Data Reduction
Week 12

PH 700A, Spring 2025

Rick Calvo

Table of contents

1 Data Reduction

1.1 Topics

- Packages
- Background
- Statistical Foundations
- Cluster Analysis

1.2 Packages

Old stuff: `tidyverse`, `explore`, `gtsummary`, `stats`

New stuff:

- `library(cluster)`
- `library(factoextra)`
- `library(dendextend)`

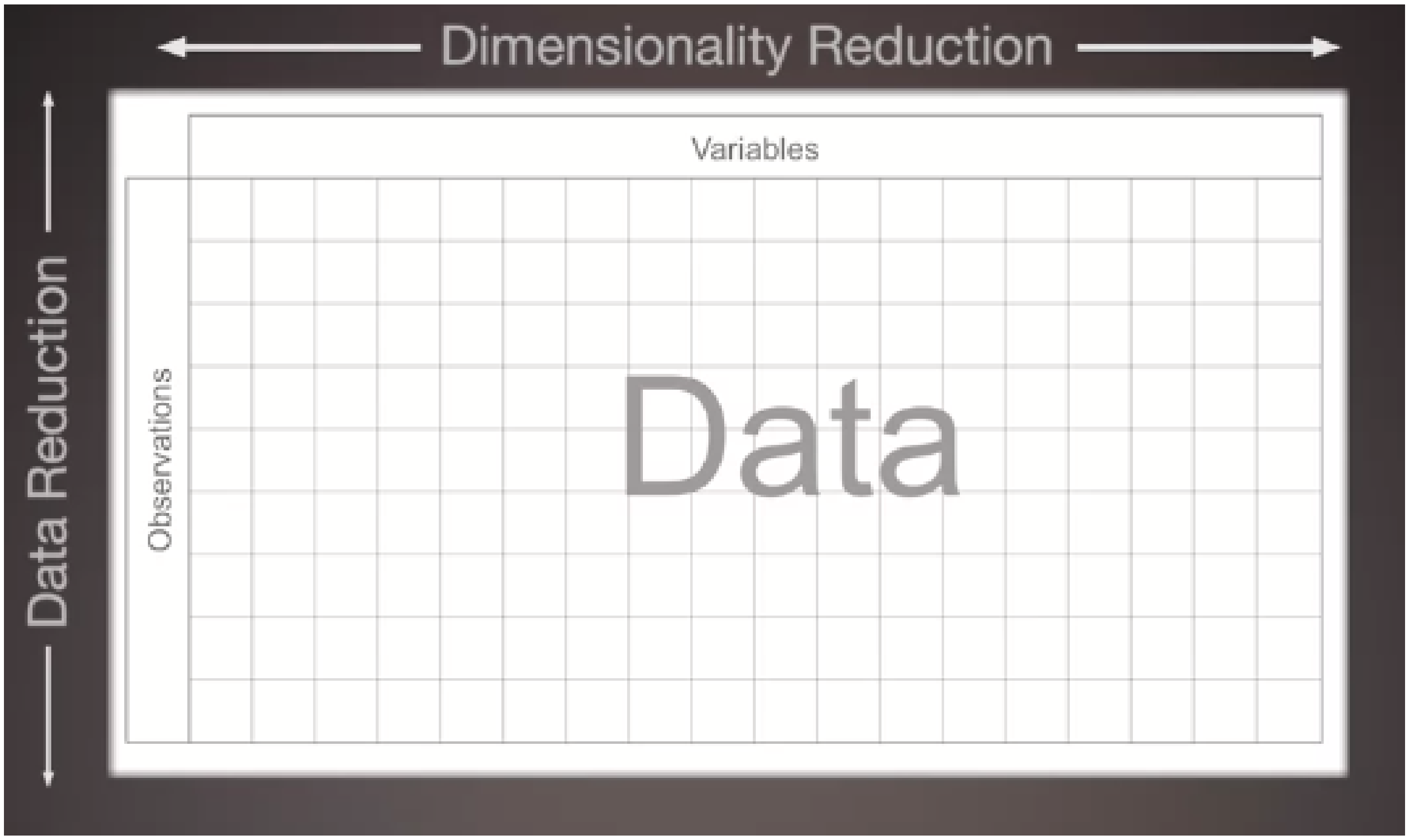
1.3 Background

This is a *two-part series* on *data reduction* and *dimensionality reduction.*

- The “behind-the-scenes” computation processes are similar – pattern recognition.
- The applications within R address different problems and will yield different results.
- Knowing when to use each method will expand your research toolkit.

Methods address the relatedness between variables to create groupings:

- Fewer observations -> **Data Reduction**
- Fewer variables -> **Dimensionality Reduction**



1.4 Reduction Methods

Data (Sample) Reduction	**Dimensionality Reduction**
Cluster Analysis	Principal Component Analysis
Neural Network	Factor Analysis
Latent Class Analysis	Correspondence Analysis
Discriminant Analysis	Total Correlation Explanation
Propensity Score Analysis	Structural Equation Modeling

1.5 Scenario

We are provided a secondhand dataset and little guidance.

- We do not know much about the sample
- Vague database description
- No specific *a priori* hypothesis

Our only option is to load and explore the data on our own.

1.6 Data Reduction

- Part of data complexity is the unknown heterogeneity of the sample.
- If we can *reduce* the complexity, we can better understand how the factors affect the patients.
- This is usually done anyways during the *Data Exploration* stage of the study
- Data-driven reduction methods apply *machine learning* to address the complexity

1.6.1 Sample

- Who are you dealing with?
- How can we classify the participants?
- Will we bias any grouping process?

1.6.2 Variables

- Which variables are important?
- Which are likely to be correlated?
- Which are clinically relevant?

1.7 Why Use a Data Reduction Algorithm?

- Manage fewer variables
- Group observations based on similarity
- Understand the data structure
- Classify variables into appropriate domains
- Minimize researcher bias

With enough time, we could manually explore every variable and figure something out to analyze...

1.8 But Save Time; More Success?

1.9 Hypothesis?

There is no hypothesis to test.

Data reduction is an exploratory process.

Purpose is to understand your data [so you can eventually formulate a hypothesis.]

Note

"Reducing" does not mean elimination of old items. The point of these methods is to consolidate any unexplained variablity into explainable "sets" – either by subsample groupings or variable groupings. These are essentially unbiased and computer-driven tasks for *data subsetting*, *variable categorization*, and *variable aggregation*. Descriptive statistics calculated using the original items are used to describe the "profiles" or groupings.

1.10 Rationale for Data Reduction Methods

- Sample should be *comparable* between **index** and **referent** groups (or **events** and **non-events**)
  - Without comparability, data may not be appropriate for all research questions
- Unknown heterogenity between groups is limiting

– Number of variables to sample size ratio may be imbalanced

– More variables exist than may be useful

– Excessive time required to understand data structure

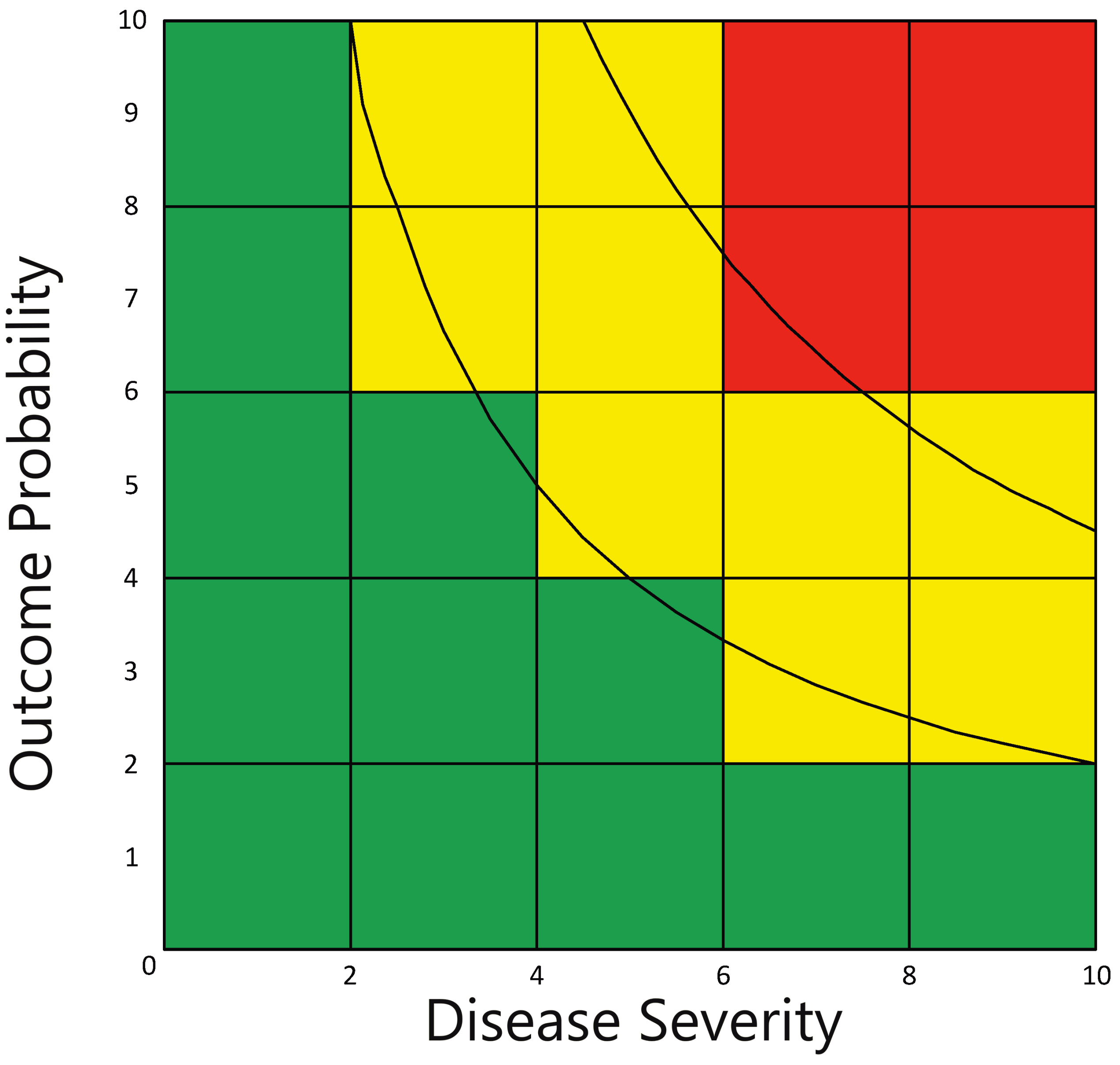


Figure 1: risk matrix

2 Technical Details

2.1 Mathematical Distance

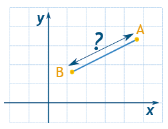
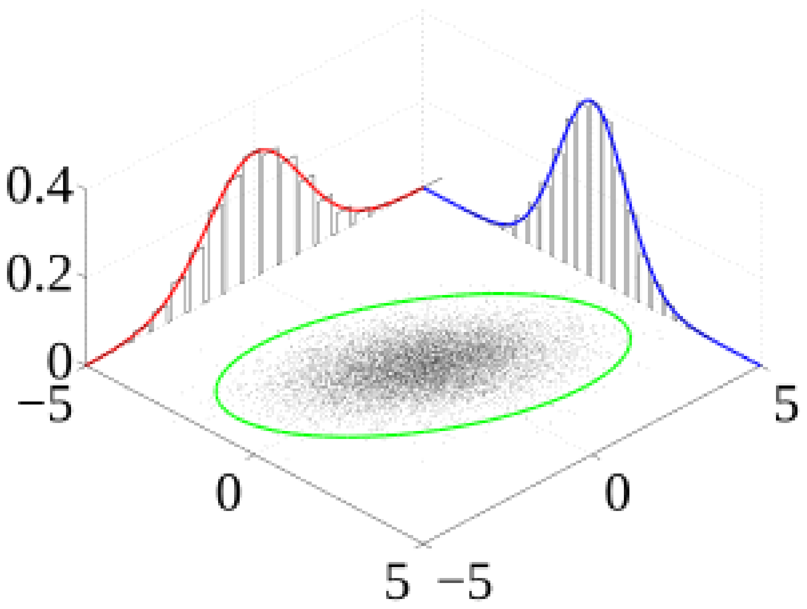


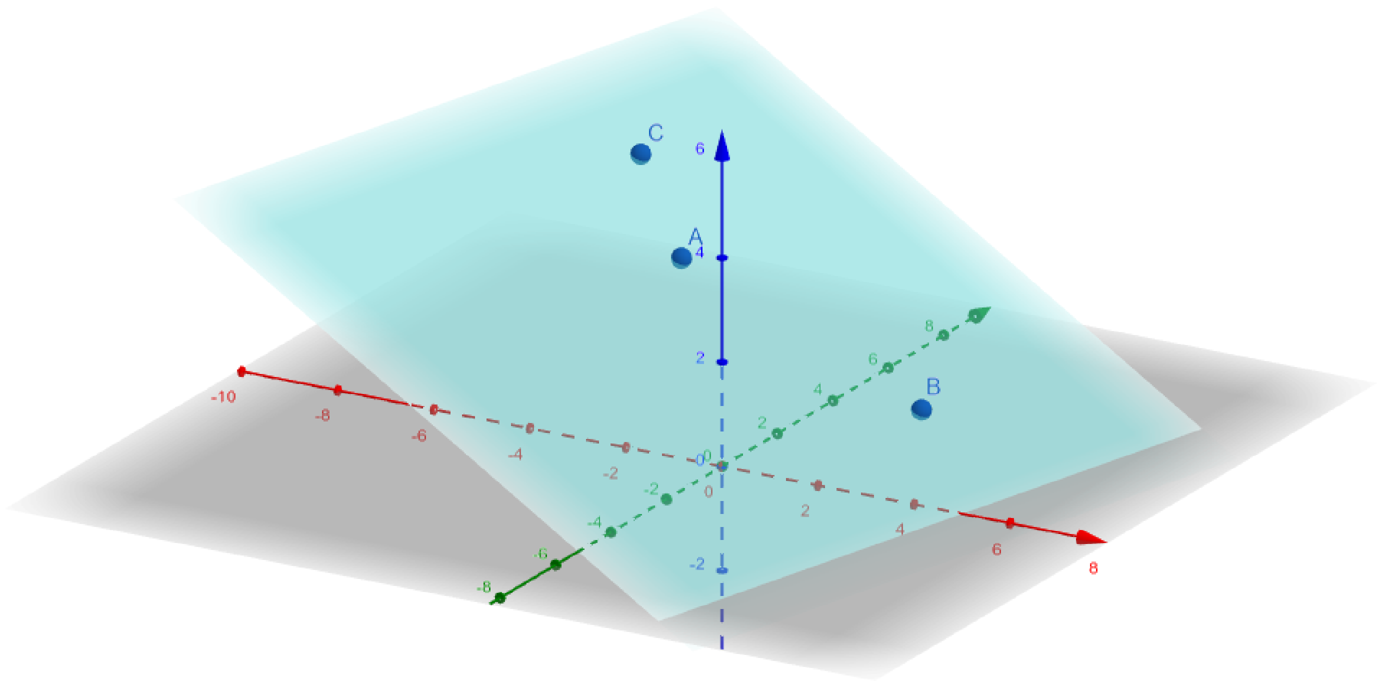
Figure 2: Source: mathisfun.com

2.2 Two-dimensional Data Representation



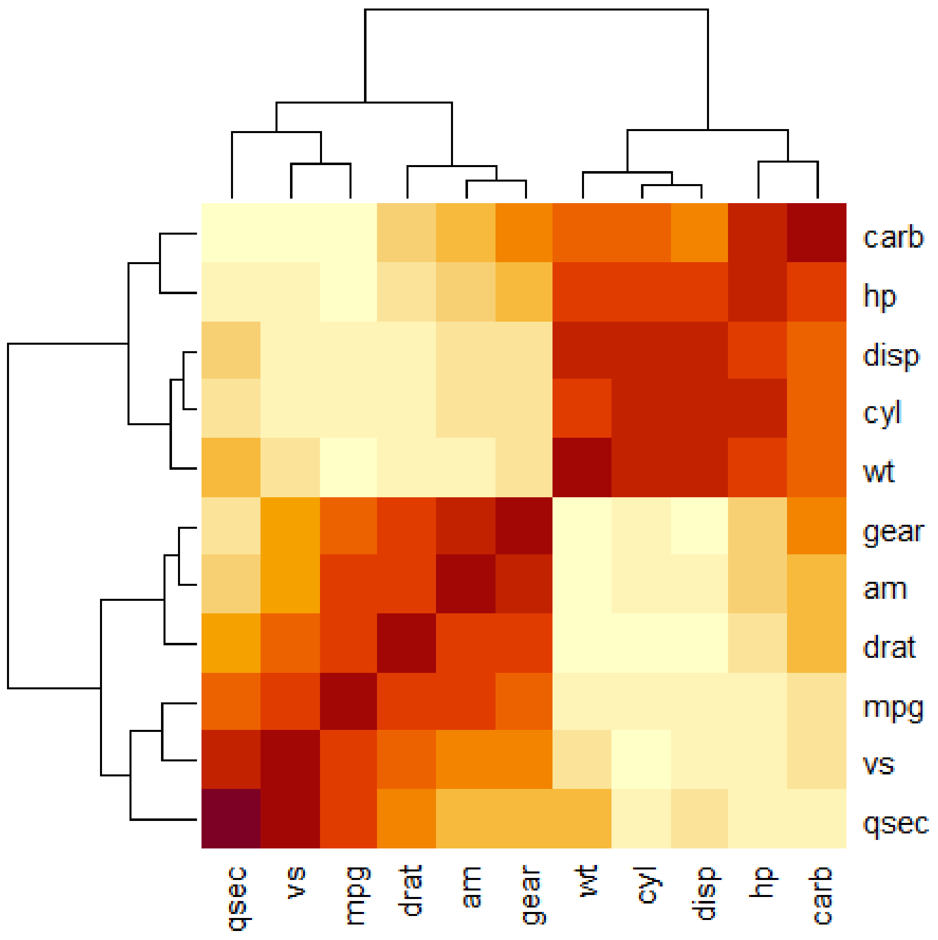
Each variable has a *mean*, and the scatter has a *joint mean*

2.3 Multidimensional Space



A *multidimensional* representation of variable proximity

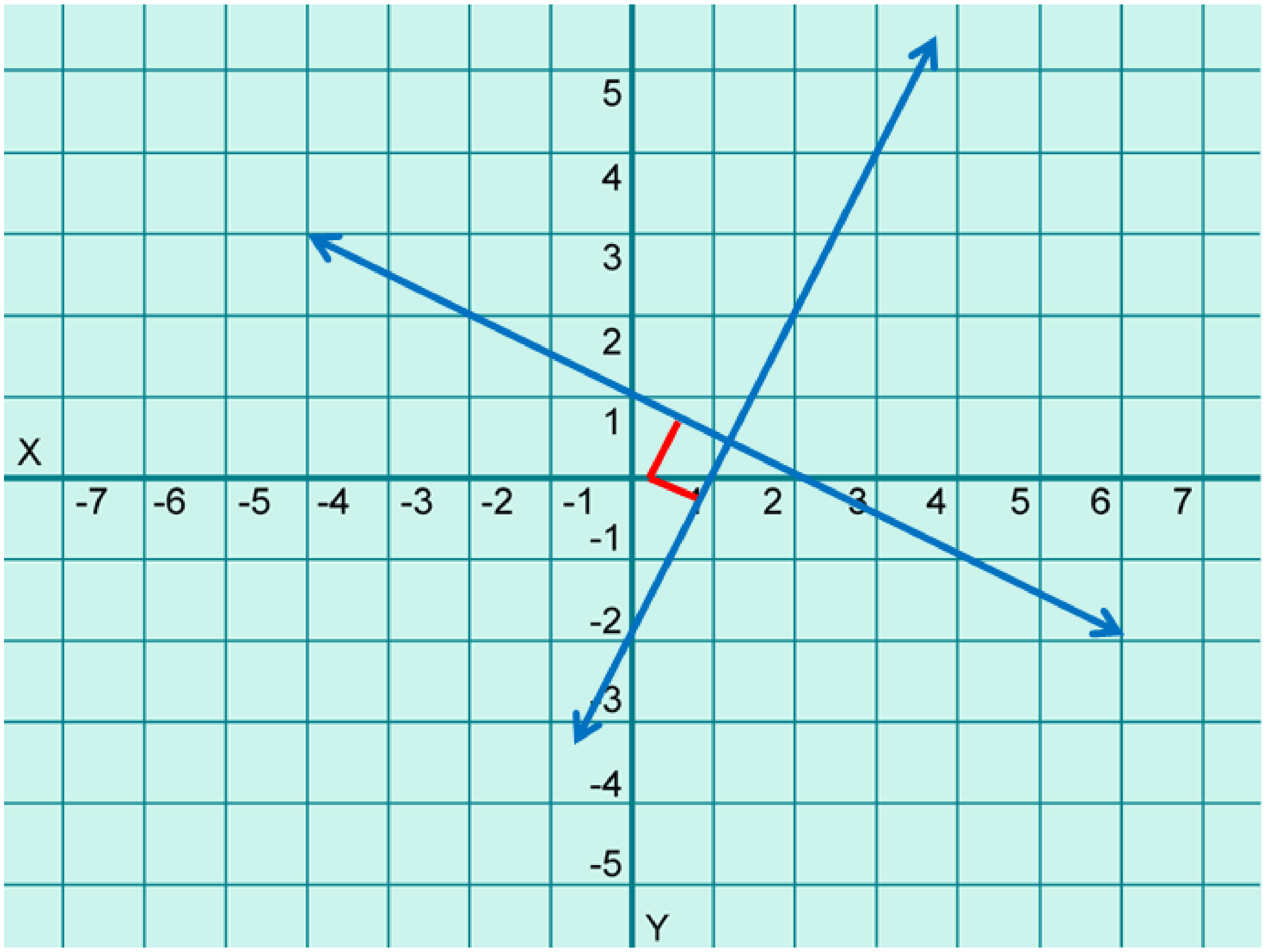
2.4 Correlation



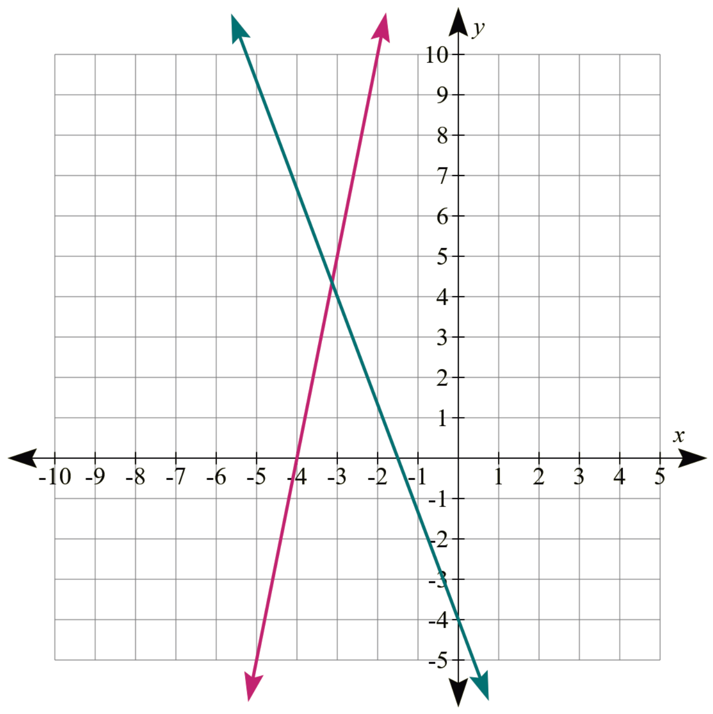
Coefficient Name	Description	Variable Types
Pearson	Linear correlation	normal continuous
Spearman	Monotonic rank-based correlation	continuous
Kendall	Monotonic rank-based correlation	continuous
Polychoric	Correlation among ordinal variables	polychotomous ordinal
Tetrachoric	Correlation among binary variables	dichotomized from continuous
Point-Biserial	Pearson for mixed variable types	one dichotomized and one continuous

2.5 Orthogonality

- The absence of correlation



- Perpendicular lines signify no correlation



- Non-perpendicular lines have *some amount* of correlation.

2.6 Variance-Covariance Matrix

Essentially a correlation matrix for the variability for multiple variables.

$$\begin{bmatrix} V_a & C_{a,b} & C_{a,c} & C_{a,d} & C_{a,e} \\ C_{a,b} & V_b & C_{b,c} & C_{b,d} & C_{b,e} \\ C_{a,c} & C_{b,c} & V_c & C_{c,d} & C_{c,e} \\ C_{a,d} & C_{b,d} & C_{c,d} & V_d & C_{d,e} \\ C_{a,e} & C_{b,e} & C_{c,e} & C_{d,e} & V_e \end{bmatrix}$$

This *variance-covariance* matrix is between 5 variables, where:

- The *diagonal values* are the *variance* of each of the five variables
  - V_a symbolizes the *variance* of variable `a`, etc.
- The *non-diagonal values* are the *covariance* for each combination of two variables and is symmetric across the diagonal

  - $C_{a,b}$ symbolizes the *covariance* between variables `a` and `b`, etc.

Variance is the variability around a variable's mean.

Covariance is the joint variability around the "joint mean" (geometric mean) of two variables.

2.7 Distance is Relatedness!

- Statistical measures of *similarity* and *dissimiliarity* are derived from *correlation*, *variance*, and *covariance* values
- The *distance* between values in a mathematical space is what drives *similarity/dissimilarity*
- There are many ways to measure these and are based on:
  - Emphasis on presence vs. absence of features
  - Handling of zeroes
  - Normalization of the variables/vectors
  - Unit of measurement
  - Incorporation of variance-covariance structure

2.8 Distance Measures in R

Measure	Use Case
euclidian	continuous only
manhattan	ordinal/interval
gower	mixed; continuous and binary
jaccard	binary only

Detailed descriptions for measures of distance can be viewed at https://www.nhm.uio.no/english/research/resources/past/help/similarity.html.

3 Cluster Analysis

3.1 Background

An exploratory technique used to form groups of observations based on common "profiles"

"Profiles" are defined by the factors that each group exemplifies after the groups are created

Clustering schema can be:

- Hierarchical
  - Agglomerative

    - Divisive

- Partitioning
    - Parametric
    - Nonparametric

In addition, they can be:

```
-   Supervised

-   Unsupervised
```

There are **MANY** ways to perform a cluster analysis, each employing a different measure for *distance* and *schema*.

> **Note**
>
> Because this is a data-driven pattern recognition method, you must set a seed at the outset otherwise subsequent attempts of the same analysis on the same data can yield different results. Clusters could be heavily dependent on the "starting point" of the algorithm which is randomly selected each time the code is run without a set seed.

3.2 Commands

`library(stats)` contains basic aspects of cluster analysis

`library(cluster)` contains advanced aspects with customization

3.2.1 Calculation of Distances

`library(stats)`: `dist()` - Only accepts numeric variables

`library(cluster)`: `daisy()` - Accepts various variable types

3.2.2 Hierarchical Clustering Algorithms

`library(stats)`: `hclust()` - Hierarchical Clustering

`library(cluster)`: `agnes()` - Agglomerative Nesting

`library(cluster)`: `diana()` - Divisive Analysis

`library(cluster)`: `mona()` - Monothetic Analysis; Binary Variables

3.2.3 Partitioning Clustering Algorithms

`library(stats)`: `kmeans()` - k-Means Clustering

`library(cluster)`: `clara()` - Clustering Large Applications

`library(cluster)`: `fanny()` - Fuzzy Analysis Clustering

`library(cluster)`: `pam()` - Partition Around Medoids

3.3 Steps

1. Evaluate and select variables
2. Pick your distance measure
3. Run your cluster analysis
4. Identify the optimal number of clusters to retain
5. Label your clusters

3.4 1. Variable Selection

- Are variables categorical, continuous, or a mixture?
  - Variable forms will determine which clustering method you can perform
- Use only the variables that provide context to develop group "profiles"
  - Omit infrequent variables
  - Categorize variables based on known thresholds
  - Address missing values and outliers
- Process continuous variables via *scaling* and *standardization*
  - Variables with different units need to be transformed
  - Standardizing will remove residual correlation

3.5 2. Pick your distance measure

- Dependent on the variable types you select
- Each distance measure offers a different benefit
- Largely determined by variables

```r
set.seed(20250416)

distances <- daisy(x = df,
                  metric = "gower",
                  type = list(factor = c("catvar1"),
                              numeric = c("var1", "var2", "var3", "var4", "var5")
                             ),
                  stand = FALSE)
```

`daisy()` will calculate distances with the variables you provide and store it in `distances`

- `x = df` specifies the data frame
- `metric = "gower"` calculates the Gower distances for mixed data
  - Other options include "euclidian" or "manhattan"
- `type = list()` allows you to specify the variables and their respective forms
  - `factor = c()` is for categorical variables
  - `numeric = c()` is for continuous variables
  - Other types include `asymm`, `symm`, `ordered`, `logratio`, `ordratio`
- `stand = FALSE` specifies not to standardize the variables in `df`
  - `TRUE` will z-score all variables
  - standardization will not work if there are categorical variables in `df`

3.6 3. Run the cluster analysis

Agglomerative hierarchical clustering with the `hclust()` command works fine if there are no special circumstances.

```r
cluster.out <- hclust(distances, method = "ward.D")

plot(cluster.out, labels = FALSE)
```

- `plot()` creates the dendrogram plot of the final clustering schema
- `distances` is the distance object created earlier
- `method` = specifies the clustering methodology to create linkages between observations
  - `ward.D`: starts with each data point its own cluster and then combines clusters in a way that minimizes the increase in total within-cluster variance
  - `ward.D2`: similar to `ward.D` but squares the distances before updating branches

   - `single`: single-linkage; the distance between two clusters is defined as the shortest distance between any two points in the two different clusters
   - `complete`: complete-linkage; the distance between two clusters is defined as the longest distance between any two points in the two different clusters
   - `average`: average-linkage; the distance between two clusters is defined as the average distance between each point in one cluster to every point in the other cluster
   - `mcquitty`: starts by assuming each observation is a separate cluster and merges clusters that have the highest average similarity value
   - `median`: a variation of k-means clustering where the median is used instead of the mean for to determine its "centroid"
   - `centroid`: vectorizes the clusters and merges groups based on distance of the vector midpoints

3.7 4. Identify the optimal number of clusters to retain

Cluster selection is performed visually using `library(factoextra)` and the `fviz_nbclust()` command.

There are two main ways:

- Silhouette Plots
- "Elbow Method" (Within-cluster Sum of Squares [WSS])

3.8 Silhouette Plots

The silhouette plot identifies how well each observation lies in a cluster on average. High values indicate good clustering.

```r
library(factoextra)

silh <- fviz_nbclust(x = df,
                     FUNcluster = hcut,
                     diss = distances,
                     method = "silhouette",
                     k.max = 10)
```

`x =` specifies the data frame

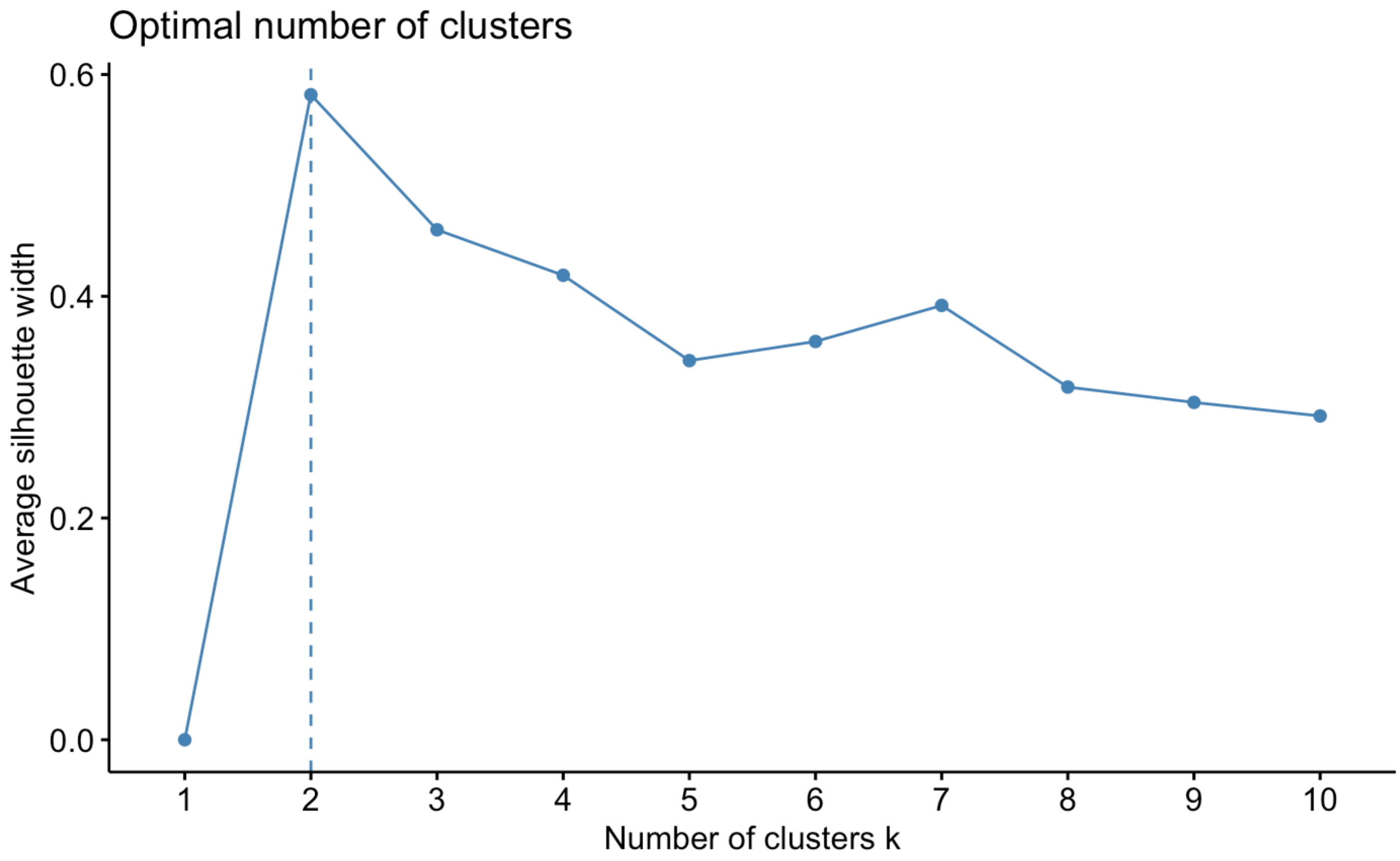
`FUNcluster = hcut` specifies the partitioning function to be hierarchical clustering

```
- Other options include: `pam`, `clara`, `fanny`, etc.
```

`diss` = specifies the `distance` matrix object

`method = "silhouette"` specifies silhouette plot generation

`k.max` specifies the maximum number of clusters to consider



3.9 "Elbow" Method

The appearance of a bend in the plot (aka the elbow) signifies when the rate of decrease in the WSS changes. Additional clusters beyond this threshold do not capture as much of the variability in the data.

```
wss <- fviz_nbclust(x = df,
                    FUNcluster = hcut,
                    diss = distances,
                    method = "wss",
                    k.max = 10)
```

`x` = specifies the data frame

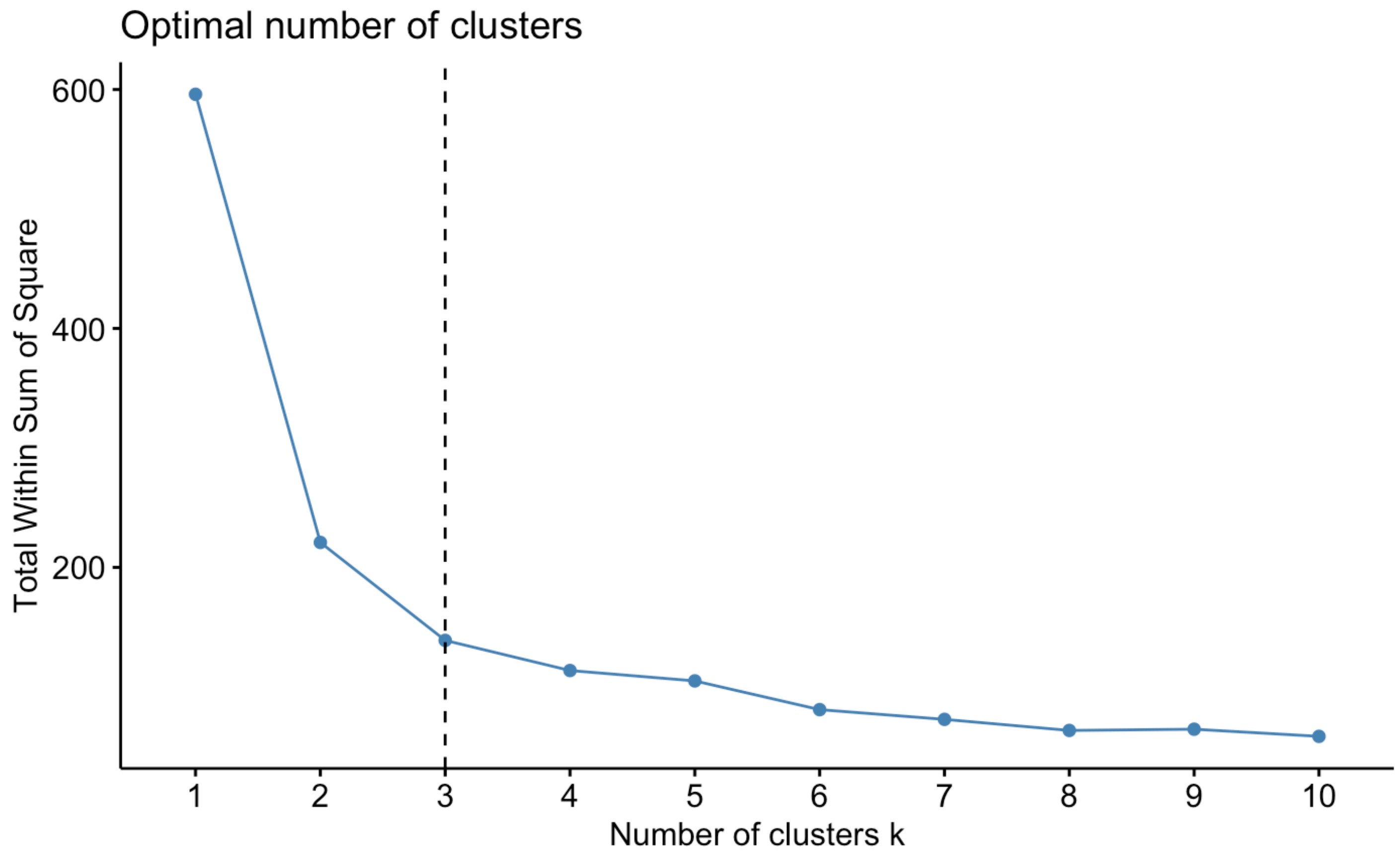
`FUNcluster = hcut` specifies the partitioning function to be hierarchical clustering

```
- Other options include: `pam`, `clara`, `fanny`, etc.
```

`diss =` specifies the `distance` matrix object

`method = "wss"` specifies within-cluster sum of squares plot generation

`k.max` specifies the maximum number of clusters to consider



The bend marks the appropriate number of clusters.

3.10 5. Characterize/profile your clusters

- Evaluate commonalities and differences
- Provide a label to each group that can be used

The `cutree()` is from `library(stats)` and will cut the dendrograph at a specified number of groups.

```r
finalgroups <- cutree(cluster.out, 3)

aggregate(df,list(finalgroups),mean)
```

- `aggregate()` is like `dplyr::summarize()` for a single statistic.
- Computes the *mean* of all variables in `df` by the groupings in `finalgroups` to use for labeling

4 Cluster Analysis Example

4.1 Prepare Dataset

```
library(cluster)

clusterSet <- df %>%
  select(arrival_transport, lnlos, disposition, temperature, heartrate, resprate, o2sat, sbp, highpa

clusterSet <- clusterSet %>%
  mutate(ambulance = if_else(arrival_transport == "AMBULANCE", 1, 0),
         otherunk = case_when(arrival_transport == "OTHER" ~ 1, arrival_transport == "UNKNOWN" ~ 1,
         walkin = if_else(arrival_transport == "WALK IN", 1, 0),
         admitted = if_else(disposition == "ADMITTED", 1, 0),
         homedisp = if_else(disposition == "HOME", 1, 0),
         otherdisp = case_when(disposition == "ELOPED" ~ 1, disposition == "LEFT AGAINST MEDICAL ADV
  )

clusterSet <- clusterSet %>%
  select(-arrival_transport, -disposition)

catvars <- c("ambulance", "otherunk", "walkin", "admitted", "homedisp", "otherdisp", "highpain")

contvars <- c("lnlos", "temperature", "heartrate", "resprate", "o2sat", "sbp")

clusterSet <- clusterSet %>%
  mutate(across(all_of(contvars), scale),
         across(all_of(catvars), as.factor))

clusterSet <- na.omit(clusterSet)
```

Continuous variables are transformed and z-scored. Categorical variables are dummied. Total vars = 13.

Observations with any missing values were removed. Current n = 194.

4.2 Generate Distances

```
set.seed(20250416)

distances <- daisy(x = clusterSet,
                  metric = "gower",
                  type = list(factor = catvars,
                              numeric = contvars),
                  stand = FALSE)
```

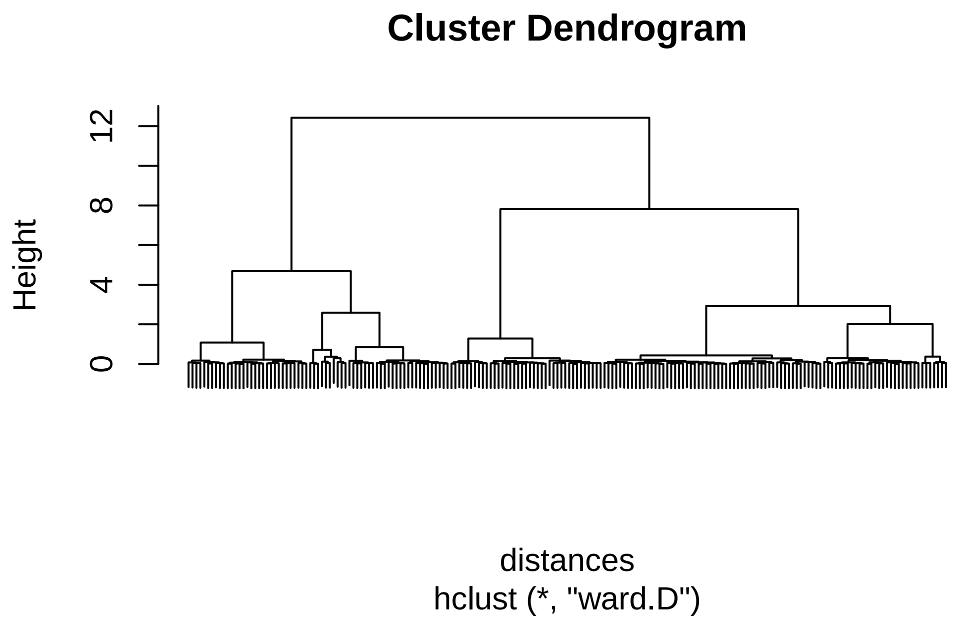
`catvars` contains a list of all factor vars.

`contvars` contains a list of all continuous vars.

4.3 Perform Cluster Analysis

```
cluster.out <- hclust(distances, method = "ward.D")

plot(cluster.out, labels = FALSE)
```

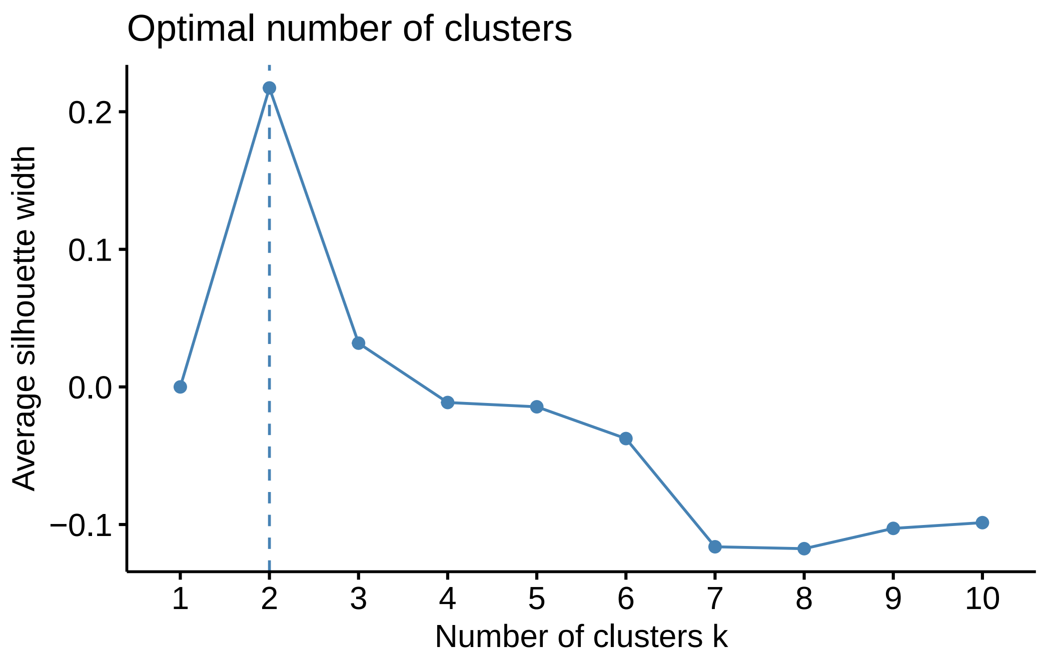


`Height` is the distance between branches. Branching that occurs at higher `heights` signify greater dissimilarity between observations after the split.

4.4 Silhouette Plot

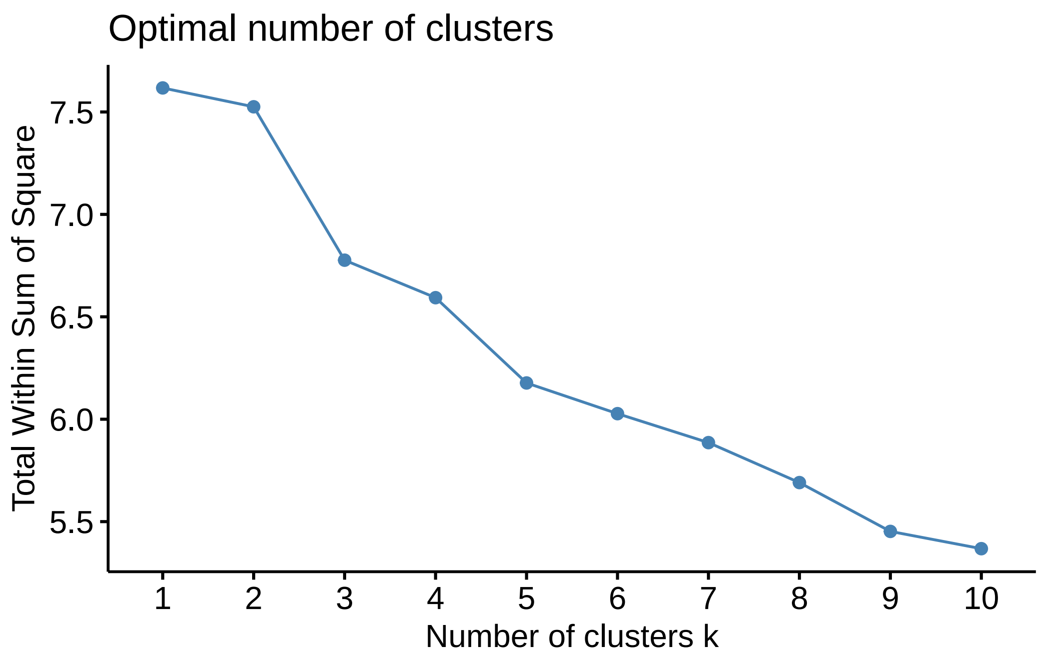
```
library(factoextra)

silh <- fviz_nbclust(x = clusterSet,
                     FUNcluster = hcut,
                     diss = distances,
                     method = "silhouette",
                     k.max = 10)
silh
```



4.5 WSS Plot

```r
wss <- fviz_nbclust(x = clusterSet,
                    FUNcluster = hcut,
                    diss = distances,
                    method = "wss",
                    k.max = 10)
wss
```



4.6 Labeling Groups

```
library(gtsummary)

finalgroups <- cutree(cluster.out, 3)

aggregate(clusterSet, list(finalgroups), mean)
```

```
  Group.1       lnlos temperature   heartrate   resprate      o2sat         sbp
1       1 -0.04659043  0.10581954  0.10999661  0.1324576 -0.1131657 -0.12232353
2       2  0.36154020 -0.17921364 -0.16944836 -0.1890102  0.3075728  0.20130938
3       3  0.11935736  0.06608512 -0.02130227 -0.2241269 -0.2562595 -0.06176659
  highpain ambulance otherunk walkin admitted homedisp otherdisp
1       NA        NA       NA     NA       NA       NA        NA
2       NA        NA       NA     NA       NA       NA        NA
3       NA        NA       NA     NA       NA       NA        NA
```

```
clusterSet <- clusterSet %>%
  mutate(rownum = row_number())

finalgroups <- as.data.frame(finalgroups)

finalgroups <- finalgroups %>%
  mutate(rownum = row_number())
```

Characteristic	**1** N = 88[1]	**2** N = 67[1]	**3** N = 39[1]
lnlos	-0.11 (-0.47, 0.39)	0.20 (-0.26, 1.06)	0.12 (-0.19, 0.34)
temperature	0.09 (-0.02, 0.23)	0.05 (-0.09, 0.09)	0.05 (-0.04, 0.11)
heartrate	0.12 (-0.78, 0.78)	-0.27 (-0.80, 0.47)	-0.17 (-0.75, 0.73)
resprate	-0.05 (-0.71, 0.62)	-0.05 (-0.71, -0.05)	-0.71 (-0.71, -0.05)
o2sat	0.11 (-0.63, 0.48)	0.48 (-0.26, 0.85)	0.11 (-0.63, 0.85)
sbp	-0.18 (-0.91, 0.49)	0.00 (-0.47, 0.79)	-0.07 (-0.97, 0.83)
highpain			
0	59 (67%)	46 (69%)	29 (74%)
1	29 (33%)	21 (31%)	10 (26%)
ambulance			
0	7 (8.0%)	35 (52%)	39 (100%)
1	81 (92%)	32 (48%)	0 (0%)
otherunk			
0	81 (92%)	66 (99%)	39 (100%)
1	7 (8.0%)	1 (1.5%)	0 (0%)
walkin			
0	88 (100%)	33 (49%)	0 (0%)
1	0 (0%)	34 (51%)	39 (100%)
admitted			
0	0 (0%)	67 (100%)	0 (0%)
1	88 (100%)	0 (0%)	39 (100%)
homedisp			
0	88 (100%)	10 (15%)	39 (100%)
1	0 (0%)	57 (85%)	0 (0%)
otherdisp			
0	88 (100%)	57 (85%)	39 (100%)
1	0 (0%)	10 (15%)	0 (0%)

[1]Median (Q1, Q3); n (%)

```r
clusterID <- full_join(clusterSet, finalgroups, by = "rownum")

clusterID %>% select(-rownum) %>% tbl_summary(by = finalgroups)
```