Project 2

201805070 → Emrah Fidan → Group 35

This is a report I created so that the answer to all questions can appear in a single file. You can look at other folders to review codes and outputs.

Project 2.1

```
# Read File
data <- read.table("DatasetNA.txt", header = TRUE, sep = " ", dec = ",")

# A : Calculate Number of observations, Minimum,Maximum,Range,
# Sum, Mean, Median, Sum of squares, Variance, Standard deviation

# Number of Observations
calculate_num_obs <- function(data) {
  count <- 0
  for (i in 1:length(data)) {
    if (!is.na(data[i])) {
      count <- count + 1
    }
  }
  return(count)
}

# Minimum
calculate_minimum <- function(data) {
  min_val <- Inf
  for (i in 1:length(data)) {
    if (!is.na(data[i]) && (is.infinite(min_val) || data[i] < min_val)) {
      min_val <- data[i]
    }
  }
  return(ifelse(is.infinite(min_val), NA, min_val))
}

# Maximum
calculate_maximum <- function(data) {
  max_val <- -Inf
  for (i in 1:length(data)) {
    if (!is.na(data[i]) && (is.infinite(max_val) || data[i] > max_val)) {
      max_val <- data[i]
    }
  }
  return(ifelse(is.infinite(max_val), NA, max_val))
}

# Range
calculate_range <- function(data) {
  min_val <- calculate_minimum(data)
  max_val <- calculate_maximum(data)
  return(ifelse(is.na(min_val) || is.na(max_val), NA, max_val - min_val))
}

# Sum
calculate_sum <- function(data) {
  sum_val <- 0
  for (i in 1:length(data)) {
```

```
    if (!is.na(data[i])) {
      sum_val <- sum_val + data[i]
    }
  }
  return(sum_val)
}

# Mean
calculate_mean <- function(data) {
  sum_val <- 0
  count <- 0
  for (i in 1:length(data)) {
    if (!is.na(data[i])) {
      sum_val <- sum_val + data[i]
      count <- count + 1
    }
  }
  if (count > 0) {
    mean_val <- sum_val / count
  } else {
    mean_val <- NA
  }
  return(mean_val)
}


# Median
calculate_median <- function(data) {
  sorted_data <- sort(data)
  num_obs <- calculate_num_obs(data)
  if (num_obs %% 2 == 0) {
    median_val <- (sorted_data[num_obs/2] + sorted_data[(num_obs/2) + 1]) / 2
  } else {
    median_val <- sorted_data[(num_obs+1)/2]
  }
  return(ifelse(is.na(median_val), NA, median_val))
}

# Sum of squares
calculate_sum_squares <- function(data) {
  mean_val <- calculate_mean(data)
  sum_squares <- 0
  for (i in 1:length(data)) {
    if (!is.na(data[i])) {
      sum_squares <- sum_squares + (data[i] - mean_val)^2
    }
  }
  return(sum_squares)
}

# Variance
calculate_variance <- function(data) {
  num_obs <- calculate_num_obs(data)
  sum_squares <- calculate_sum_squares(data)
  variance <- sum_squares / (num_obs - 1)
  return(ifelse(is.na(variance), NA, variance))
}

# Standard deviation
calculate_std_dev <- function(data) {
  variance <- calculate_variance(data)
  return(ifelse(is.na(variance), NA, sqrt(variance)))
}

calculate_statistics <- function(data) {
  stats <- c(
    "Number of observations" = calculate_num_obs(data),
    "Minimum" = calculate_minimum(data),
    "Maximum" = calculate_maximum(data),
    "Range" = calculate_range(data),
    "Sum" = calculate_sum(data),
```

```
    "Mean" = calculate_mean(data),
    "Median" = calculate_median(data),
    "Sum of squares" = calculate_sum_squares(data),
    "Variance" = calculate_variance(data),
    "Standard deviation" = calculate_std_dev(data)
  )
  return(stats)
}

# Select Var1-Var8 columns
veri_altkume <- data[, c("Var1", "Var2", "Var3", "Var4", "Var5", "Var6", "Var7", "Var8")]

# String -> numeric
veri_numeric <- apply(veri_altkume, 2, function(x) as.numeric(as.character(gsub(",", ".", x))))


for (i in 1:ncol(veri_numeric)) {
  stats <- calculate_statistics(veri_numeric[, i])
  main_title <- paste("Statistics of", colnames(veri_numeric)[i])
  cat(main_title, "\n")
  cat("Number of observations:", stats[1], "\n")
  cat("Minimum:", stats[2], "\n")
  cat("Maximum:", stats[3], "\n")
  cat("Range:", stats[4], "\n")
  cat("Sum:", stats[5], "\n")
  cat("Mean:", stats[6], "\n")
  cat("Median:", stats[7], "\n")
  cat("Sum of squares:", stats[8], "\n")
  cat("Variance:", stats[9], "\n")
  cat("Standard deviation:", stats[10], "\n\n")
}


#-----------------------------------------------------------------------#

# B : Calculate Cross-products, Covariance, Correlations



calculate_cross_products <- function(data, vars) {
  cross_products <- matrix(NA, nrow = length(vars), ncol = length(vars), dimnames = list(vars, vars))

  for (i in 1:length(vars)) {
    for (j in 1:length(vars)) {
      cross_products[i, j] <- sum(data[[vars[i]]] * data[[vars[j]]], na.rm = TRUE)
    }
  }

  return(cross_products)
}


calculate_covariance <- function(x, y) {
  n <- sum(!is.na(x) & !is.na(y))
  sum_x <- sum(x, na.rm = TRUE)
  sum_y <- sum(y, na.rm = TRUE)

  mean_x <- sum_x / n
  mean_y <- sum_y / n

  covariance <- sum((x - mean_x) * (y - mean_y), na.rm = TRUE) / (n - 1)

  return(covariance)
}

calculate_correlation <- function(x, y) {
  covariance <- calculate_covariance(x, y)
  n <- sum(!is.na(x) & !is.na(y))
  sum_x <- sum(x, na.rm = TRUE)
  sum_y <- sum(y, na.rm = TRUE)
```

```
  mean_x <- sum_x / n
  mean_y <- sum_y / n

  var_x <- sum((x - mean_x)^2, na.rm = TRUE) / (n - 1)
  var_y <- sum((y - mean_y)^2, na.rm = TRUE) / (n - 1)

  correlation <- covariance / (sqrt(var_x) * sqrt(var_y))

  return(correlation)
}


calculate_covariance_correlation <- function(data, vars) {
  cov_matrix <- matrix(NA, nrow = length(vars), ncol = length(vars), dimnames = list(vars, vars))
  corr_matrix <- matrix(NA, nrow = length(vars), ncol = length(vars), dimnames = list(vars, vars))

  for (i in 1:length(vars)) {
    for (j in 1:length(vars)) {
      cov_matrix[i, j] <- calculate_covariance(data[[vars[i]]], data[[vars[j]]])
      corr_matrix[i, j] <- calculate_correlation(data[[vars[i]]], data[[vars[j]]])
    }
  }

  return(list(covariance = cov_matrix, correlation = corr_matrix))
}


# Specify the continuous variables
continuous_vars <- c("Var1", "Var2", "Var3", "Var4", "Var5", "Var6", "Var7", "Var8")

# Calculate cross-products
cross_products <- calculate_cross_products(data, continuous_vars)

# Print the resulting cross-products matrix
print(cross_products)
print("-----------------------------------------------------------------------------------------------------")


# Calculate covariance and correlation
cov_corr <- calculate_covariance_correlation(data, continuous_vars)

# Print the resulting covariance matrix
print(cov_corr$covariance)
print("-----------------------------------------------------------------------------------------------------")

# Print the resulting correlation matrix
print(cov_corr$correlation)
print("-----------------------------------------------------------------------------------------------------")
```

Project 2.2

```
# Read File
data <- read.table("DatasetNA.txt", header = TRUE, sep = " ", dec = ",")

# Function to calculate number of observations
calculate_num_obs <- function(data) {
  count <- 0
  for (i in 1:length(data)) {
    if (!is.na(data[i])) {
      count <- count + 1
    }
  }
  return(count)
}

# Function to calculate minimum
```

```
calculate_minimum <- function(data) {
  min_val <- Inf
  for (i in 1:length(data)) {
    if (!is.na(data[i]) && (is.infinite(min_val) || data[i] < min_val)) {
      min_val <- data[i]
    }
  }
  return(ifelse(is.infinite(min_val), NA, min_val))
}


# Function to calculate maximum
calculate_maximum <- function(data) {
  max_val <- -Inf
  for (i in 1:length(data)) {
    if (!is.na(data[i]) && (is.infinite(max_val) || data[i] > max_val)) {
      max_val <- data[i]
    }
  }
  return(ifelse(is.infinite(max_val), NA, max_val))
}

# Function to calculate range
calculate_range <- function(data) {
  min_val <- calculate_minimum(data)
  max_val <- calculate_maximum(data)
  return(ifelse(is.na(min_val) || is.na(max_val), NA, max_val - min_val))
}

# Function to calculate sum
calculate_sum <- function(data) {
  sum_val <- 0
  for (i in 1:length(data)) {
    if (!is.na(data[i])) {
      sum_val <- sum_val + data[i]
    }
  }
  return(sum_val)
}

# Function to calculate mean
calculate_mean <- function(data) {
  sum_val <- 0
  count <- 0
  for (i in 1:length(data)) {
    if (!is.na(data[i])) {
      sum_val <- sum_val + data[i]
      count <- count + 1
    }
  }
  if (count > 0) {
    mean_val <- sum_val / count
  } else {
    mean_val <- NA
  }
  return(mean_val)
}

# Function to calculate median
calculate_median <- function(data) {
  sorted_data <- sort(data)
  num_obs <- calculate_num_obs(data)
  if (num_obs %% 2 == 0) {
    median_val <- (sorted_data[num_obs/2] + sorted_data[(num_obs/2) + 1]) / 2
  } else {
    median_val <- sorted_data[(num_obs+1)/2]
  }
  return(ifelse(is.na(median_val), NA, median_val))
}

# Function to calculate sum of squares
```

```
calculate_sum_squares <- function(data) {
  mean_val <- calculate_mean(data)
  sum_squares <- 0
  for (i in 1:length(data)) {
    if (!is.na(data[i])) {
      sum_squares <- sum_squares + (data[i] - mean_val)^2
    }
  }
  return(sum_squares)
}

# Function to calculate variance
calculate_variance <- function(data) {
  num_obs <- calculate_num_obs(data)
  sum_squares <- calculate_sum_squares(data)
  variance <- sum_squares / (num_obs - 1)
  return(ifelse(is.na(variance), NA, variance))
}

# Function to calculate standard deviation
calculate_std_dev <- function(data) {
  variance <- calculate_variance(data)
  return(ifelse(is.na(variance), NA, sqrt(variance)))
}

# Function to calculate statistics for a given variable and factor combination
calculate_statistics <- function(data, factor) {
  stats <- list()

  # Number of observations
  stats$num_obs <- calculate_num_obs(data)

  # Minimum
  stats$minimum <- calculate_minimum(data)

  # Maximum
  stats$maximum <- calculate_maximum(data)

  # Range
  stats$range <- calculate_range(data)

  # Sum
  stats$sum <- calculate_sum(data)

  # Mean
  stats$mean <- calculate_mean(data)

  # Median
  stats$median <- calculate_median(data)

  # Sum of squares
  stats$sum_squares <- calculate_sum_squares(data)

  # Variance
  stats$variance <- calculate_variance(data)

  # Standard deviation
  stats$std_dev <- calculate_std_dev(data)

  return(stats)
}

# Function for a given variable by a single factor
calculate_stats_by_factor <- function(data, variable, factor) {
  factor_levels <- unique(data[[factor]])
  stats <- list()

  for (level in factor_levels) {
    subset_data <- data[data[[factor]] == level, variable]
    stats[[as.character(level)]] <- calculate_statistics(subset_data, factor)
  }
```

```
  return(stats)
}

# Function for a given variable by group factor
calculate_stats_by_group <- function(data, variable) {
  stats <- calculate_stats_by_factor(data, variable, "Group")
  return(stats)
}

# Function for a given variable by gender factor
calculate_stats_by_gender <- function(data, variable) {
  stats <- calculate_stats_by_factor(data, variable, "Gender")
  return(stats)
}

# Function for a given variable by group and gender factors
calculate_stats_by_group_gender <- function(data, variable) {
  group_levels <- unique(data[["Group"]])
  gender_levels <- unique(data[["Gender"]])
  stats <- list()

  for (group in group_levels) {
    group_stats <- list()

    for (gender in gender_levels) {
      subset_data <- data[data[["Group"]] == group & data[["Gender"]] == gender, variable]
      group_stats[[as.character(gender)]] <- calculate_statistics(subset_data, "Group")
    }

    stats[[as.character(group)]] <- group_stats
  }

  return(stats)
}

# Select Var1-Var8 columns
veri_altkume <- data[, c("Var1", "Var2", "Var3", "Var4", "Var5", "Var6", "Var7", "Var8")]

# Convert strings to numeric values
veri_numeric <- apply(veri_altkume, 2, function(x) as.numeric(as.character(gsub(",", ".", x))))

# Calculate statistics for Var1 by Group factor [I use Var1]
stats_by_group_Var1 <- calculate_stats_by_group(data, "Var1")
print("Statistics for Var1 by Group factor:")
print(stats_by_group_Var1)

# Calculate statistics for Var1 by Gender factor [I use Var1]
stats_by_gender_Var1 <- calculate_stats_by_gender(data, "Var1")
print("Statistics for Var1 by Gender factor:")
print(stats_by_gender_Var1)

# Calculate statistics for Var1 by Group and Gender factors [I use Var1]
stats_by_group_gender_Var1 <- calculate_stats_by_group_gender(data, "Var1")
print("Statistics for Var1 by Group and Gender factors:")
print(stats_by_group_gender_Var1)


#---------------------------------------------------------------------#



calculate_cross_products <- function(data, factor_vars, continuous_vars) {
  cross_products <- list()

  for (factor_var in factor_vars) {
    factor_levels <- unique(data[[factor_var]])
    factor_cross_products <- list()

    for (level in factor_levels) {
      level_data <- data[data[[factor_var]] == level, ]
```

```
      cross_product_matrix <- matrix(NA, nrow = length(continuous_vars), ncol = length(continuous_vars))
      colnames(cross_product_matrix) <- continuous_vars
      rownames(cross_product_matrix) <- continuous_vars

      for (i in 1:length(continuous_vars)) {
        for (j in 1:length(continuous_vars)) {
          if (i != j) {
            var1 <- continuous_vars[i]
            var2 <- continuous_vars[j]
            cross_product <- sum(level_data[[var1]] * level_data[[var2]], na.rm = TRUE)
            cross_product_matrix[i, j] <- cross_product
          }
        }
      }

      factor_cross_products[[level]] <- cross_product_matrix
    }

    cross_products[[factor_var]] <- factor_cross_products
  }

  return(cross_products)
}



# Specify the factor variables and continuous variables
factor_vars <- c("Gender", "Group")
continuous_vars <- c("Var1", "Var2", "Var3", "Var4", "Var5", "Var6", "Var7", "Var8")

# Calculate the cross-products by gender and group
cross_products_by_factors <- calculate_cross_products(data, factor_vars, continuous_vars)

# Print the cross-products for each factor level
for (factor_var in factor_vars) {
  factor_cross_products <- cross_products_by_factors[[factor_var]]
  print(paste("Cross-Products for", factor_var, ":"))

  for (level in names(factor_cross_products)) {
    print(paste("#------------------------------------------------------------------------------#"))
    print(paste("Level:", level))
    print(paste("#------------------------------------------------------------------------------#"))
    print(factor_cross_products[[level]])
  }
}


calculate_covariance <- function(data, factor_var, continuous_vars) {
  factor_levels <- unique(data[[factor_var]])
  factor_covariances <- list()

  for (level in factor_levels) {
    level_data <- data[data[[factor_var]] == level, ]
    covariance_matrix <- matrix(NA, nrow = length(continuous_vars), ncol = length(continuous_vars))
    colnames(covariance_matrix) <- continuous_vars
    rownames(covariance_matrix) <- continuous_vars

    for (i in 1:length(continuous_vars)) {
      for (j in 1:length(continuous_vars)) {
        var1 <- continuous_vars[i]
        var2 <- continuous_vars[j]
        cross_product <- sum(level_data[[var1]] * level_data[[var2]], na.rm = TRUE)
        mean_var1 <- mean(level_data[[var1]], na.rm = TRUE)
        mean_var2 <- mean(level_data[[var2]], na.rm = TRUE)
        n <- sum(!is.na(level_data[[var1]]) & !is.na(level_data[[var2]]))
        covariance <- (cross_product - (mean_var1 * mean_var2 * n)) / (n - 1)
        covariance_matrix[i, j] <- covariance
      }
    }

    factor_covariances[[level]] <- covariance_matrix
```

```
  }

  return(factor_covariances)
}

calculate_correlation <- function(data, factor_var, continuous_vars) {
  factor_levels <- unique(data[[factor_var]])
  factor_correlations <- list()

  for (level in factor_levels) {
    level_data <- data[data[[factor_var]] == level, ]
    correlation_matrix <- matrix(NA, nrow = length(continuous_vars), ncol = length(continuous_vars))
    colnames(correlation_matrix) <- continuous_vars
    rownames(correlation_matrix) <- continuous_vars

    for (i in 1:length(continuous_vars)) {
      for (j in 1:length(continuous_vars)) {
        var1 <- continuous_vars[i]
        var2 <- continuous_vars[j]
        cross_product <- sum(level_data[[var1]] * level_data[[var2]], na.rm = TRUE)
        sum_sq_var1 <- sum(level_data[[var1]]^2, na.rm = TRUE)
        sum_sq_var2 <- sum(level_data[[var2]]^2, na.rm = TRUE)
        n <- sum(!is.na(level_data[[var1]]) & !is.na(level_data[[var2]]))
        correlation <- cross_product / sqrt(sum_sq_var1 * sum_sq_var2)
        correlation_matrix[i, j] <- correlation
      }
    }

    factor_correlations[[level]] <- correlation_matrix
  }

  return(factor_correlations)
}



# Specify the factor variable and continuous variables
gender_var <- "Gender"
group_var <- "Group"
continuous_vars <- c("Var1", "Var2", "Var3", "Var4", "Var5", "Var6", "Var7", "Var8")

# Calculate covariance for gender levels
gender_covariances <- calculate_covariance(data, gender_var, continuous_vars)

# Print covariances for gender levels
print("Covariance for Gender:")
for (level in names(gender_covariances)) {
  print(paste("Level:", level))
  print(gender_covariances[[level]])
}

# Calculate covariance for group levels
group_covariances <- calculate_covariance(data, group_var, continuous_vars)

# Print covariances for group levels
print("Covariance for Group:")
for (level in names(group_covariances)) {
  print(paste("Level:", level))
  print(group_covariances[[level]])
}

# Calculate correlations for gender levels
gender_correlations <- calculate_correlation(data, gender_var, continuous_vars)

# Print correlations for gender levels
print("Correlation for Gender:")
for (level in names(gender_correlations)) {
  print(paste("Level:", level))
  print(gender_correlations[[level]])
}
```

```
# Calculate correlations for group levels
group_correlations <- calculate_correlation(data, group_var, continuous_vars)

# Print correlations for group levels
print("Correlation for Group:")
for (level in names(group_correlations)) {
  print(paste("Level:", level))
  print(group_correlations[[level]])
}
```

Project 2.3

```
#fonksiyon
draw_scatterplot <- function(data, x_var, y_var) {
  # x ve y değişkenleri belirlenir
  x <- data[[x_var]]
  y <- data[[y_var]]

  #draw
  plot(x, y, xlab = x_var, ylab = y_var, main = "Scatterplot")
}

#matrix
draw_scatterplot_matrix <- function(data) {
  # pairs fonksiyonu kullanılarak scatterplot matrisi çizimi yapılır
  pairs(~., data = data, main = "Scatterplot Matrix")
}

data <- read.table("DatasetNA.txt", header = TRUE, dec = ",")

#string -> numeric
numeric_cols <- c("Var1", "Var2", "Var3", "Var4", "Var5", "Var6", "Var7", "Var8")
data[numeric_cols] <- lapply(data[numeric_cols], function(x) as.numeric(gsub(",", ".", x)))

# between two variable
draw_scatterplot(data, "Var1", "Var2")

# all matrix
draw_scatterplot_matrix(data[, numeric_cols])
```

Project 2.4

```
#create my scale()
myScale <- function(data, vars) {
  if (!is.character(vars)) {
    vars <- as.character(vars)
  }

  for (var in vars) {
    if (var %in% colnames(data)) {
      variable <- data[[var]]
      variable <- variable[!is.na(variable)]

      mean_val <- sum(variable) / length(variable)
      sd_val <- sqrt(sum((variable - mean_val)^2) / length(variable))

      scaled_variable <- round((variable - mean_val) / sd_val, 2)

      # update data
      data[[var]][!is.na(data[[var]])] <- as.vector(scaled_variable)
    } else {
```

```
      print(paste(var, " warning"))
    }
  }

  return(data)
}

data <- read.table("DatasetNA.txt", header = TRUE, sep = " ", dec = ",", na.strings = "NA")
data <- myScale(data, c("Var1","Var2", "Var3", "Var4","Var5", "Var6", "Var7","Var8"))

#print
print(data)
```

> I separated the outputs according to their folders and put them in the Project1 file in pdf format