Data Cleaning

SP

2024-04-01

Data Preparation and Analysis with R

This document outlines the process of loading, transforming, and analyzing a dataset related to medical diagnostics using R.

Setup

###First, we set the working directory and load the necessary libraries.

```
setwd("~/Library/CloudStorage/OneDrive-Personal/My GitHub/R codes/Data Cleaning")
# Set the working directory to your data location

# Load the required libraries
library(readxl) # For reading Excel files
library(tidyverse) # For data manipulation and visualization
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

##Loading Data

###Load the dataset and inspect its column names.

```
df <- read_xlsx("Data.xlsx") # Load data from an Excel file
colnames(df) # Display column names
```

```
##  [1] "CASE NO"                         "NAME"
##  [3] "AGE"                             "SEX"
##  [5] "RESIDENCE"                       "PANA"
##  [7] "DURATION OF CHEST PAIN(HOURS)" "CHEST PAIN (TYPICAL/ATYPICAL)"
##  [9] "OTHER ASSOCIATED SYMPTOMS"     "COMPLICATIONS"
## [11] "DIABETIC"                        "HYPERTENSIVE"
## [13] "PREVIOUS STROKE"                 "ALCOHOLIC"
## [15] "SMOKER"                          "BMI"
## [17] "SYSTOLIC BP"                     "DIASTOLIC BP"
## [19] "PULSE RATE"                      "RAISED JVP"
## [21] "KILLIP CLASS"                    "CKNAC"
## [23] "CKMB"                               "TROP-T"
## [25] "LIPID PROFILE"                      "RBS"
## [27] "HBA1C"                              "Hb LEVELS"
## [29] "ECG FINDIGS"                        "DIAGNOSIS"
## [31] "MANAGEMENT"
```

Print some columns to get an idea of what we are dealing with

```
print(head(df$`OTHER ASSOCIATED SYMPTOMS`))
```

```
## [1] "DIAPHORESIS,GHABRAHAT"           "GHABRAHAT"
## [3] "DYSPNOEA, GHABRAHAT"             "DIAPHORESIS,GHABRAHAT"
## [5] "DIAPHORESIS,GHABRAHAT,DYSPNOEA" "DIAPHORESIS, GHABRAHAT"
```

```
print(head(df$`LIPID PROFILE`))
```

```
## [1] "TC224HDL44LDL98TG112"  "TC244HDL38LDL98TG100"  "TC190HDL44LDL102TG186"
## [4] "TC256HDL50LDL250TG270" "TC184HDL58LDL202TG135" "TC156HDL58LDL202TG155"
```

```
print(head(df$DIAGNOSIS))
```

```
## [1] "CAD/ACS/EXTENSIVEMI"      "CAD/ACS/ANTEROLATERALMI"
## [3] "CAD/ACS/ANTEROLATERALMI" "CAD/ACS/NSTEMI"
## [5] "CAD/ACS/INFERIORWALLMI"  "CAD/ACS/NSTEMI"
```

##Adding ID Column

###Add a unique identifier for each row in the dataset.

```
df$ID <- seq_len(nrow(df)) # Add a sequential ID
#as a new column
```

##Transforming ‘OTHER ASSOCIATED SYMPTOMS’ Column

###Unnest the ‘OTHER ASSOCIATED SYMPTOMS’ column, which contains comma-separated values, into a long format and trim white spaces.

```
df_long <- df %>%
  mutate(`OTHER ASSOCIATED SYMPTOMS` =
           strsplit(as.character(`OTHER ASSOCIATED SYMPTOMS`), ",\\s*")) %>%
  # Split the 'OTHER ASSOCIATED SYMPTOMS' by comma and optional space
  unnest(`OTHER ASSOCIATED SYMPTOMS`) %>%
  mutate(`OTHER ASSOCIATED SYMPTOMS` = trimws(`OTHER ASSOCIATED SYMPTOMS`))
# Trim whitespace from each symptom
```

##Creating One-Hot Encoding for Symptoms

###Convert the ‘OTHER ASSOCIATED SYMPTOMS’ into a one-hot encoding format.

```
df_one_hot <- model.matrix(~ `OTHER ASSOCIATED SYMPTOMS` + 0, data = df_long)
# Create one-hot encoding for symptoms
df_one_hot <- as.data.frame(df_one_hot)
df_long$one_hot <- 1 # Add a column to indicate presence of the symptom
df_one_hot <- merge(df_long, df_one_hot, by = "row.names", all.x = TRUE)
# Merge the one-hot encoding with the long dataframe
```

##Aggregating Symptoms Data

###Aggregate the one-hot encoded symptoms data by the unique ID.

```r
df_final <- df_one_hot %>%
  group_by(ID) %>%
  summarize(across(starts_with("`OTHER ASSOCIATED SYMPTOMS`"),
                   max, na.rm = TRUE)) %>%
  left_join(df, by = "ID")
```

```
## Warning: There was 1 warning in `summarize()`.
## i In argument: `across(starts_with("`OTHER ASSOCIATED SYMPTOMS`"), max, na.rm =
##   TRUE)`.
## i In group 1: `ID = 1`.
## Caused by warning:
## ! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
## Supply arguments directly to `.fns` through an anonymous function instead.
##
##   # Previously
##   across(a:b, mean, na.rm = TRUE)
##
##   # Now
##   across(a:b, \(x) mean(x, na.rm = TRUE))
```

```r
# Aggregate symptoms data and join with original dataframe
df <- select(df_final, -ID) # Remove the ID column
```

##Extracting and Transforming Lipid Profile Data

###Extract numerical values from the 'LIPID PROFILE' column and transform them into separate columns.

```r
df <- df %>%
  mutate(
    TC = as.numeric(str_extract(`LIPID PROFILE`, "(?<=TC)\\d+")),
    # Extract and convert TC value to numeric
    HDL = as.numeric(str_extract(`LIPID PROFILE`, "(?<=HDL)\\d+")),
    # Extract and convert HDL value to numeric
    LDL = as.numeric(str_extract(`LIPID PROFILE`, "(?<=LDL)\\d+")),
    # Extract and convert LDL value to numeric
    TG = as.numeric(str_extract(`LIPID PROFILE`, "(?<=TG)\\d+"))
    # Extract and convert TG value to numeric
  ) %>%
  select(-`LIPID PROFILE`) # Remove the original 'LIPID PROFILE' column
```

##Transforming 'DIAGNOSIS' Column

###Unnest the 'DIAGNOSIS' column into a long format, trim spaces, and convert to a factor.

```r
df$ID <- seq_len(nrow(df)) # Re-add the ID column
df_long <- df %>%
  separate_rows(DIAGNOSIS, sep = "/") %>%
  mutate(DIAGNOSIS = str_trim(DIAGNOSIS))
# Split 'DIAGNOSIS' into separate rows and trim spaces
df_long$DIAGNOSIS <- factor(df_long$DIAGNOSIS) # Convert 'DIAGNOSIS' to factor
```

##Creating One-Hot Encoding for Diagnosis

###Convert the 'DIAGNOSIS' column into a one-hot encoding format.

```r
df_one_hot <- model.matrix(~ DIAGNOSIS + 0, data = df_long)
# Create one-hot encoding for diagnosis
df_one_hot <- as.data.frame(df_one_hot)
```

```
df_one_hot$ID <- df_long$ID
```

##Aggregating Diagnosis Data

###Aggregate the one-hot encoded diagnosis data by the unique ID.

```
df_final <- df_one_hot %>%
  group_by(ID) %>%
  summarise(across(everything(), max, na.rm = TRUE), .groups = 'drop')
# Aggregate diagnosis data
df_merged <- left_join(df, df_final, by = "ID")
# Join aggregated data with original dataframe
df_merged <- select(df_merged, -ID)
# Remove the ID column for the final dataframe
```