R PROJECT 1

SAMEER

2024-04-17

#READING THE DATASET

```r
# Read the dataset
data <- read.csv("C:/Users/samee/Downloads/Global Food Security Index 2022.csv")
# Checking column names of data dataset
print(colnames(data))
```

```
## [1] "Number"                          "Country"
## [3] "Overall.Score"                   "Affordability"
## [5] "Availability"                    "Quality.Safety"
## [7] "Sustainability.and.Adaptation"
```

#DATA CLEANING

```r
# Read the dataset
data <- read.csv("C:/Users/samee/Downloads/Global Food Security Index 2022.csv")

# Checking for missing values
print(colSums(is.na(data)))
```

```
##                        Number                       Country
##                             0                             0
##                 Overall.Score                 Affordability
##                             0                             0
##                  Availability                Quality.Safety
##                             0                             0
## Sustainability.and.Adaptation
##                             0
```

```r
# Rename columns for easier manipulation
colnames(data) <- c("Rank", "Country", "Overall_Score", "Affordability", "Availability", "Quality_Safet

# Remove rows with missing values, if any
data <- na.omit(data)

# Ensure data consistency and proper data types
data$Overall_Score <- as.numeric(data$Overall_Score)
data$Affordability <- as.numeric(data$Affordability)
data$Availability <- as.numeric(data$Availability)
data$Quality_Safety <- as.numeric(data$Quality_Safety)
```

```r
data$Sustainability.and.Adaptation <- as.numeric(data$Sustainability.and.Adaptation)


# Check summary statistics of numeric columns
summary(data[, c("Overall_Score", "Affordability", "Availability", "Quality_Safety", "Sustainability.and
```

```
##  Overall_Score   Affordability    Availability   Quality_Safety
##  Min.   :36.30   Min.   :25.00   Min.   :26.60   Min.   :34.90
##  1st Qu.:51.90   1st Qu.:52.70   1st Qu.:50.20   1st Qu.:54.00
##  Median :63.00   Median :73.40   Median :59.30   Median :69.00
##  Mean   :62.16   Mean   :69.02   Mean   :57.78   Mean   :65.88
##  3rd Qu.:73.00   3rd Qu.:87.10   3rd Qu.:65.60   3rd Qu.:77.60
##  Max.   :83.70   Max.   :93.30   Max.   :81.20   Max.   :89.50
##  Sustainability.and.Adaptation
##  Min.   :32.80
##  1st Qu.:45.50
##  Median :53.70
##  Mean   :54.13
##  3rd Qu.:60.10
##  Max.   :87.40
```

```r
# Check the structure of the cleaned dataset
str(data)
```

```
## 'data.frame':    113 obs. of  7 variables:
##  $ Rank                         : chr  "1st" "2nd" "3rd" "4th" ...
##  $ Country                      : chr  "Finland" "Ireland" "Norway" "France" ...
##  $ Overall_Score                : num  83.7 81.7 80.5 80.2 80.1 79.5 79.1 79.1 78.8 78.7 ...
##  $ Affordability                : num  91.9 92.6 87.2 91.3 92.7 89.8 91.9 88.3 91.5 90 ...
##  $ Availability                 : num  70.5 70.5 60.4 69 70.7 81.2 68.3 75.7 71.6 77 ...
##  $ Quality_Safety               : num  88.4 86.1 86.8 87.7 84.7 77.4 85 89.5 77.6 79.8 ...
##  $ Sustainability.and.Adaptation: num  82.6 75.1 87.4 70.3 69.2 66.1 68.3 60.1 71.1 64.5 ...
```

#Null values after cleaning the dataset

```r
colSums(is.na(data))
```

```
##                          Rank                       Country
##                             0                             0
##                 Overall_Score                 Affordability
##                             0                             0
##                  Availability                Quality_Safety
##                             0                             0
## Sustainability.and.Adaptation
##                             0
```

#PEARSON CORRELATION

```r
# Check the structure of the dataset
data <- read.csv("C:/Users/samee/Downloads/Global Food Security Index 2022.csv")
str(data)
```

```
## 'data.frame':    113 obs. of  7 variables:
##  $ Number                       : chr  "1st" "2nd" "3rd" "4th" ...
##  $ Country                      : chr  "Finland" "Ireland" "Norway" "France" ...
##  $ Overall.Score                : num  83.7 81.7 80.5 80.2 80.1 79.5 79.1 79.1 78.8 78.7 ...
##  $ Affordability                : num  91.9 92.6 87.2 91.3 92.7 89.8 91.9 88.3 91.5 90 ...
##  $ Availability                 : num  70.5 70.5 60.4 69 70.7 81.2 68.3 75.7 71.6 77 ...
##  $ Quality.Safety               : num  88.4 86.1 86.8 87.7 84.7 77.4 85 89.5 77.6 79.8 ...
##  $ Sustainability.and.Adaptation: num  82.6 75.1 87.4 70.3 69.2 66.1 68.3 60.1 71.1 64.5 ...
```

```r
# Correct column names
names(data)[names(data) == "Quality_Safety"] <- "Quality_Safety"

# Perform Pearson correlation analysis
correlation_results <- cor(data[, c("Overall.Score", "Affordability", "Availability","Sustainability.and


# Perform Pearson correlation analysis including the new column "Sustainability.and.Adaptation"
correlation_results <- cor(data[, c("Overall.Score", "Affordability", "Availability", "Quality.Safety",
                           method = "pearson")

# Print correlation results
print(correlation_results)
```

```
##                               Overall.Score Affordability Availability
## Overall.Score                     1.0000000     0.9373218    0.8608440
## Affordability                     0.9373218     1.0000000    0.7460485
## Availability                      0.8608440     0.7460485    1.0000000
## Quality.Safety                    0.9007715     0.7925404    0.7046659
## Sustainability.and.Adaptation     0.7291524     0.5350029    0.5683663
##                               Quality.Safety Sustainability.and.Adaptation
## Overall.Score                      0.9007715                     0.7291524
## Affordability                      0.7925404                     0.5350029
## Availability                       0.7046659                     0.5683663
## Quality.Safety                     1.0000000                     0.6148655
## Sustainability.and.Adaptation      0.6148655                     1.0000000
```

#LINEAR REGRESSION

```r
# Check the structure of the dataset
str(data)
```

```
## 'data.frame':    113 obs. of  7 variables:
##  $ Number                       : chr  "1st" "2nd" "3rd" "4th" ...
##  $ Country                      : chr  "Finland" "Ireland" "Norway" "France" ...
##  $ Overall.Score                : num  83.7 81.7 80.5 80.2 80.1 79.5 79.1 79.1 78.8 78.7 ...
##  $ Affordability                : num  91.9 92.6 87.2 91.3 92.7 89.8 91.9 88.3 91.5 90 ...
##  $ Availability                 : num  70.5 70.5 60.4 69 70.7 81.2 68.3 75.7 71.6 77 ...
##  $ Quality.Safety               : num  88.4 86.1 86.8 87.7 84.7 77.4 85 89.5 77.6 79.8 ...
##  $ Sustainability.and.Adaptation: num  82.6 75.1 87.4 70.3 69.2 66.1 68.3 60.1 71.1 64.5 ...
```

```r
# Perform linear regression analysis
lm_model <- lm(Overall.Score ~ Affordability, data = data)
```

```r
# Print the summary of the linear regression model
summary(lm_model)
```

```
##
## Call:
## lm(formula = Overall.Score ~ Affordability, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.0462  -3.1256   0.6264   2.9869   9.5924
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   20.23455    1.53696   13.16   <2e-16 ***
## Affordability  0.60742    0.02143   28.34   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.432 on 111 degrees of freedom
## Multiple R-squared:  0.8786, Adjusted R-squared:  0.8775
## F-statistic: 803.1 on 1 and 111 DF,  p-value: < 2.2e-16
```

#MULTIPLE REGRESSION

```r
# Check the structure of the dataset
str(data)
```

```
## 'data.frame':    113 obs. of  7 variables:
##  $ Number                       : chr  "1st" "2nd" "3rd" "4th" ...
##  $ Country                      : chr  "Finland" "Ireland" "Norway" "France" ...
##  $ Overall.Score                : num  83.7 81.7 80.5 80.2 80.1 79.5 79.1 79.1 78.8 78.7 ...
##  $ Affordability                : num  91.9 92.6 87.2 91.3 92.7 89.8 91.9 88.3 91.5 90 ...
##  $ Availability                 : num  70.5 70.5 60.4 69 70.7 81.2 68.3 75.7 71.6 77 ...
##  $ Quality.Safety               : num  88.4 86.1 86.8 87.7 84.7 77.4 85 89.5 77.6 79.8 ...
##  $ Sustainability.and.Adaptation: num  82.6 75.1 87.4 70.3 69.2 66.1 68.3 60.1 71.1 64.5 ...
```

```r
# Perform multiple regression analysis
lm_model <- lm(Overall.Score ~ Affordability + Availability + Quality.Safety+Sustainability.and.Adaptati

# Print the summary of the multiple regression model
summary(lm_model)
```

```
##
## Call:
## lm(formula = Overall.Score ~ Affordability + Availability + Quality.Safety +
##     Sustainability.and.Adaptation, data = data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.067551 -0.024210  0.000106  0.021552  0.075264
##
```

```
## Coefficients:
##                              Estimate Std. Error  t value Pr(>|t|)
## (Intercept)                 0.0111279  0.0183247    0.607    0.545
## Affordability               0.2997854  0.0002878 1041.470   <2e-16 ***
## Availability                0.2502815  0.0004380  571.388   <2e-16 ***
## Quality.Safety              0.2250127  0.0003874  580.804   <2e-16 ***
## Sustainability.and.Adaptation 0.2248175  0.0003625  620.135   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03271 on 108 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 4.196e+06 on 4 and 108 DF,  p-value: < 2.2e-16
```

#Logistic Regression Analysis

```r
# Check the structure of the dataset
str(data)
```

```
## 'data.frame':    113 obs. of  7 variables:
##  $ Number                       : chr  "1st" "2nd" "3rd" "4th" ...
##  $ Country                      : chr  "Finland" "Ireland" "Norway" "France" ...
##  $ Overall.Score                : num  83.7 81.7 80.5 80.2 80.1 79.5 79.1 79.1 78.8 78.7 ...
##  $ Affordability                : num  91.9 92.6 87.2 91.3 92.7 89.8 91.9 88.3 91.5 90 ...
##  $ Availability                 : num  70.5 70.5 60.4 69 70.7 81.2 68.3 75.7 71.6 77 ...
##  $ Quality.Safety               : num  88.4 86.1 86.8 87.7 84.7 77.4 85 89.5 77.6 79.8 ...
##  $ Sustainability.and.Adaptation: num  82.6 75.1 87.4 70.3 69.2 66.1 68.3 60.1 71.1 64.5 ...
```

```r
# Assuming 'Quality...safety' is a categorical variable representing food safety ratings
# Convert 'Quality...safety' to a factor if it's not already
data$Quality...safety <- as.factor(data$Quality.Safety)

# Perform logistic regression analysis
log_model <- glm(Quality...safety ~ Affordability + Availability, data = data, family = "binomial")

# Print the summary of the logistic regression model
summary(log_model)
```

```
##
## Call:
## glm(formula = Quality...safety ~ Affordability + Availability,
##     family = "binomial", data = data)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.43367    5.03838  -0.285    0.776
## Affordability  0.07825    0.09212   0.849    0.396
## Availability   0.04220    0.12154   0.347    0.728
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 11.4459  on 112  degrees of freedom
## Residual deviance:  8.9913  on 110  degrees of freedom
```

```
## AIC: 14.991
##
## Number of Fisher Scoring iterations: 9
```

#Analysis of Variance (ANOVA):

```r
# Read the dataset

# Specify the number of intervals
num_intervals <- 3

# Create breaks based on quantiles
breaks <- quantile(data$Overall.Score, probs = seq(0, 1, length.out = num_intervals + 1))

# Create labels for each interval
labels <- c("Low", "Medium", "High")

# Create grouping variable based on Overall Score quartiles
data$Country_Group <- cut(data$Overall.Score, breaks = breaks, labels = labels)

# Perform ANOVA to compare mean food security index across different groups of countries
anova_result <- aov(Overall.Score ~ Country_Group, data = data)

# Print the summary of the ANOVA analysis
summary(anova_result)
```

```
##                Df Sum Sq Mean Sq F value Pr(>F)
## Country_Group   2  15252    7626     409 <2e-16 ***
## Residuals     109   2033      19
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 1 observation deleted due to missingness
```

```r
# Calculate the Pearson correlation coefficient between Affordability and Availability
cor_affordability_availability <- cor(data$Affordability, data$Availability, use = "complete.obs", meth
print(paste("Correlation between Affordability and Availability:", cor_affordability_availability))
```

```
## [1] "Correlation between Affordability and Availability: 0.746048518407468"
```

```r
# Calculate the Pearson correlation coefficient between Affordability and Quality.Safety
cor_affordability_quality_safety <- cor(data$Affordability, data$`Quality.Safety`, use = "complete.obs"
print(paste("Correlation between Affordability and Quality.Safety:", cor_affordability_quality_safety))
```

```
## [1] "Correlation between Affordability and Quality.Safety: 0.792540420972602"
```

```r
# Calculate the Pearson correlation coefficient between Affordability and Sustainability.and.Adaptation
cor_affordability_sustainability_adaptation <- cor(data$Affordability, data$`Sustainability.and.Adaptat
print(paste("Correlation between Affordability and Sustainability.and.Adaptation:", cor_affordability_su
```

```
## [1] "Correlation between Affordability and Sustainability.and.Adaptation: 0.535002940236989"
```

```r
# Calculate the Pearson correlation coefficient between Availability and Quality.Safety
cor_availability_quality_safety <- cor(data$Availability, data$`Quality.Safety`, use = "complete.obs", m
print(paste("Correlation between Availability and Quality.Safety:", cor_availability_quality_safety))
```

```
## [1] "Correlation between Availability and Quality.Safety: 0.704665893038559"
```

```r
# Calculate the Pearson correlation coefficient between Availability and Sustainability.and.Adaptation
cor_availability_sustainability_adaptation <- cor(data$Availability, data$`Sustainability.and.Adaptatio
print(paste("Correlation between Availability and Sustainability.and.Adaptation:", cor_availability_sust
```

```
## [1] "Correlation between Availability and Sustainability.and.Adaptation: 0.568366345197699"
```

```r
# Calculate the Pearson correlation coefficient between Quality.Safety and Sustainability.and.Adaptatio
cor_quality_safety_sustainability_adaptation <- cor(data$`Quality.Safety`, data$`Sustainability.and.Adap
print(paste("Correlation between Quality.Safety and Sustainability.and.Adaptation:", cor_quality_safety_
```

```
## [1] "Correlation between Quality.Safety and Sustainability.and.Adaptation: 0.614865468743303"
```

#shapiro wilk test

```r
# Assuming 'data' is already loaded
# Perform Shapiro-Wilk test for normality on multiple columns
shapiro_results <- lapply(data[, c( "Overall.Score","Affordability", "Availability", "Quality.Safety",
  shapiro.test(column[!is.na(column)])  # Exclude NA values for the test
})

# Print results
print(shapiro_results)
```

```
## $Overall.Score
##
##  Shapiro-Wilk normality test
##
## data:  column[!is.na(column)]
## W = 0.9517, p-value = 0.0004534
##
##
## $Affordability
##
##  Shapiro-Wilk normality test
##
## data:  column[!is.na(column)]
## W = 0.91564, p-value = 2.52e-06
##
##
## $Availability
##
##  Shapiro-Wilk normality test
##
## data:  column[!is.na(column)]
## W = 0.98465, p-value = 0.2241
```

```
##
##
## $Quality.Safety
##
##  Shapiro-Wilk normality test
##
## data:  column[!is.na(column)]
## W = 0.95759, p-value = 0.001232
##
##
## $Sustainability.and.Adaptation
##
##  Shapiro-Wilk normality test
##
## data:  column[!is.na(column)]
## W = 0.98502, p-value = 0.2407
```

#Kruskal-Wallis test

```r
# Categorize 'Overall.Score' into three groups
data$Score_Group <- cut(data$Overall.Score,
                        breaks = quantile(data$Overall.Score, probs = c(0, 1/3, 2/3, 1), na.rm = TRUE),
                        labels = c("Low", "Medium", "High"),
                        include.lowest = TRUE)

# Perform Kruskal-Wallis test to compare 'Affordability' across 'Score_Group'
kruskal_test_affordability <- kruskal.test(Affordability ~ Score_Group, data = data)

# Print the result
print(kruskal_test_affordability)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  Affordability by Score_Group
## Kruskal-Wallis chi-squared = 90.508, df = 2, p-value < 2.2e-16
```

```r
# You can replicate the above test for other variables by changing 'Affordability' to 'Availability', e
```

DATA VISUALIZATION

```r
# Load necessary libraries
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.4.1
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(car)
```

```
## Warning: package 'car' was built under R version 4.4.3
```

```
## Loading required package: carData
```

```
## Warning: package 'carData' was built under R version 4.4.3
```

```
##
## Attaching package: 'car'
```

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

```r
library(multcomp)
```

```
## Warning: package 'multcomp' was built under R version 4.4.3
```

```
## Loading required package: mvtnorm
```

```
## Warning: package 'mvtnorm' was built under R version 4.4.3
```

```
## Loading required package: survival
```

```
## Loading required package: TH.data
```

```
## Warning: package 'TH.data' was built under R version 4.4.3
```

```
## Loading required package: MASS
```

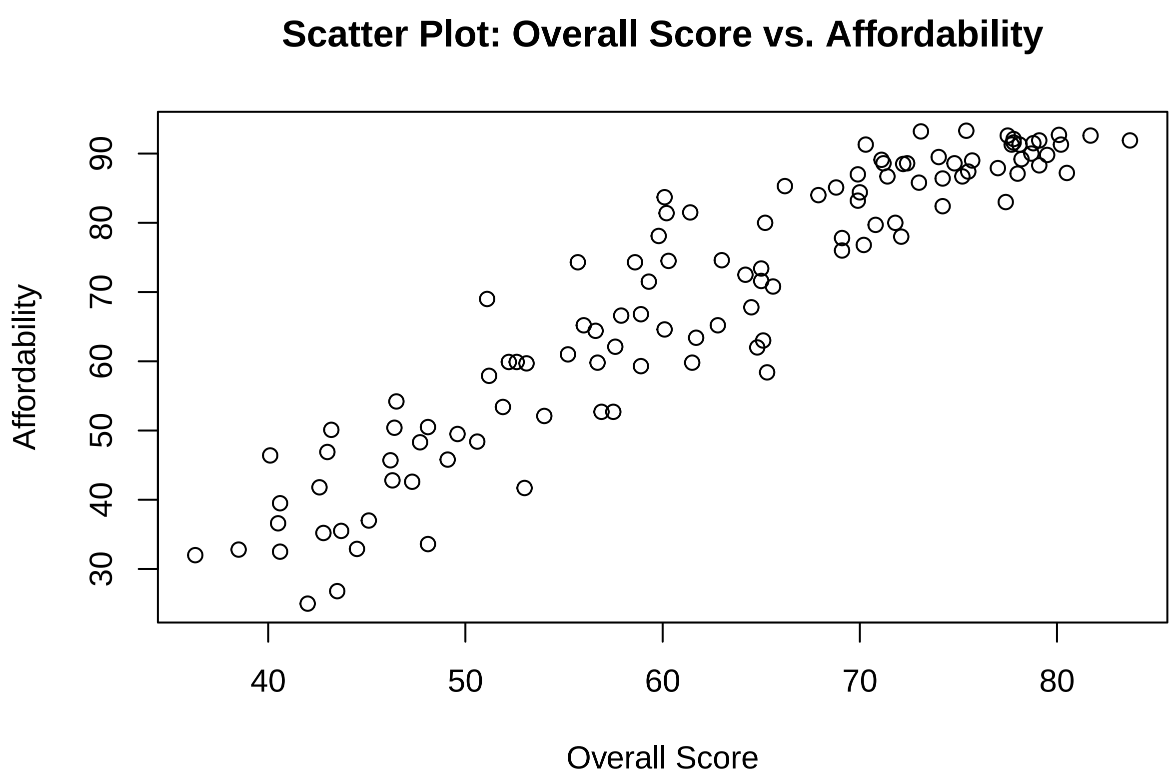
```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
##
## Attaching package: 'TH.data'
```

```
## The following object is masked from 'package:MASS':
##
##     geyser
```

```r
# Scatter plot between Overall Score and Affordability
plot(data$Overall.Score, data$Affordability,
     xlab = "Overall Score", ylab = "Affordability",
     main = "Scatter Plot: Overall Score vs. Affordability")
```




```r
library(ggplot2)
```

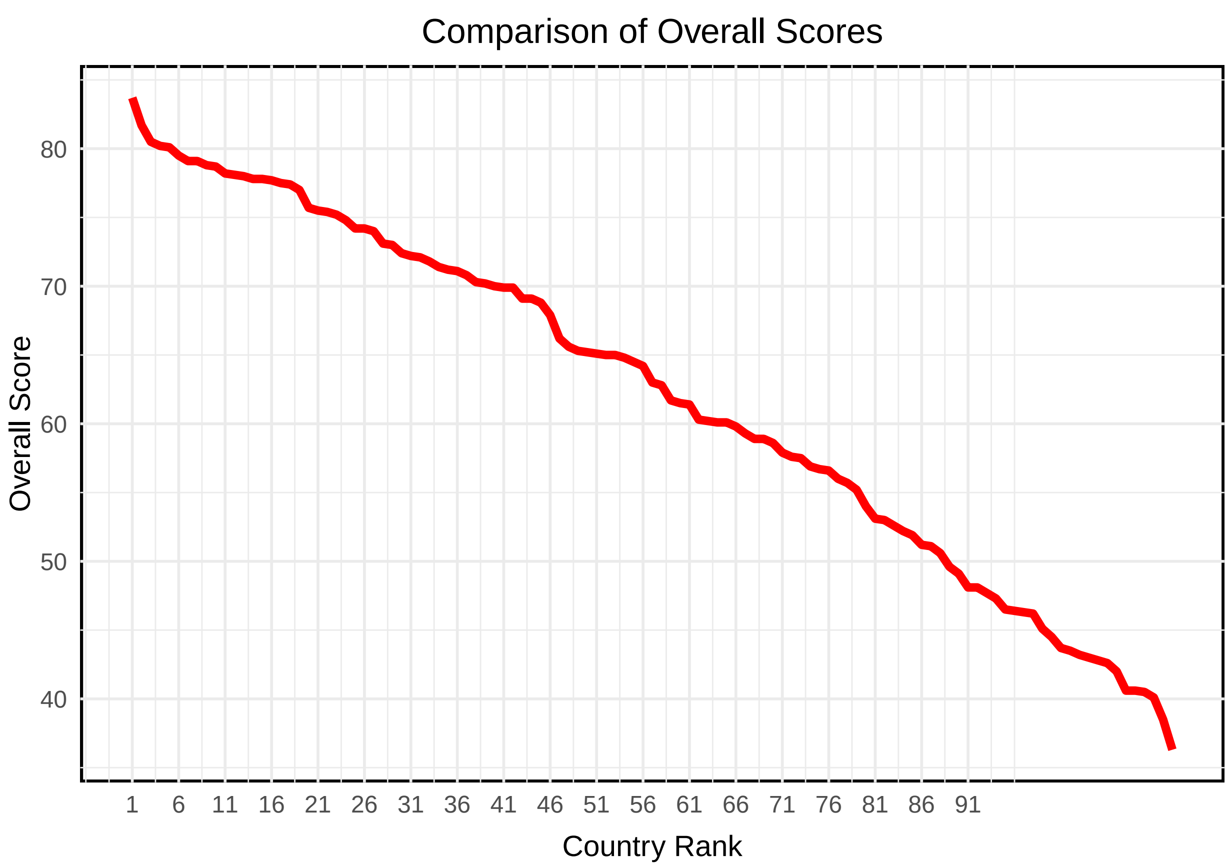
```
## Warning: package 'ggplot2' was built under R version 4.4.1
```

```r
# Assuming your data is sorted in the order you want to use for ranking
# Add a 'Country Rank' column if it does not exist
data$Country.Rank <- seq_along(data$`Overall.Score`)

# Create the line plot with enhanced visualization
ggplot(data, aes(x = Country.Rank, y = `Overall.Score`)) +
  geom_line(color = "red", size = 1.5) +  # Blue line with increased width for better visibility
  scale_x_continuous(breaks = seq(1, 95, by = 5)) +  # Adjust the x-axis limits and breaks
  labs(
    x = "Country Rank",
    y = "Overall Score",
    title = "Comparison of Overall Scores"
  ) +
  theme_minimal() +  # Using a minimal theme for a clean look
  theme(
```

```
    plot.title = element_text(hjust = 0.5),  # Center the title
    axis.title.x = element_text(vjust = -0.5),  # Adjust the x-axis label position
    axis.title.y = element_text(vjust = 0.5),  # Adjust the y-axis label position
    panel.background = element_rect(fill = "white", colour = "black", size = 1), # White background wit
    plot.background = element_rect(fill = "white", colour = NA) # Light gray plot background
  )
```

```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use 'linewidth' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```

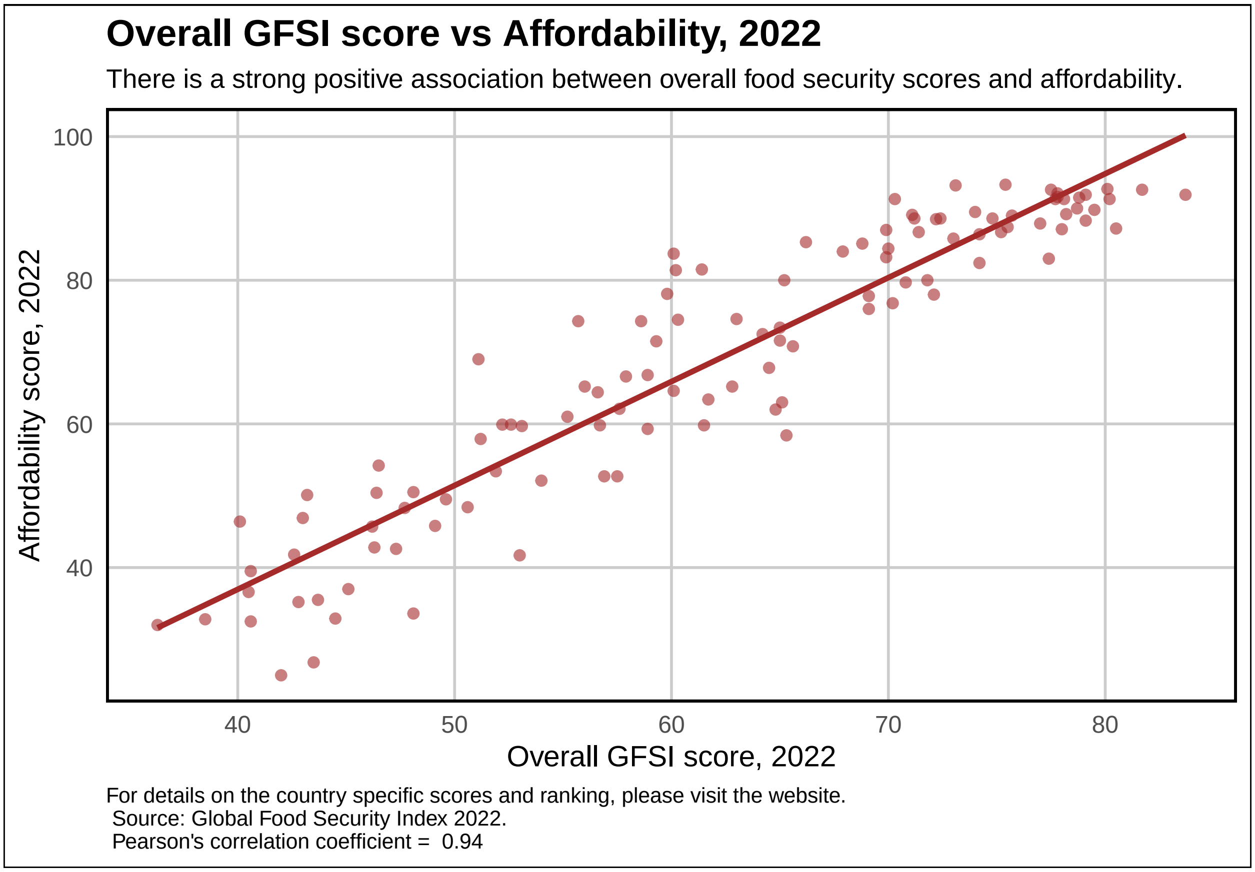
```
## Warning: The 'size' argument of 'element_rect()' is deprecated as of ggplot2 3.4.0.
## i Please use the 'linewidth' argument instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```




```
library(ggplot2)

# Assuming your data frame 'data' is already loaded and has the correct types for numeric columns

# Calculate Pearson's correlation coefficient for Overall.Score and Affordability
```

```r
correlation_coefficient <- cor(data$Overall.Score, data$Affordability, use = "complete.obs")

# Create the scatter plot with a trend line for Overall.Score vs Affordability
ggplot_object <- ggplot(data, aes(x = Overall.Score, y = Affordability)) +
  geom_point(color = "brown", alpha = 0.6) +  # Blue points with some transparency
  geom_smooth(method = "lm", color = "brown", se = FALSE) +  # Add a linear regression line without sta
  labs(title = "Overall GFSI score vs Affordability, 2022",
       subtitle = "There is a strong positive association between overall food security scores and affo
       x = "Overall GFSI score, 2022",
       y = "Affordability score, 2022",
       caption = paste("For details on the country specific scores and ranking, please visit the websit
                       "Source: Global Food Security Index 2022.\n",
                       "Pearson's correlation coefficient = ", round(correlation_coefficient, 2))) +
  theme_minimal() +
  theme(plot.title = element_text(size = 14, face = "bold"),
        plot.subtitle = element_text(size = 10),
        plot.caption = element_text(size = 8, hjust = 0),
        plot.background = element_rect(fill = "white"),
        panel.grid.major = element_line(color = "grey80"),
        panel.grid.minor = element_blank(),
        panel.border = element_rect(color = "black", fill = NA, size = 1),
        legend.position = "none")  # Remove the legend

# Print the plot
print(ggplot_object)
```

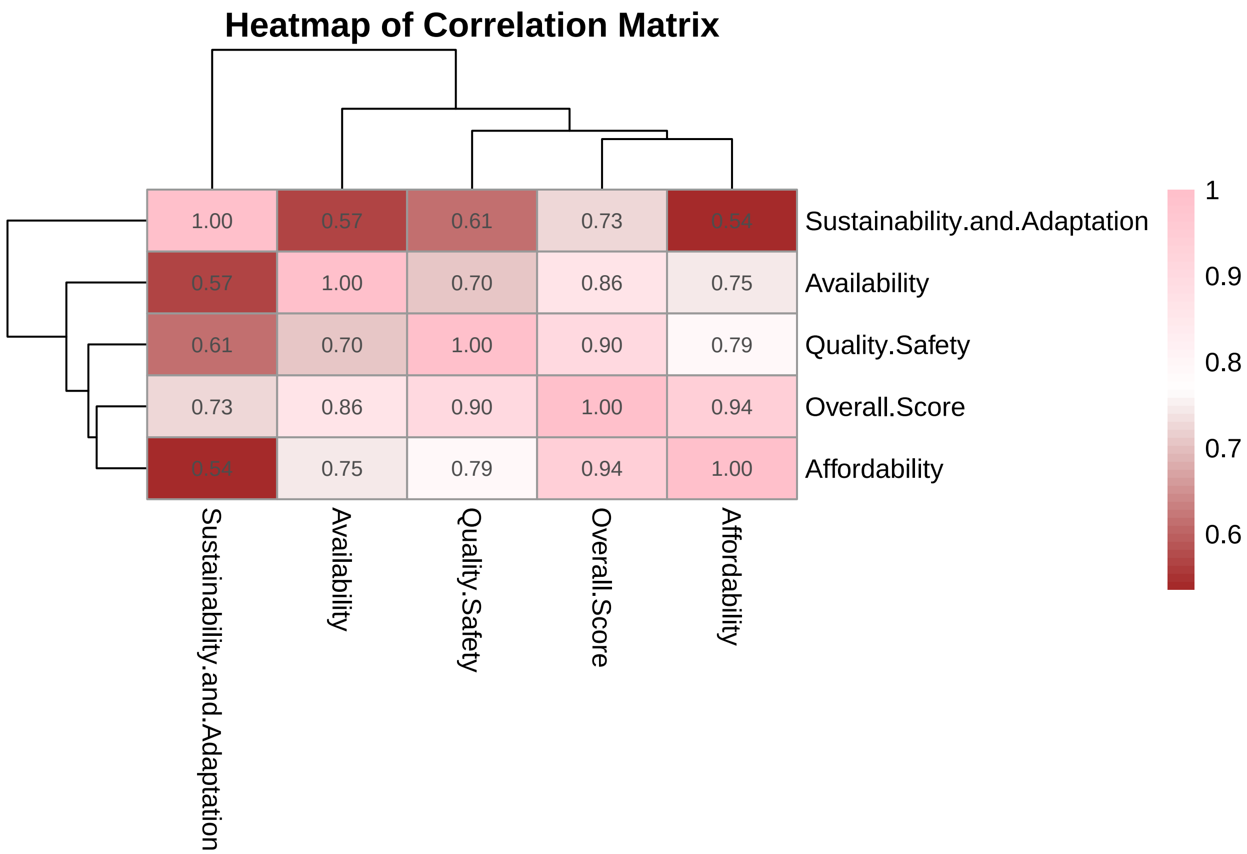
```
## `geom_smooth()` using formula = 'y ~ x'
```




```r
library(pheatmap)
```

```
## Warning: package 'pheatmap' was built under R version 4.4.3
```

```r
# Compute the correlation matrix
correlation_matrix <- cor(data[, c("Overall.Score", "Affordability", "Availability", "Sustainability.and

# Define a color palette from red (negative correlation) to white (zero correlation) to blue (positive
color_palette <- colorRampPalette(c("brown", "white", "pink"))(50)

# Create the heatmap
pheatmap(correlation_matrix,
         color = color_palette,
         display_numbers = TRUE,  # Show correlation values on the heatmap
         clustering_distance_rows = "euclidean",
         clustering_distance_cols = "euclidean",
         clustering_method = "complete",
         main = "Heatmap of Correlation Matrix",
         xlab = "Food Security Index Components",
         ylab = "Food Security Index Components")
```



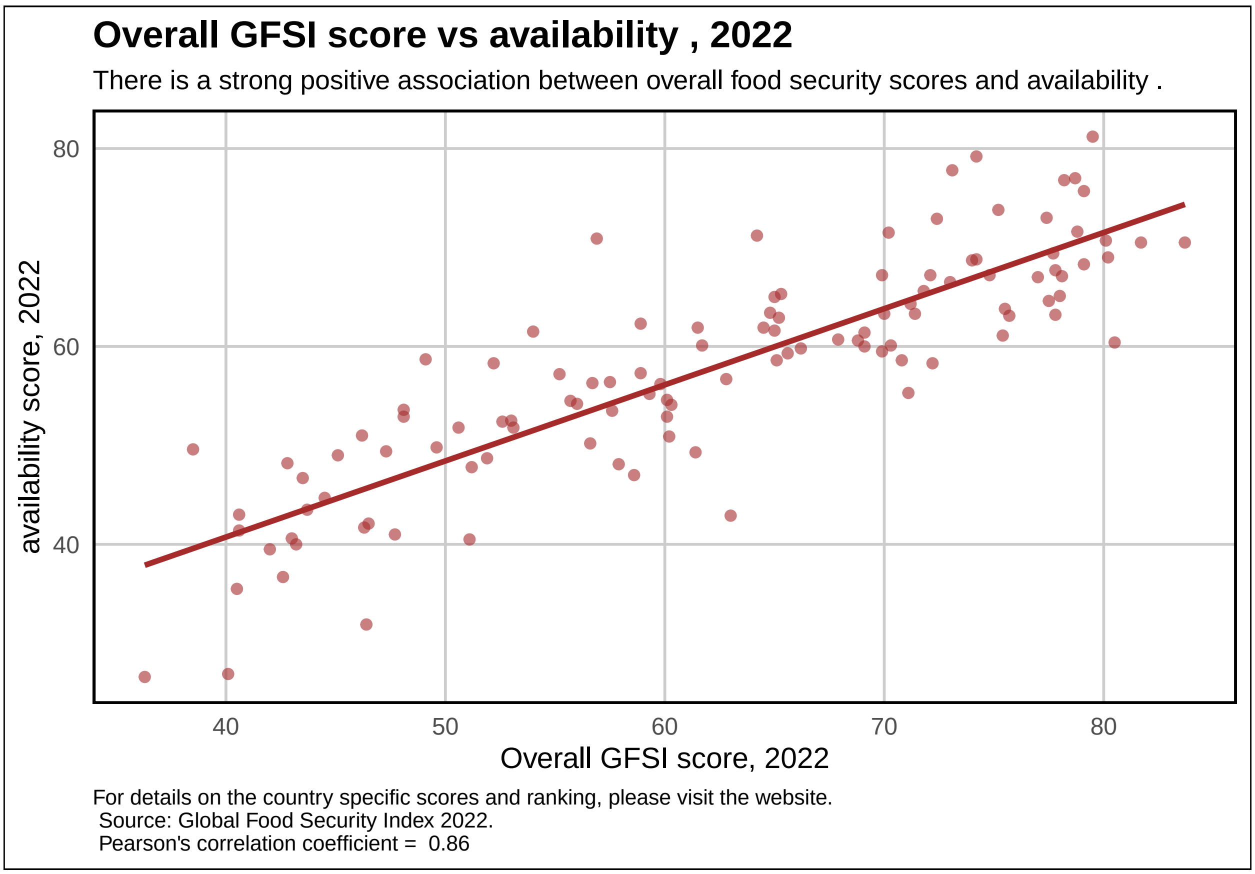

```
library(ggplot2)

# Define a function to create a scatter plot with Overall.Score for a given comparison variable
create_scatter_plot <- function(data, comparison_var, comparison_label) {
  correlation_coefficient <- cor(data$Overall.Score, data[[comparison_var]], use = "complete.obs")

  ggplot(data, aes_string(x = "Overall.Score", y = comparison_var)) +
    geom_point(color = "brown", alpha = 0.6) +
    geom_smooth(method = "lm", color = "brown", se = FALSE) +
    labs(title = paste("Overall GFSI score vs", comparison_label, ", 2022"),
         subtitle = paste("There is a strong positive association between overall food security scores a
         x = "Overall GFSI score, 2022",
         y = paste(comparison_label, "score, 2022"),
         caption = paste("For details on the country specific scores and ranking, please visit the webs
                         "Source: Global Food Security Index 2022.\n",
                         "Pearson's correlation coefficient = ", round(correlation_coefficient, 2))) +
    theme_minimal() +
    theme(plot.title = element_text(size = 14, face = "bold"),
          plot.subtitle = element_text(size = 10),
          plot.caption = element_text(size = 8, hjust = 0),
          plot.background = element_rect(fill = "white"),
          panel.grid.major = element_line(color = "grey80"),
          panel.grid.minor = element_blank(),
          panel.border = element_rect(color = "black", fill = NA, size = 1),
          legend.position = "none")
}
```

```r
# Create scatter plot for Overall.Score vs Availability
scatter_plot_availability <- create_scatter_plot(data, "Availability", "availability")
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

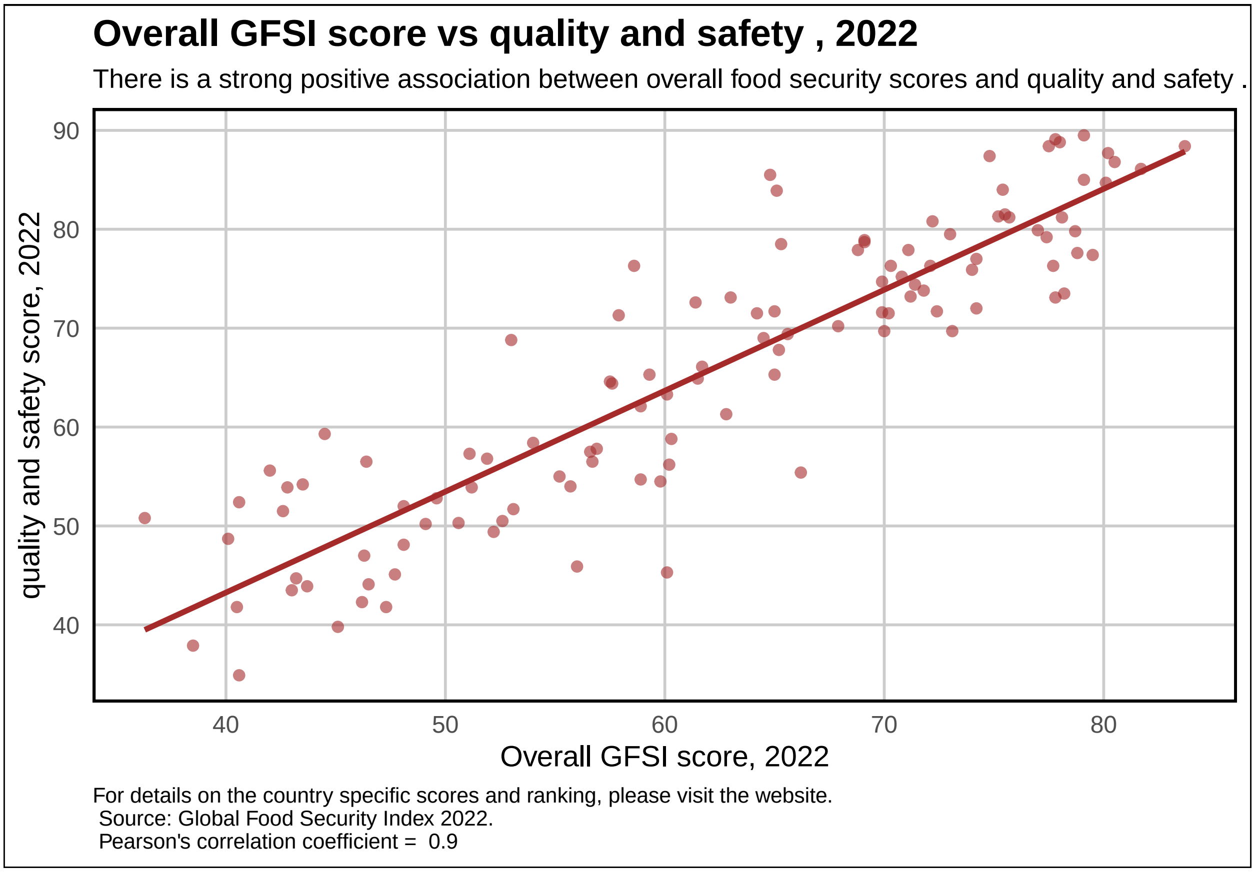
```r
print(scatter_plot_availability)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



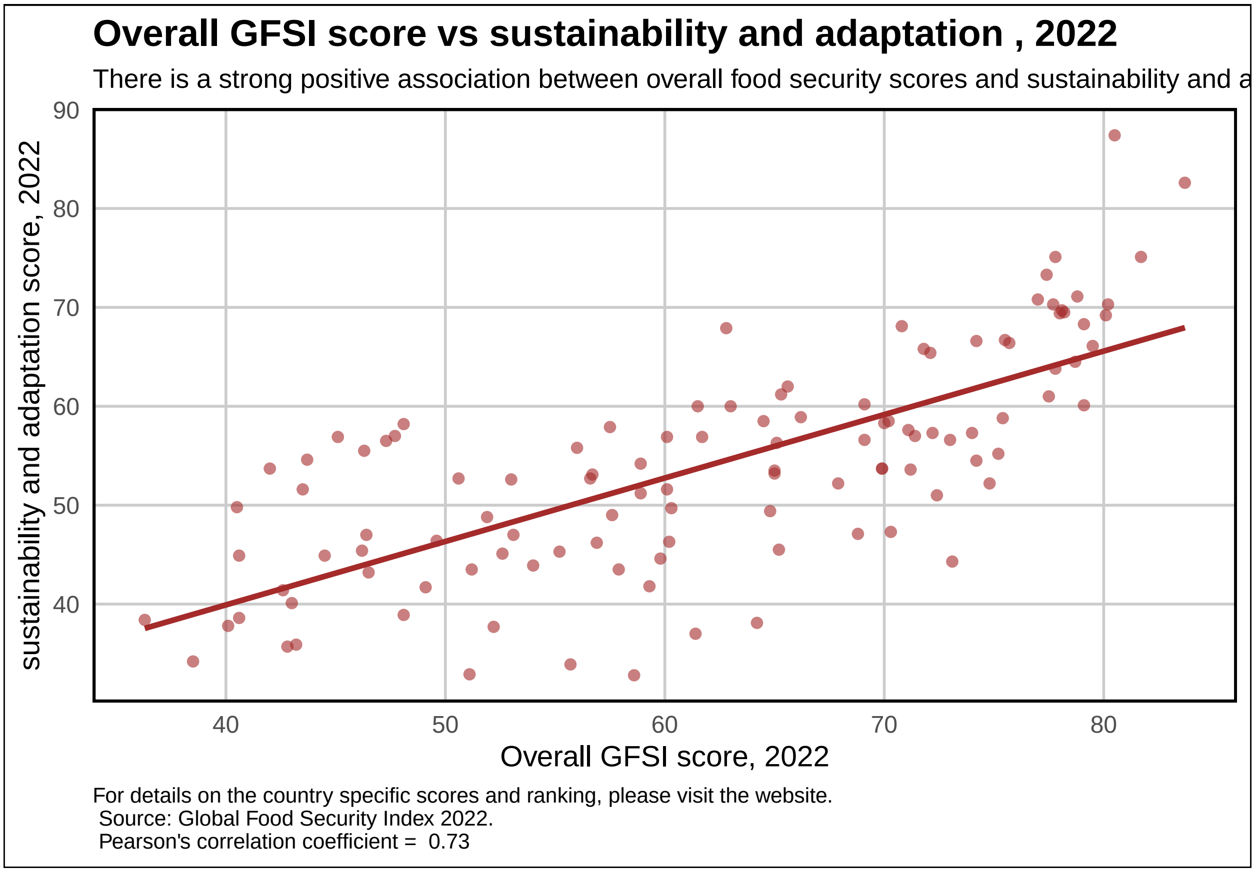

```r
# Create scatter plot for Overall.Score vs Quality.Safety
scatter_plot_quality_safety <- create_scatter_plot(data, "Quality.Safety", "quality and safety")
print(scatter_plot_quality_safety)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

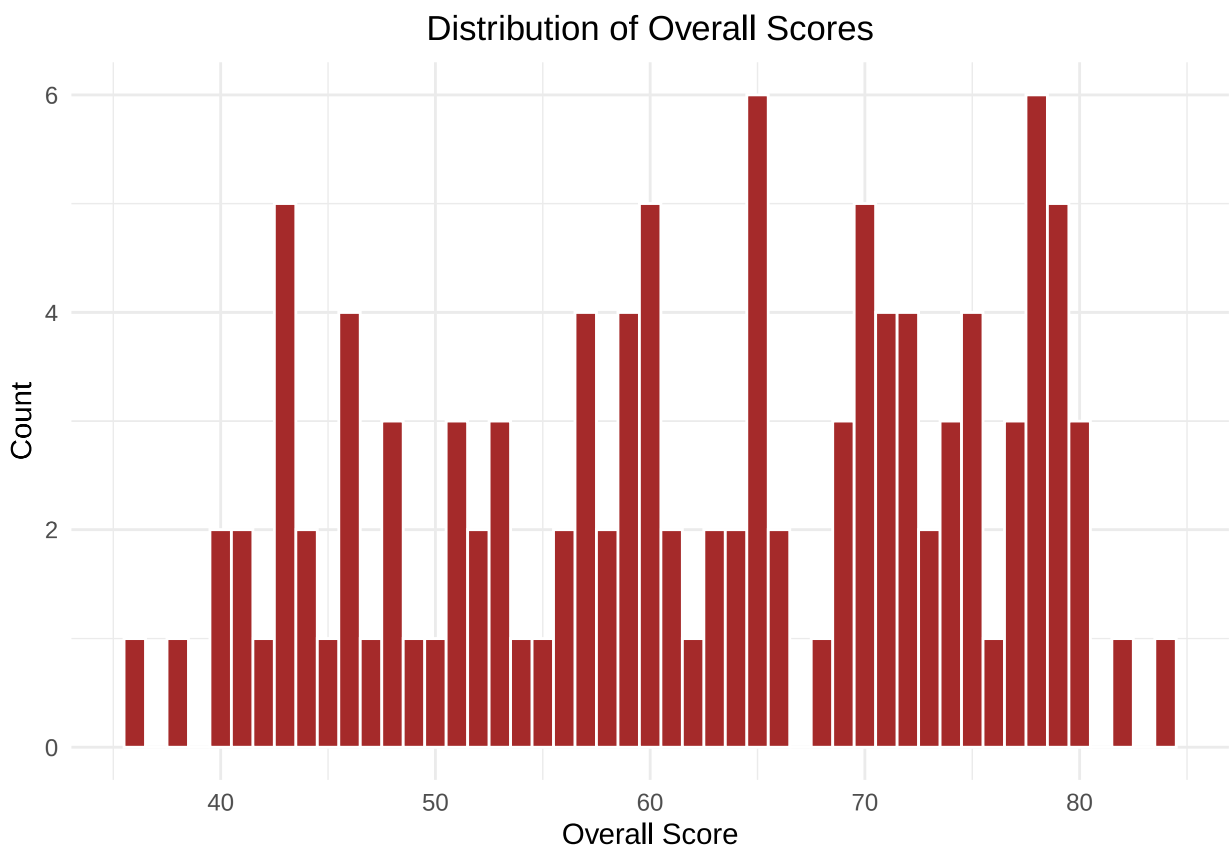



```r
# Create scatter plot for Overall.Score vs Sustainability.and.Adaptation
scatter_plot_sustainability_adaptation <- create_scatter_plot(data, "Sustainability.and.Adaptation", "su
print(scatter_plot_sustainability_adaptation)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```r
library(ggplot2)

# Create histogram using ggplot2
ggplot(data, aes(x = `Overall.Score`)) +  # Ensure correct backticks if needed
  geom_histogram(binwidth = 1, fill = "brown", color = "white") +
  labs(title = "Distribution of Overall Scores", x = "Overall Score", y = "Count") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))  # Correct usage for centering the title
```

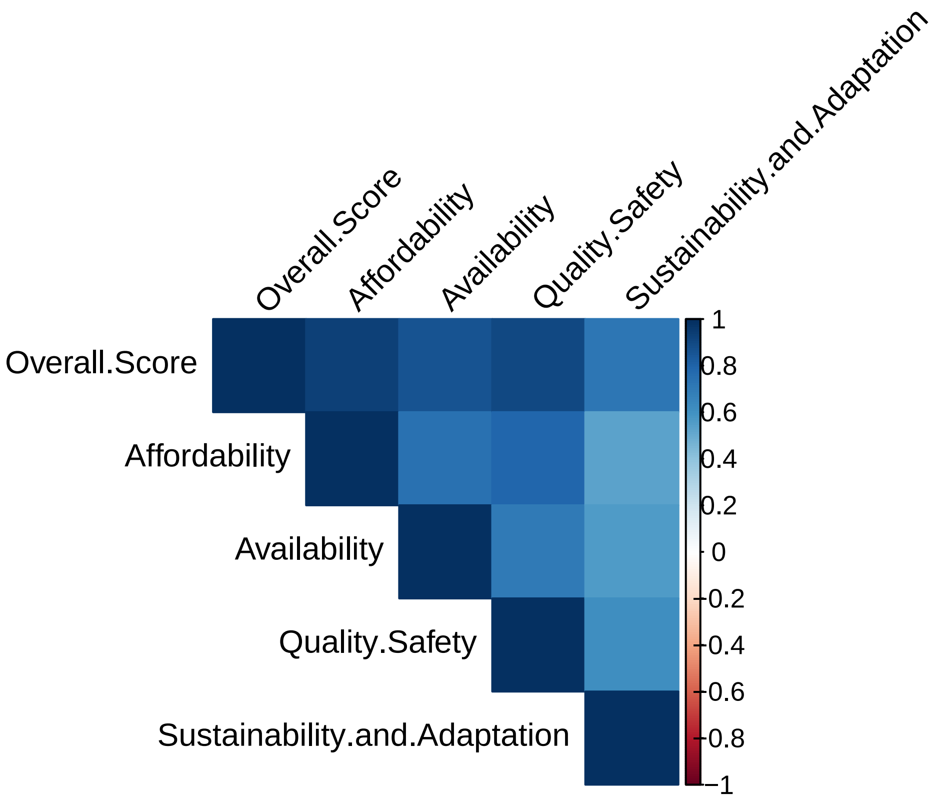



```r
library(corrplot)
```

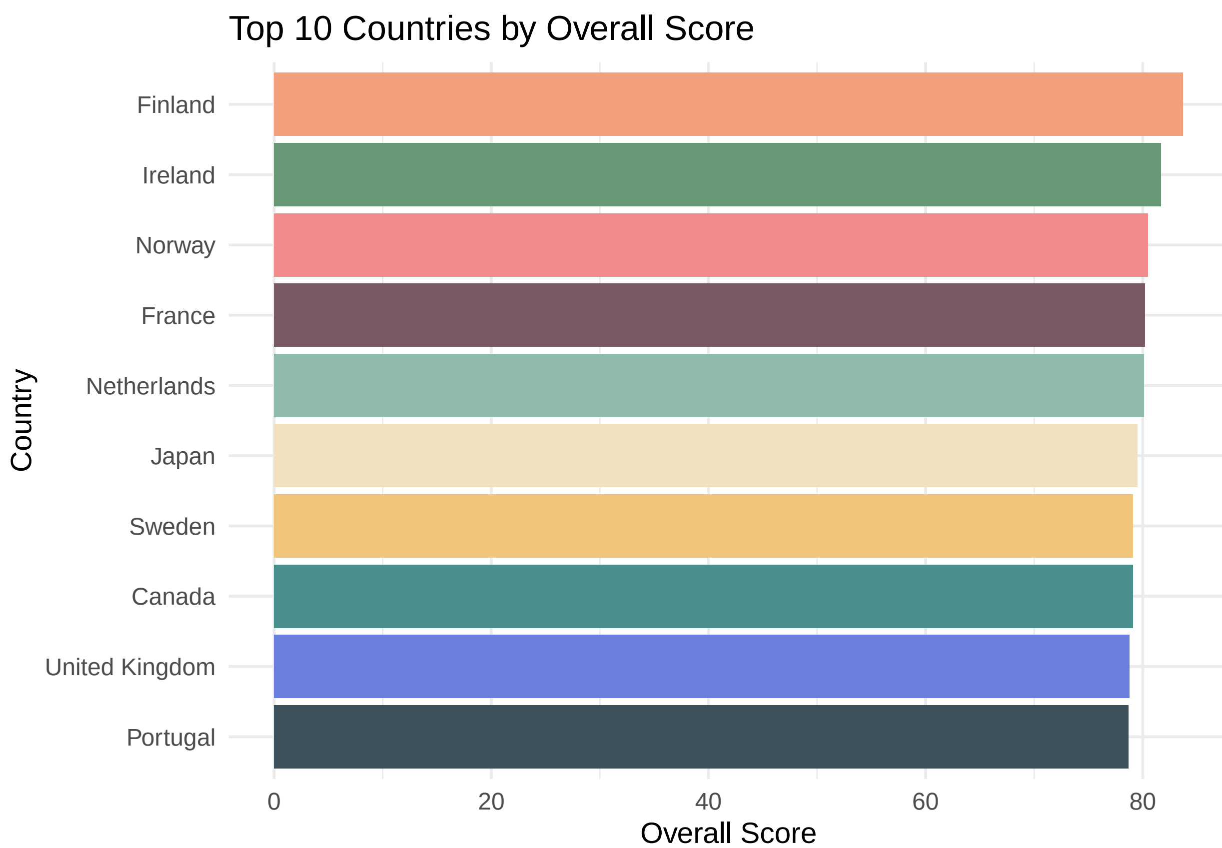
```
## Warning: package 'corrplot' was built under R version 4.4.3
```

```
## corrplot 0.95 loaded
```

```r
numeric_data <- data[, c("Overall.Score", "Affordability", "Availability", "Quality.Safety", "Sustainabi
cor_matrix <- cor(numeric_data)

corrplot(cor_matrix, method = "color", type = "upper", tl.col = "black", tl.srt = 45)
```



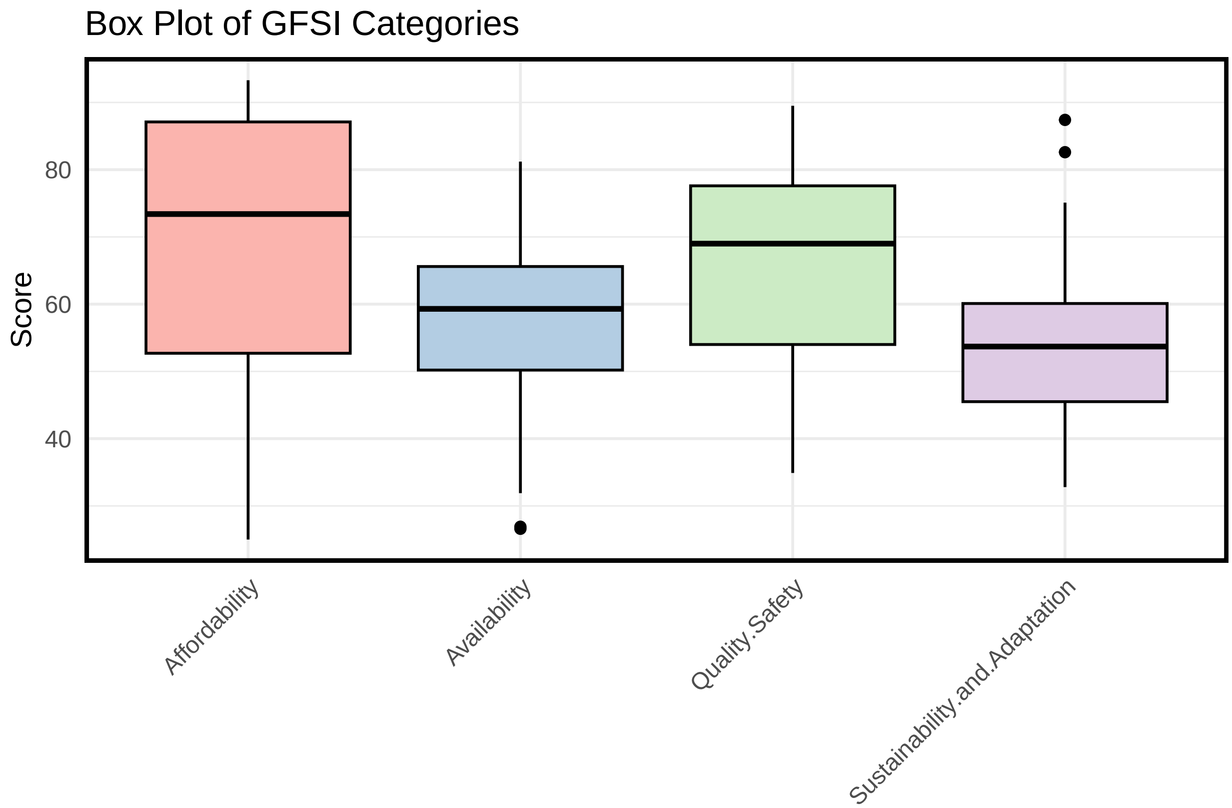

```r
library(ggplot2)

# Assuming data is your dataframe and Overall.Score is the column of interest.

# Get the top 10 countries by Overall Score
top_countries <- data[order(-data$Overall.Score),][1:10,]

color_palette <- c("#4B8F8C", "#F2A07B", "#785964", "#689775", "#F2E1C1",
                   "#8FB9AA", "#F28C8C", "#3E505B", "#F2C57C", "#6A7FDB")

# Create the ggplot bar chart
ggplot(top_countries, aes(x = reorder(Country, Overall.Score), y = Overall.Score, fill = Country)) +
  geom_bar(stat = "identity") +
  coord_flip() +  # Makes it horizontal for better readability
  scale_fill_manual(values = color_palette) +  # Use the custom color palette
  labs(title = "Top 10 Countries by Overall Score", x = "Country", y = "Overall Score") +
  theme_minimal() +  # Use a minimal theme for a clean look
  theme(legend.position = "none")  # Remove the legend as the country names are on the axis
```



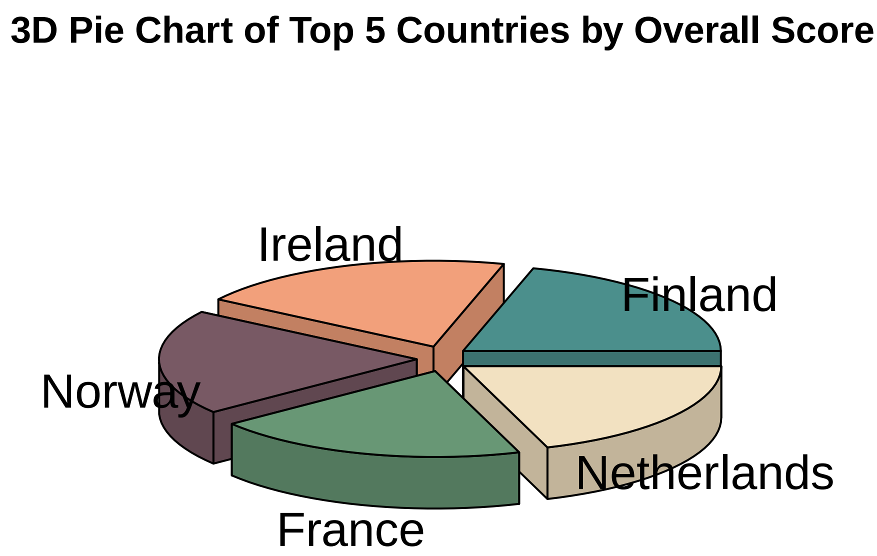

```r
library(ggplot2)
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.4.3
```

```r
# Reshape the data from wide to long format
long_data <- gather(data, key = "Category", value = "Score",
                    Affordability, Availability, `Quality.Safety`, `Sustainability.and.Adaptation`)

# Create box plots for all categories
ggplot_object <- ggplot(long_data, aes(x = Category, y = Score, fill = Category)) +
  geom_boxplot(color = "black") +
  scale_fill_brewer(palette = "Pastel1") + # Use a color palette for aesthetics
  labs(title = "Box Plot of GFSI Categories", y = "Score", x = "") +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1), # Rotate x-axis text for better legibility
    legend.position = "none", # Remove the legend if not needed
    panel.border = element_rect(linetype = "solid", colour = "black", size = 1.5, fill = NA), # Add a b
    panel.background = element_rect(fill = "white"), # Set panel background to white
    plot.background = element_rect(fill = "white", colour = NA) # Set plot background to white (remove f
  )

# Print the plot
print(ggplot_object)
```




```
library(plotrix)

# Assuming data is your dataframe and Overall_Score is the column of interest.

# Sort the data by Overall Score and take the top 5
top_countries <- head(data[order(-data$Overall.Score), ], 5)

# Create a custom color palette
colors <- c("#4B8F8C", "#F2A07B", "#785964", "#689775", "#F2E1C1")

# Create a 3D pie chart for the top 5 countries based on Overall Score
pie3D(top_countries$Overall.Score,
      labels = top_countries$Country,
      main = "3D Pie Chart of Top 5 Countries by Overall Score",
      explode = 0.1,  # This creates a slight separation between the slices
      col = colors)
```




```r
library(ggplot2)
library(dplyr)
library(rnaturalearth)
```

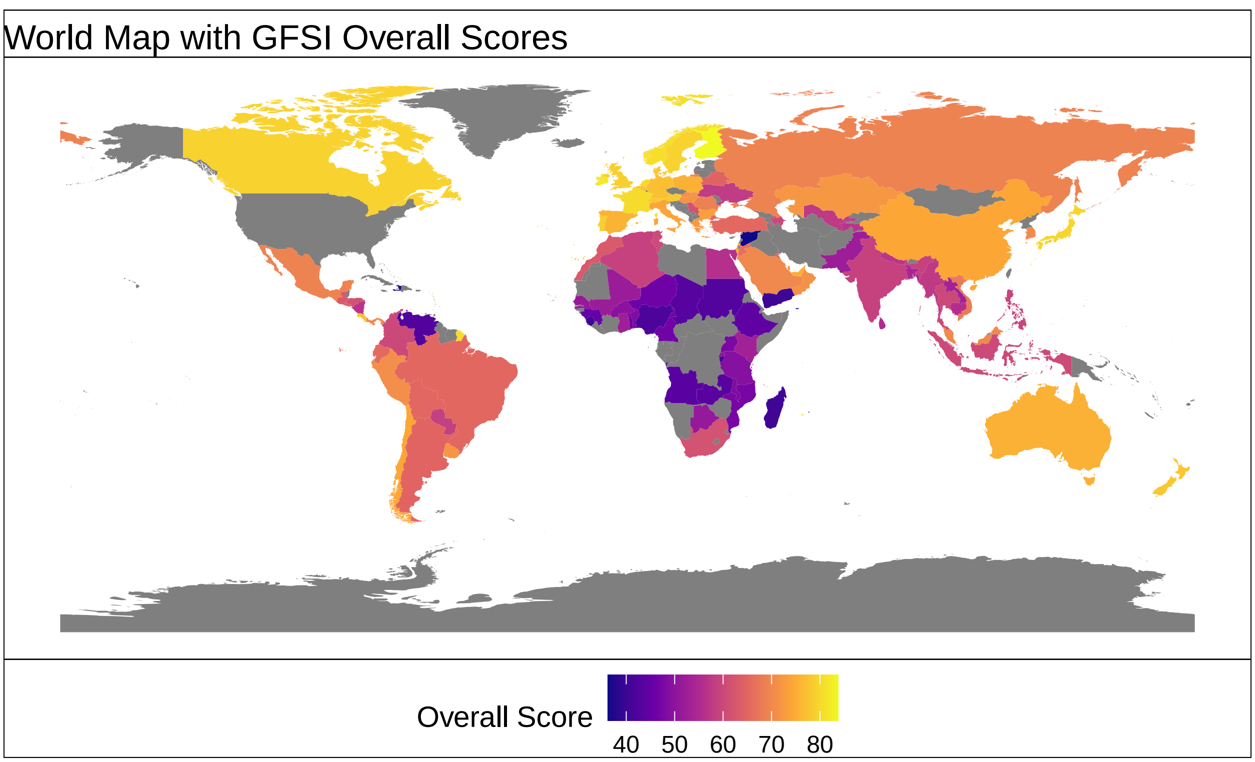
```
## Warning: package 'rnaturalearth' was built under R version 4.4.3
```

```r
library(rnaturalearthdata)
```

```
## Warning: package 'rnaturalearthdata' was built under R version 4.4.3

##
## Attaching package: 'rnaturalearthdata'

## The following object is masked from 'package:rnaturalearth':
##
##     countries110
```

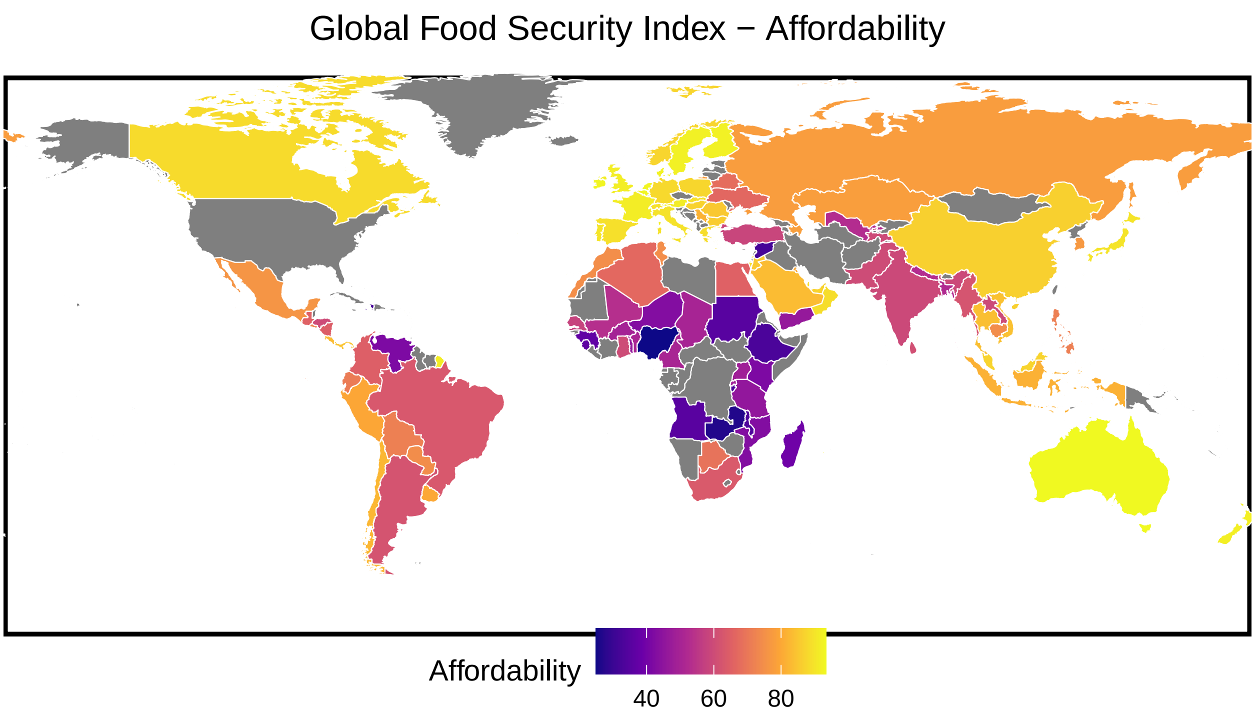
```r
# Load your dataset
# Your dataset should be already loaded into a variable called `data`
# And it must have a column 'Country' that contains the country names

# Get world map data
world <- ne_countries(scale = "medium", returnclass = "sf")
```

```r
# Ensure that the country names in your data match the names in the world map data
# You might need to clean or transform the country names in your dataset for them to match
# This is a crucial step and may require specific adjustments depending on your data

# For demonstration, let's assume the country names match exactly and can be directly joined
world_data <- left_join(world, data, by = c("name" = "Country"))

# Plot the world map with Overall.Score
ggplot_object <- ggplot(data = world_data) +
  geom_sf(aes(fill = Overall.Score), color = NA) + # Fill countries based on Overall.Score
  scale_fill_viridis_c(option = "C") + # Use viridis color scale
  labs(fill = "Overall Score",
       title = "World Map with GFSI Overall Scores") +
  theme_void() +  # A clean theme without axes and grids
  theme(legend.position = "bottom",
        plot.background = element_rect(fill = "white"),
        panel.border = element_rect(color = "black", fill = NA))

# Print the map
print(ggplot_object)
```

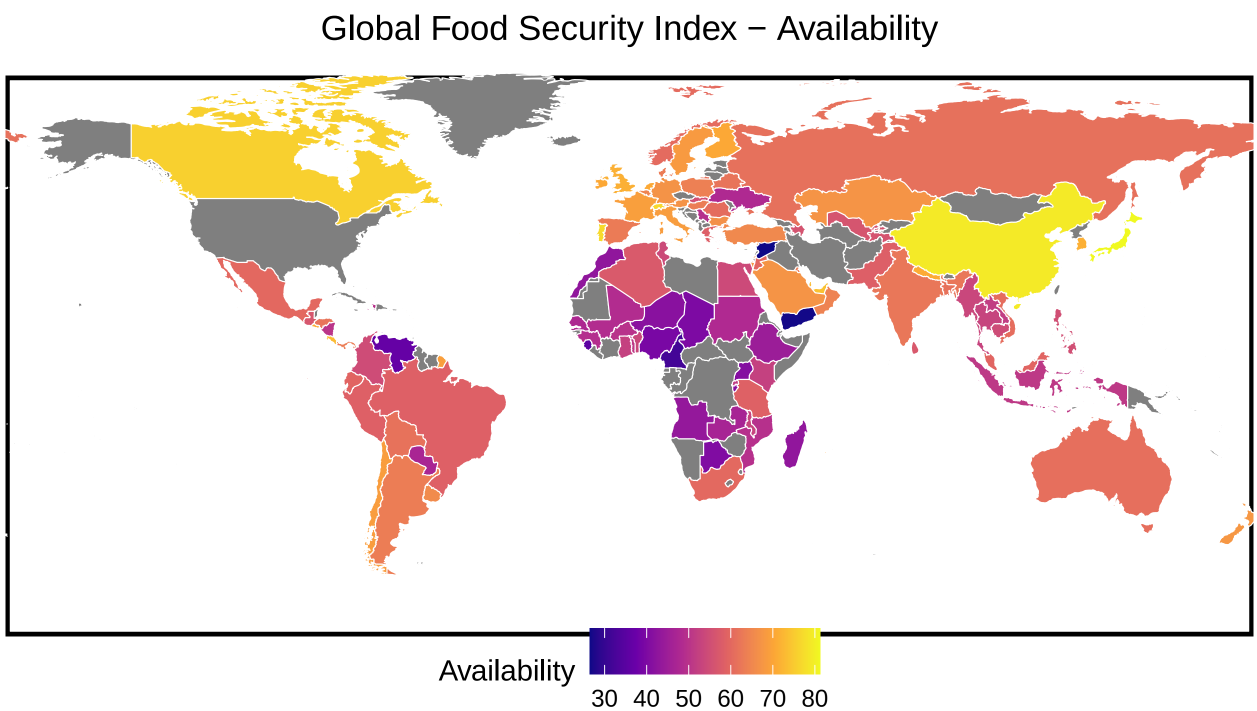



```r
library(ggplot2)
library(dplyr)
library(rnaturalearth)
library(rnaturalearthdata)
```

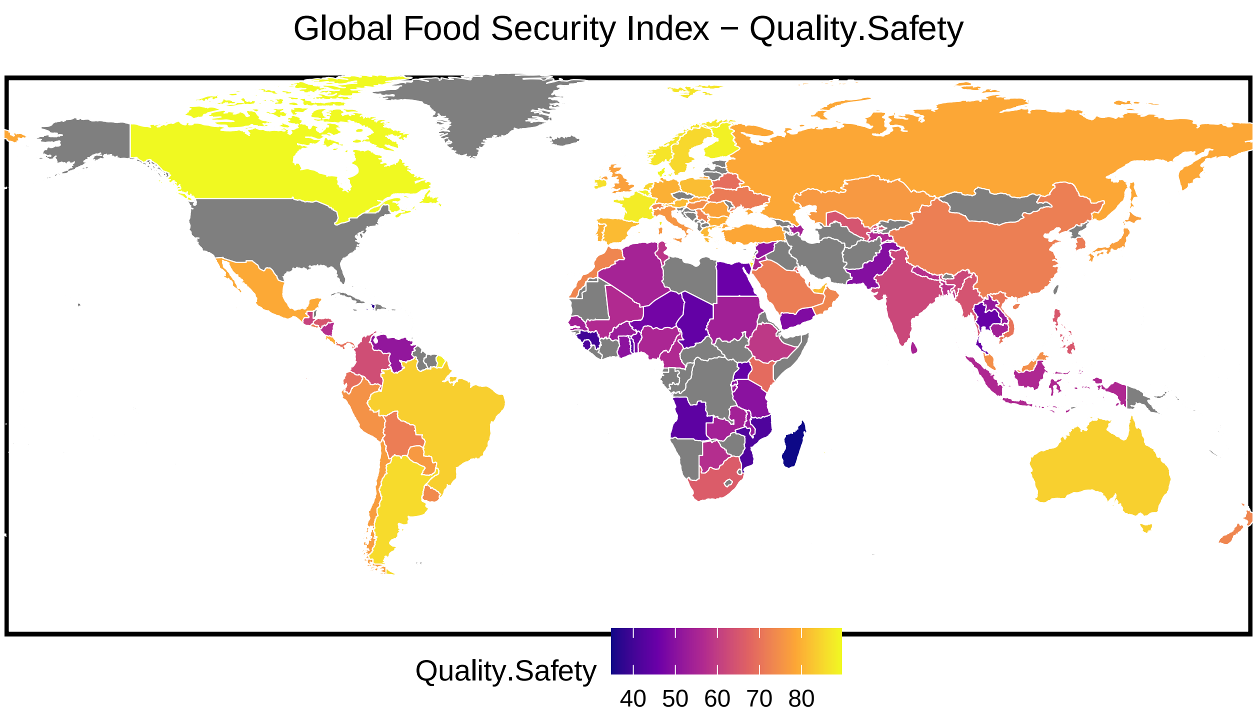
```r
# Load the world map, exclude Antarctica
world <- ne_countries(scale = "medium", returnclass = "sf") %>%
  filter(name != "Antarctica")

# Prepare a list of the variables to loop through
variables <- c("Affordability", "Availability", "Quality.Safety", "Sustainability.and.Adaptation")

# Create a function to generate a map given a score column
generate_map <- function(score_column, title) {
  # Merge your dataset with the world map data on the country names
  # Make sure the country names in your dataset match those in the world dataset
  world_data <- left_join(world, data, by = c("name" = "Country"))

  # Create the map
  p <- ggplot(data = world_data) +
    geom_sf(aes_string(fill = score_column), color = "white") +
    scale_fill_viridis_c(
      name = title,
      na.value = "grey50",  # Color for NA values
      option = "C"
    ) +
    labs(title = paste("World Map with GFSI", title, "Scores")) +
    ggtitle(paste("Global Food Security Index -", title)) +  # Adding title at the top
    theme_void() +
    theme(
      legend.position = "bottom",
      plot.margin = unit(rep(-1, 4), "cm"),
      plot.background = element_rect(fill = "transparent", colour = "black", size = 1.5), # Adding a bo
      plot.title = element_text(hjust = 0.5) # Centering the title
    )

  # Return the plot
  return(p)
}

# Loop through the variables and create a map for each
maps_list <- lapply(variables, function(var) {
  generate_map(var, var)
})

# Print the maps
print(maps_list[[1]]) # Affordability
```



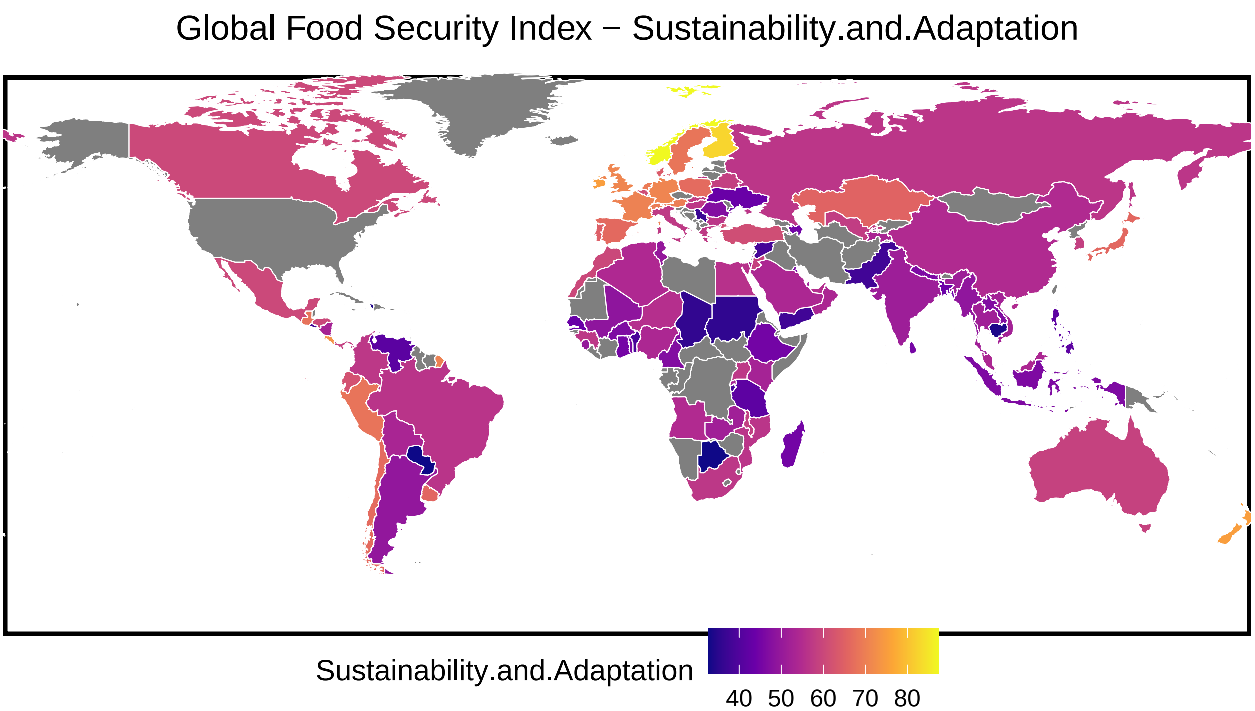

```
print(maps_list[[2]]) # Availability
```




```r
print(maps_list[[3]]) # Quality.Safety
```




```
print(maps_list[[4]]) # Sustainability.and.Adaptation
```




```r
# Post-hoc test using Tukey HSD test
if (summary(anova_result)[[1]][["Pr(>F)"]][1] < 0.05) {
  tukey_result <- TukeyHSD(anova_result)
  print(tukey_result)
}
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Overall.Score ~ Country_Group, data = data)
##
## $Country_Group
##                 diff      lwr      upr p adj
## Medium-Low  15.67027 13.28466 18.05588     0
## High-Low    28.48826 26.11840 30.85813     0
## High-Medium 12.81799 10.44813 15.18786     0
```

```r
library(dplyr)
library(car)
library(multcomp)

# Assuming 'data' is your dataframe and 'Overall_Score' is a variable in the dataframe
# Convert Overall_Score to numeric if it's not already numeric
# Make sure to handle any non-numeric characters that might be present
data$Overall.Score <- as.numeric(as.character(data$Overall.Score))
```

```r
# Handle possible conversion warnings or errors
if(any(is.na(data$Overall_Score))){
  warning("NAs introduced by coercion")
}

# Use cut to create Economic_Group based on quantiles of Overall_Score
data <- mutate(data, Economic_Group = cut(data$Overall.Score,
                                          breaks = quantile(data$Overall.Score, probs = c(0, 1/3, 2/3, 1
                                          labels = c("Low", "Middle", "High"),
                                          include.lowest = TRUE))

# Levene's test for homogeneity of variances
leveneTest_result <- leveneTest(Affordability ~ Economic_Group, data = data)
print(leveneTest_result)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value    Pr(>F)
## group   2  18.292 1.386e-07 ***
##       110
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Fit ANOVA model
anova_model <- aov(Affordability ~ Economic_Group, data = data)

# Check for normality of residuals using Shapiro-Wilk test
shapiro_test_res <- shapiro.test(residuals(anova_model))
print(shapiro_test_res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(anova_model)
## W = 0.98516, p-value = 0.2477
```

```r
# ANOVA
anova_summary <- summary(anova_model)
print(anova_summary)
```

```
##                 Df Sum Sq Mean Sq F value Pr(>F)
## Economic_Group   2  33433   16717     197 <2e-16 ***
## Residuals      110   9333      85
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Post-hoc Tukey HSD test if ANOVA is significant
if(anova_summary[[1]][["Pr(>F)"]][1] < 0.05){
  post_hoc <- glht(anova_model, linfct = mcp(Economic_Group = "Tukey"))
  post_hoc_summary <- summary(post_hoc)
  print(post_hoc_summary)
}
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = Affordability ~ Economic_Group, data = data)
##
## Linear Hypotheses:
##                    Estimate Std. Error t value Pr(>|t|)
## Middle - Low == 0    24.791      2.127  11.653   <1e-10 ***
## High - Low == 0      41.705      2.113  19.736   <1e-10 ***
## High - Middle == 0   16.914      2.127   7.951   <1e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```r
library(ggplot2)
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.4.3
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(RColorBrewer)

# Define your data frames
df1 <- data.frame(Category = c("Change in average food costs", "Proportion of population under global po
                               "Inequality-adjusted income index", "Agricultural trade", "Food safety ne
                  Score = c(48.5, 96.2, 63.1, 50.0, 79.3),
                  Change = c(-50.5, NA, NA, NA, NA))

df2 <- data.frame(Category = c("Access to agricultural inputs", "Agricultural research and development"
                               "Farm infrastructure", "Volatility of agricultural production", "Food los
                               "Supply chain infrastructure", "Sufficiency of supply",
                               "Political and social barriers to access", "Food security and access poli
                  Score = c(49.0, 43.1, 54.1, 78.2, 58.2, 34.4, 100.0, 42.1, 52.5),
                  Change = c(4.5, 7.1, -0.4, -3.4, -0.4, NA, NA, -8.4, NA))

df3 <- data.frame(Category = c("Dietary diversity", "Nutritional standards", "Micronutrient availabilit
                  Score = c(49.1, 0.0, 64.7, 67.3, 92.4),
                  Change = c(-1.0, 0.0, NA, 1.5, 0.2))

df4 <- data.frame(Category = c("Exposure", "Water", "Land", "Oceans, rivers and lakes", "Political comm
                  Score = c(76.2, 33.7, 64.7, 26.6, 28.5, 100.0),
                  Change = c(NA, NA, -0.1, NA, -0.3, NA))
```

```r
# Function to create each plot
plot_fun <- function(data, title){
  ggplot(data, aes(x = reorder(Category, -Score), y = Score, fill = Score)) +
    geom_bar(stat="identity") +
    coord_flip() +
    scale_fill_gradientn(colors = colorRampPalette(rev(brewer.pal(11, "Spectral")))(100)) +
    theme_minimal() +
    labs(title = title, x = NULL, y = "Score")
}

# Create individual plots
p1 <- plot_fun(df1, "AFFORDABILITY")
p2 <- plot_fun(df2, "AVAILABILITY")
p3 <- plot_fun(df3, "QUALITY AND SAFETY")
p4 <- plot_fun(df4, "SUSTAINABILITY AND ADAPTATION")

# Arrange the plots into a grid
grid.arrange(p1, p2, p3, p4, nrow = 2, ncol = 2)
```