Cluster <- clusters\_kmeans  
  
# Plot the first two principal components with cluster assignments  
ggplot(pca\_data, aes(x = Comp.1, y = Comp.2, color = Cluster)) +  
 geom\_point(size = 3) +  
 labs(title = "K-means Clustering on PCA Results",  
 x = "Principal Component 1",  
 y = "Principal Component 2") +  
 theme\_minimal()



## Hierarchical Clustering

# Compute the distance matrix  
distance\_matrix <- dist(pc\_scores)  
  
# Perform hierarchical clustering  
hc\_result <- hclust(distance\_matrix, method = "ward.D2")  
  
# Plot the dendrogram  
plot(hc\_result)



# Cut the dendrogram to form clusters  
num\_clusters <- 3 # Example: 3 clusters  
clusters\_hclust <- cutree(hc\_result, k = num\_clusters)  
  
# Print the clusters  
table(clusters\_hclust)

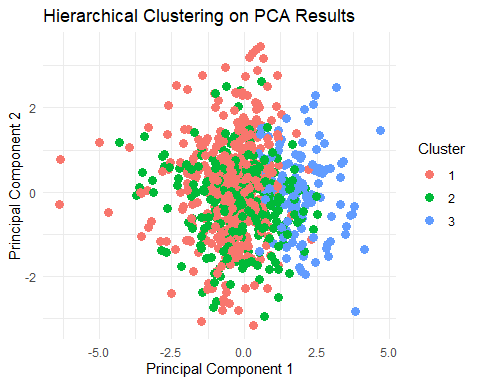
clusters\_hclust  
 1 2 3   
416 295 129

Notice they do not have to be the same as kmeans

table(clusters\_kmeans, clusters\_hclust)

clusters\_hclust  
clusters\_kmeans 1 2 3  
 1 51 99 0  
 2 258 89 1  
 3 107 107 128

# Add cluster assignments to the principal components data frame  
pca\_data <- as.data.frame(pc\_scores)  
pca\_data$Cluster <- factor(clusters\_hclust)  
  
# Plot the first two principal components with cluster assignments  
ggplot(pca\_data, aes(x = Comp.1, y = Comp.2, color = Cluster)) +  
 geom\_point(size = 3) +  
 labs(title = "Hierarchical Clustering on PCA Results",  
 x = "Principal Component 1",  
 y = "Principal Component 2") +  
 theme\_minimal()



# Comparison with outcomes

table(clusters\_kmeans, data\_to\_model$DIED)

clusters\_kmeans FALSE TRUE  
 1 124 26  
 2 277 71  
 3 293 49

table(clusters\_kmeans, data\_to\_model$DIED) |>   
 prop.table(margin = 1)

clusters\_kmeans FALSE TRUE  
 1 0.8266667 0.1733333  
 2 0.7959770 0.2040230  
 3 0.8567251 0.1432749

table(clusters\_kmeans, data\_to\_model$ANY\_COMPLICATION)

clusters\_kmeans no yes  
 1 68 82  
 2 139 209  
 3 145 197

table(clusters\_kmeans, data\_to\_model$ANY\_COMPLICATION) |>   
 prop.table(margin = 1)

clusters\_kmeans no yes  
 1 0.4533333 0.5466667  
 2 0.3994253 0.6005747  
 3 0.4239766 0.5760234

# If you have categorical data

Multiple Correspondence Analysis (MCA): Use MCA for dimension reduction of categorical data. Hierarchical Clustering on MCA Results: Perform hierarchical clustering on the results of MCA. k-modes Clustering: Use k-modes for clustering directly on categorical data.