Week-1

Rajesh Kalakoti

2023-08-03

```r
# Include the script from the R directory
project_path <- here()
source(here("R", "utils.R"))
source(here("R","distance_functions.R"))
```

1 Distance Functions

- **Euclidean distance:**
  $S(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$
- **Manhattan distance :**
  $S(x, y) = \sum_{i=1}^{n} |x_i - y_i|$
- **Chebyshev distance:**
  $S(x, y) = max_i(|x_i - y_i|)$
- **Mahalanobis Distance:**
  $S(x, y) = \sqrt{\sum_{i=1}^{n} (x - y)^T C^{-1} (x - y)}$
- **Canberra Distance:**
  $S(x, y) = \sum\limits_{i=1}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|}$
- **Cosine Distance:**
  $S(x, y) = \frac{x \cdot y}{\|x\| \|y\|}$

```r
set.seed(123)
n <-  100
a <- runif(n)  # Generate random vector a
b <- runif(n)
euclidean_dist <- euclidean_distance(a,b)
manhattan_dist <- manhattan_distance(a,b)
chebyshev_dist <-chebyshev_distance(a,b)
mahalanobis_dist <- mahalanobis_distance(a,b)
canberra_dist  <- canberra_distance(a,b)
cosine_dist <- cosine_distance(a,b)

print(paste("Euclidean distance:", euclidean_dist))
```

```
## [1] "Euclidean distance: 4.0314595095184"
```

```r
print(paste("Manhattan distance:", manhattan_dist))
```

```
## [1] "Manhattan distance: 33.2492298050784"
```

```r
print(paste("Chebyshev distance:", chebyshev_dist))
```

```
## [1] "Chebyshev distance: 0.912031705025584"
```

```r
print(paste("Mahalanobis distance:", mahalanobis_dist))
```

```
## [1] "Mahalanobis distance: 1.4142135623731"
```

```r
print(paste("Canberra distance:", canberra_dist))
```

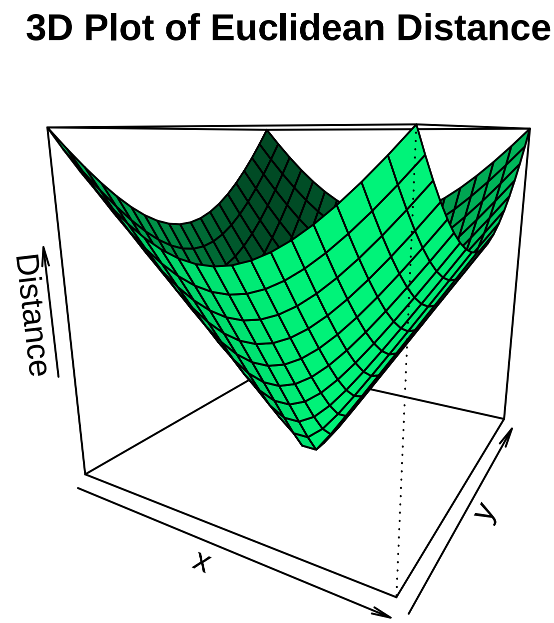
```
## [1] "Canberra distance: 36.8762171479272"
```

```r
print(paste("Cosine distance:", cosine_dist))
```

```
## [1] "Cosine distance: 0.245388685498833"
```

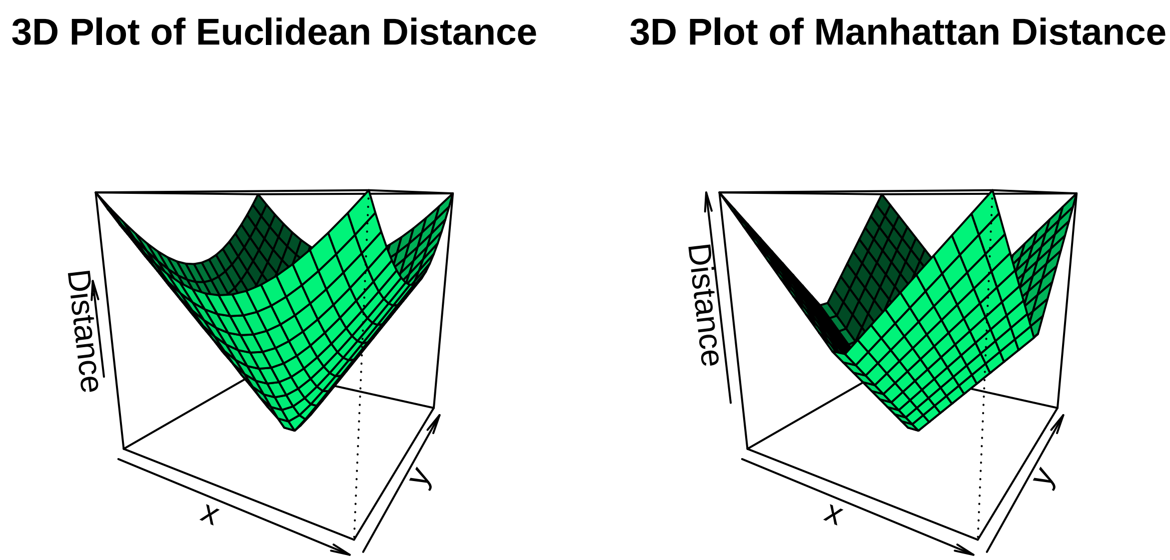
1.1 Impact of the rotation of underlying data set

```r
x <- y <- seq(-1, 1, length = 20)
grid <- expand.grid(x = x, y = y)  # Create a grid of points
z <- matrix(0, nrow = length(x), ncol = length(y))  # Initialize the z matrix

for (i in 1:length(x)) {
  for (j in 1:length(y)) {
    z[i, j] <- euclidean_distance(c(x[i], y[j]), c(0, 0))
  }
}
persp(x, y, z,
      main = "3D Plot of Euclidean Distance",
      zlab = "Distance",
      theta = 30, phi = 15,
      col = "springgreen", shade = 0.5)
```




```
x <- y <- seq(-1, 1, length = 20)
grid <- expand.grid(x = x, y = y)  # Create a grid of points
z_euclidean <- matrix(0, nrow = length(x), ncol = length(y))  # Initialize the z matrix for Euclidean d
z_manhattan <- matrix(0, nrow = length(x), ncol = length(y))   # Initialize the z matrix for Manhattan

for (i in 1:length(x)) {
  for (j in 1:length(y)) {
    z_euclidean[i, j] <- euclidean_distance(c(x[i], y[j]), c(0, 0))
    z_manhattan[i, j] <- manhattan_distance(c(x[i], y[j]), c(0, 0))
  }
}

# Create a layout of subplots to show both Euclidean and Manhattan distances
par(mfrow = c(1, 2))

# Plot for Euclidean distance
persp(x, y, z_euclidean,
      main = "3D Plot of Euclidean Distance",
      zlab = "Distance",
      theta = 30, phi = 15,
      col = "springgreen", shade = 0.5)

# Plot for Manhattan distance
persp(x, y, z_manhattan,
      main = "3D Plot of Manhattan Distance",
      zlab = "Distance",
```

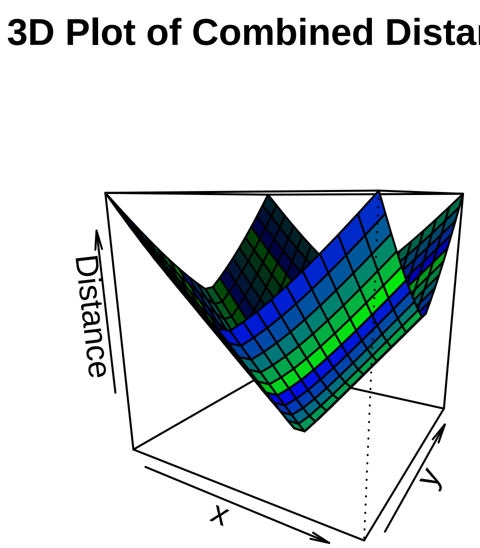
```r
      theta = 30, phi = 15,
      col = "springgreen", shade = 0.5)
```




```r
# Reset the layout
par(mfrow = c(1, 1))
```

```r
x <- y <- seq(-5, 5, length = 20)
grid <- expand.grid(x = x, y = y)  # Create a grid of points

z_euclidean <- matrix(0, nrow = length(x), ncol = length(y))  # Initialize the z matrix for Euclidean d
z_manhattan <- matrix(0, nrow = length(x), ncol = length(y))   # Initialize the z matrix for Manhattan

for (i in 1:length(x)) {
  for (j in 1:length(y)) {
    z_euclidean[i, j] <- euclidean_distance(c(x[i], y[j]), c(0, 0))
    z_manhattan[i, j] <- manhattan_distance(c(x[i], y[j]), c(0, 0))
  }
}

# Combine the distances and choose different colors for each
combined_distances <- z_euclidean + z_manhattan
color_palette <- colorRampPalette(c("blue", "green"))(100)  # Choose colors for mapping distances

# Create a layout of subplots
layout(matrix(c(1, 2), nrow = 1))
```

```r
# Plot both distances on the same 3D plane with different colors
persp(x, y, combined_distances,
      main = "3D Plot of Combined Distances",
      zlab = "Distance",
      theta = 30, phi = 15,
      col = color_palette, shade = 0.5)

# Reset the layout
layout(1)
```




```r
library("car")
```

```
## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following object is masked from 'package:purrr':
##
##     some
```

```r
library("rgl")

data(iris)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```r
sep.l <- iris$Sepal.Length
sep.w <- iris$Sepal.Width
pet.l <- iris$Petal.Length
```

```r
library("car")
library("rgl")
data(iris)
sep.l <- iris$Sepal.Length
sep.w <- iris$Sepal.Width
pet.l <- iris$Petal.Length

save <- getOption("rgl.useNULL")
options(rgl.useNULL = TRUE)
scatter3d(x = sep.l, y = pet.l, z = sep.w, groups = iris$Species,
          surface = FALSE, ellipsoid = TRUE)
```

```
## Loading required namespace: mgcv
```

```r
# widget <- rglwidget()
#
# # Explicitly set the elementId property
# widget$elementId <- "my-rgl-plot"
# widget
#
```

```r
library(rgl)

# Load the Iris dataset
data(iris)

# Create an interactive 3D scatter plot
scatter3d(x = iris$Sepal.Length, y = iris$Petal.Length, z = iris$Sepal.Width,
          groups = iris$Species, surface = FALSE, ellipsoid = TRUE)

# Display the interactive plot#
# rglwidget()
```

```r
library(rgl)
rgl::setupKnitr(autoprint = FALSE)
# Adding Titles and Labeling Axes to Plot
cone <- function(x, y){
sqrt(x ^ 2 + y ^ 2)
}

# prepare variables.
x <- y <- seq(-1, 1, length = 30)
z <- outer(x, y, cone)

# plot the 3D surface
# Adding Titles and Labeling Axes to Plot
persp3d(x, y, z,col = "orange")
# add animaton
play3d(spin3d(axis = c(0, 0, 1)), duration = 10)

# Un comment the code below if you want to see the interactve plot.
# rglwidget()
# rgl::setupKnitr(autoprint =FALSE)
```

```r
library(rgl)

x <- y <- seq(-5, 5, length = 20)
grid <- expand.grid(x = x, y = y)  # Create a grid of points

z_euclidean <- matrix(0, nrow = length(x), ncol = length(y))  # Initialize the z matrix for Euclidean d
z_manhattan <- matrix(0, nrow = length(x), ncol = length(y))   # Initialize the z matrix for Manhattan

for (i in 1:length(x)) {
  for (j in 1:length(y)) {
    z_euclidean[i, j] <- euclidean_distance(c(x[i], y[j]), c(0, 0))
    z_manhattan[i, j] <- manhattan_distance(c(x[i], y[j]), c(0, 0))
  }
}

# Combine the distances and choose different colors for each
combined_distances <- z_euclidean + z_manhattan
color_palette <- colorRampPalette(c("blue", "green"))(100)  # Choose colors for mapping distances

# Create a layout of subplots
layout(matrix(c(1, 2), nrow = 1))

# Plot both distances on the same 3D plane with different colors
persp3d(x, y, combined_distances,
      main = "3D Plot of Combined Distances",
      zlab = "Distance",
      theta = 30, phi = 15,
      col = color_palette, shade = 0.5)

# Reset the layout
# layout(1)
# rglwidget()
```