Processing

Eduardo Yuki Yada

2022-07-10

Imports

```r
library(tidyverse)
library(readxl)
library(yaml)
library(lubridate)
```

Loading files

```r
df <- read_csv("../dataset/BD Marcapasso_27jun22.csv", show_col_types = FALSE) %>%
  select(-record_id)
df_names <- read_excel("../dataset/Dicionario_dados_BD Marcapasso_27jun22.xlsx")
```

Fixing data dictionary

```r
names(df_names) <- make.names(names(df_names), unique=TRUE) %>% tolower

df_names <- df_names %>%
  mutate(variable.name = case_when(variable.name == 'icu_days_t0' ~ 'icu_t0',
                                   variable.name == 'dialysis_days_t0' ~ 'dialysis_t0',
                                   TRUE ~ variable.name),  # mismatch with dataset column name
         variable.name = str_replace(variable.name, "\\[", ""),
         variable.name = str_replace(variable.name, "\\]", ""),
         field.label = str_replace(field.label, "_", " "), # add spaces
         field.label = str_replace_all(field.label, "[\r\n]" , ""),
         abbrev.field.label = str_replace(field.label, " \\s*\\(([^\\)]+\\))", ""),
         abbrev.field.label = if_else(variable.name %in% c('admission_posop_count',
                                                           'admission_pre_t0_count'),
                                     str_replace(field.label, "de episódios", ""),
                                     abbrev.field.label)) %>%
  rename(momento.aquisicao = momento.da.aquisição.do.dado..admissão.t0.ou.pós.t0.)
```

Separating columns by type

```r
outcome_columns <- df_names %>%
  filter(str_detect(momento.aquisicao, 'Desfecho')) %>%
  .$variable.name

categorical_columns <- df_names %>%
  filter(stringr::str_detect(options..definition, '\\|')) %>%
  .$variable.name %>%
  setdiff(outcome_columns)

date_columns <- df_names %>%
  filter(options..definition == 'data') %>%
```

```
  .$variable.name

location_columns <- c('zipcode', 'patient_city')

other_columns <- c('record_id')

numerical_columns <- setdiff(names(df),
                             c(categorical_columns, date_columns,
                               location_columns, other_columns))

df[categorical_columns] <- lapply(df[categorical_columns],
                                  as.character)
df[outcome_columns] <- lapply(df[outcome_columns],
                              as.numeric)
df[date_columns] <- lapply(df[date_columns],
                           dmy)
```

```
## Warning: 1 failed to parse.
```

```
columns_list <- list('categorical_columns' = categorical_columns,
                     'numerical_columns' = numerical_columns,
                     'date_columns' = date_columns,
                     'location_columns' = location_columns,
                     'outcome_columns' = outcome_columns)

con <- file('./auxiliar/columns_list.yaml', "w")
write_yaml(columns_list, con)
close(con)
```

Recalculating outcome columns for modeling

```
df <- df %>%
  mutate(readmission_1year = readmission_30d + readmission_60d + readmission_180d + readmission_1year,
         readmission_180d = readmission_30d + readmission_60d + readmission_180d,
         readmission_60d = readmission_30d + readmission_60d)
```

Saving processed data

```
saveRDS(df, "../dataset/processed_data.rds")
saveRDS(df_names, "../dataset/processed_dictionary.rds")

save(df, file = "../dataset/processed_data.RData")
save(df_names, file = "../dataset/processed_dictionary.RData")
```