Sustainability Mapper - EUT

SEI

2022-12-21

Sustainability Mapper Tool - EUT

A tool for mapping the projects of the World Bank to the EU Taxonomy

Authors: *Stockholm Environment Institute (SEI)*

Introduction

This RMarkdown file showcases the European Union Taxonomy (EUT) Mapping Tool developed by the Stockholm Environment Institute - Latin America Center - to automate and improve the process of mapping how different WB projects are connected to the EUT Goals. This exercise is based upon the mapping methodology developed by SEI to use for World Bank projects from the FY20 and the FY21 portfolios and the KH coder text-mining protocol for the same years.

The script detailed below is written in the R programming language and employs the Random Forest Machine Learning algorithm, which belongs to the supervised learning techniques. It is based in ensemble learning, combining multiple classifiers to solve complex classification problems. Furthermore, in addition to classifying the WB projects to the EUT, this script also includes the automatic generation of visualizations.

General objective

- Develop a memo explaining the updated mapping methodology, including detailed guidance to allow independent mapping by the World Bank.

Features included within the script

1. Text information extraction from the diverse WB documentation PDFs
2. Random Forest Machine Learning classification model with an 80% of precision
3. Classification of WB projects to the EUT Goals
4. Visualizations

Download R and Rstudio

First, download and install R, then install RStudio

Install R Install RStudio

Running the app

The script should be used with **R 4.2.1** or more recent. Also, it has been tested on **Windows 10 and 11** machine with 8 Gb RAM and 2100 Mhz 6-core processor, so we recommend using a machine with those characteristics or better if you want to analyze many documents. Regarding the OS, we advice to use the same we used to test the script.

For running the script, you should open the *"Sustainability mapper.Rproj"* file, located in the main directory of the app – i.e., the main folder that contains the app's files. When you open the app for the first time, you may need to open the app's script. To do it, you can click on *File* in the RStudio toolbar, and look for the *EUT_Map.R* file, located in the same folder as *"Sustainability mapper.Rproj"*.

After that, you should see a file that begins with the following lines of code:

These lines are there to set up the initial environment conditions needed for running the app. They will import libraries and functions that are used for the analysis.

To run the app, you must run every code expression in order. Each time you press *Ctrl + ENTER*, an expression will be run. Expressions may consist of multiple lines of code, so if you want to check what you run when pressing *Ctrl + ENTER*, you can always check the console.

Importing packages

This program requires to install and import external libraries for its functioning. They are, among others, *cli*, *ggplot2*, *ggraph*, *glue*, *here*, *igraph*, *jsonlite*, *pdftools*, *readr*, *showtext*, *sysfonts*, *tidygraph*, *tidytext*, and *tidyverse*. If you have them installed (for example, if you are using the Sustainability Mapper for the first time), you can skip the following chunk to import them. However, **if you need to install the packages**, you can do it here:

```r
install.packages('cli')
install.packages('ggplot2')
install.packages('ggraph')
install.packages('DBI')
install.packages('RSQLite')
install.packages('glue')
install.packages('here')
install.packages('igraph')
install.packages('jsonlite')
install.packages('pdftools')
install.packages('readr')
install.packages('showtext')
install.packages('sysfonts')
install.packages('tidygraph')
install.packages('tidytext')
install.packages('tidyverse')
install.packages('tm')
install.packages('caTools')
```

In addition to the external packages, the functions of the Sustainability Mapper must be sourced. Also, the fonts used for plotting must be initialized. You can do all of that with the next code chunk.

```r
source(here::here('src', 'Initialiser.R'))

# ===== Initialise the app ====================================================
```

```
# Imports all functions, verifies output folders, and sets up fonts and DPI

initialise_app()
```

Loading the dataset

There are two options for loading your data to the Sustainability Mapper. The first one is to create a new analysis and extract the texts of all the PDF files from a folder. The second one is to retrieve the saved data from a previous analysis. For your first run, you must **start a new analysis from scratch**.

Start a new analysis from scratch

You can process the PDF files contained in a folder for mapping them to the EUT. The following chunk of code can be modified for:

1. Specifying the name of the folder that contains the PDF files.
2. Choosing whether to save the tidy data (i.e., the pre-processed data that the model uses for classifying the documents).
3. The name of the output file.

You can adjust these variables using the first three lines of code in the chunk.

1. `FOLDER` can receive any string (i.e., word/short sentence) between quotation marks as input. It is the name of the folder containing the PDFs and must be located inside the "PDF" folder in the main directory of this app (i.e., the folder containing the R Markdown file you opened).
2. `SAVE_RESULTS` must be either `TRUE` or `FALSE` without quotation marks. If you want to save the results, you must write `TRUE`.
3. `FILENAME` Is the name of the file that will be saved with the pre-processed data. It will be located in the *'Saves/data'* folder in the main directory of the app. You can write any string inside quotation marks.

```
FOLDERNAME <- 'PADs'       # Name of the folder that contains the PDF files
SAVE_RESULTS <- TRUE       # Must be TRUE or FALSE
FILENAME <- 'Test'         # The name of the file that will contain the saved results

# Extracting the texts from all the PDF files in a folder
texts <- extract(FOLDERNAME)

# You can tidy the extracted texts
tidy <- tidify(texts,
               token='sentences',
               low_lim = 0.65,
               up_lim = 0.7,
               export_json = SAVE_RESULTS,
               version_name = FILENAME)
```

Merging the training dataset with the input data

The training dataset is a collection of phrases that are mapped to the EUT Goals. They are imported as `data_training`. On the other hand, the *input data* are the extracted texts from the PDFs, here named as `tidy`.

```r
# ===== Machine learning model =======================================================

# ----- Data set-up for the classification model ----------

# Import the data used for training the model
# Data set-up for the classification model
# Connect to the training database
conn <- dbConnect(RSQLite::SQLite(),
                  here("Settings", "training_data.db"))

# Import the data used for training the model
data_training <- dbGetQuery(conn, glue("SELECT * FROM EUT"))

dbDisconnect(conn)

# Combine the training data set with the extracted texts
data_complete <- rbind(data_training, tidy)

data_complete <- data_complete %>%
    mutate(Text = str_replace_all(data_complete$Text, '[^A-Za-z ]', ''))

# Capturing the data sets' length
nrows_data_t <- nrow(data_training)
nrows_data_c <- nrow(data_complete)

rm(data_training)
```

Create the working data set using a Data Term Matrix (DTM) and the complete data set

```r
# Create a Document Term Matrix (DTM)
dtm <- corpus_dtm(data_complete)

# Create the working data set
working_dataset <- codify(dtm, data_complete)
```

```
## Time difference of 13.97421 secs
```

```r
rm(dtm)
```

Create the model and train it with the training data set

```r
# Set random seed
set.seed(612)

# Select training data
x_train <- working_dataset[1:nrows_data_t,]
x_train$Target <- factor(x_train$Target)

# Select test data
```

```
x_test <- working_dataset[nrows_data_t:nrows_data_c,]
x_test <- x_test %>% slice(-1)
x_test$Target <- factor(x_test$Target)

rm(working_dataset)

ncols_dtm <- dim(x_train)[2]

# Train the random forest model using the training data set
t0 <- Sys.time()
cli_alert_info(paste0("The model is being trained. Please wait, this",
                       "process can take several minutes."))
```

```
## i The model is being trained. Please wait, thisprocess can take several minutes.
```

```
classifier <- randomForest(x = x_train[, -ncols_dtm],
                           y = x_train$Target,
                           ntree = 51)
rm(x_train)
cli_alert_success(paste0(
    "The model was trained successfully after",
    difftime(time1 = t0, time2 = Sys.time(), units = "min")))
```

```
## v The model was trained successfully after-8.97206843296687
```

Classifying the documents

```
# Classify the input data -- Map the projects to the EUT Goals
y_pred <- predict(classifier, newdata = x_test[, -ncols_dtm])
rm(x_test)

# Save the results in a character vector
classified <- character()
for (goal in y_pred) {classified = c(classified, goal)}

# Subset the data frame and paste the classified data
results <- data_complete %>%
    slice(nrows_data_t+1: nrows_data_c)

rm(data_complete)

results$Goal <- classified

results <- results %>% dplyr::filter(Goal != "No map")

results <- as_tibble(results)
```

Data analysis

Creating summary data tables

The machine learning model outputs a table of texts mapped to the EUT Goals. The data can be summarized using the `count_matches()`, `results_matrix()`, and `get_network()`' functions. These functions are used in the following code chunk to get the following tables:

1. **matches:** it is the total matches to the EUT Goals by project.
2. **matrix_relative:** it is a matrix with the mapping results in a wide format. The relative matrix displays the percentages that every EUT Goal share from the total matches in a project.
3. **matrix_absolute:** it is a matrix with the mapping results in a wide format. The absolute matrix displays the total mappings of each EUT Goal across the projects.
4. **net:** it is an *igraph* network dataframe that can be plotted with *ggraph*.

```r
# ====== Summaries

# Total matches by project
matches <- count_matches(results,
                         by = 'Goal',
                         sorted = 'Frequency')


# Results as matrix

matrix_relative <- results_matrix(results,
                                  'Goal',
                                  relative_freqs = TRUE,
                                  with_main_SDG = FALSE)

matrix_absolute <- results_matrix(results,
                                  'Goal',
                                  relative_freqs = TRUE,
                                  with_main_SDG = FALSE)

# Generate network from results
net <- generate_network_EUT(results)
```

Exporting the summary data tables

These summary tables can be exported using the functions in the next chunk. The function `export_summary()` takes two arguments inside its brackets. The first argument is a summary data table (any of the ones created before). The second, is the name of the file exported. This function exports *'.csv'* files to the folder *'Output/data'*, inside the main directory of this app.

```r
export_summary('EUT', matches, 'total_matches_by_project')
```

```
## v Data successfully exported to the path 'Output/EU Taxonomy/data/total_matches_by_project.csv'
```

```r
export_summary('EUT', matrix_relative, 'results_matrix_rel')
```

```
## v Data successfully exported to the path 'Output/EU Taxonomy/data/results_matrix_rel.csv'
```

```
export_summary('EUT', matrix_absolute, 'results_matrix_abs')
```

```
## v Data successfully exported to the path 'Output/EU Taxonomy/data/results_matrix_abs.csv'
```

Data visualization

The Sustainability Mapper Tool - EUT can create two different plots for the analysis. They are:

1. **Matches to the EUT Goals** across the portfolio.
2. **Network graph** of the connections between the goals across the portfolio.

EUT Matches

The EUT Matches shows a bar plot of the EUT Goals according to their number of matches across the World Bank's portfolio

```
plot_EUT_results(results, xlabel='Goal', ylabel='Frequency')
```
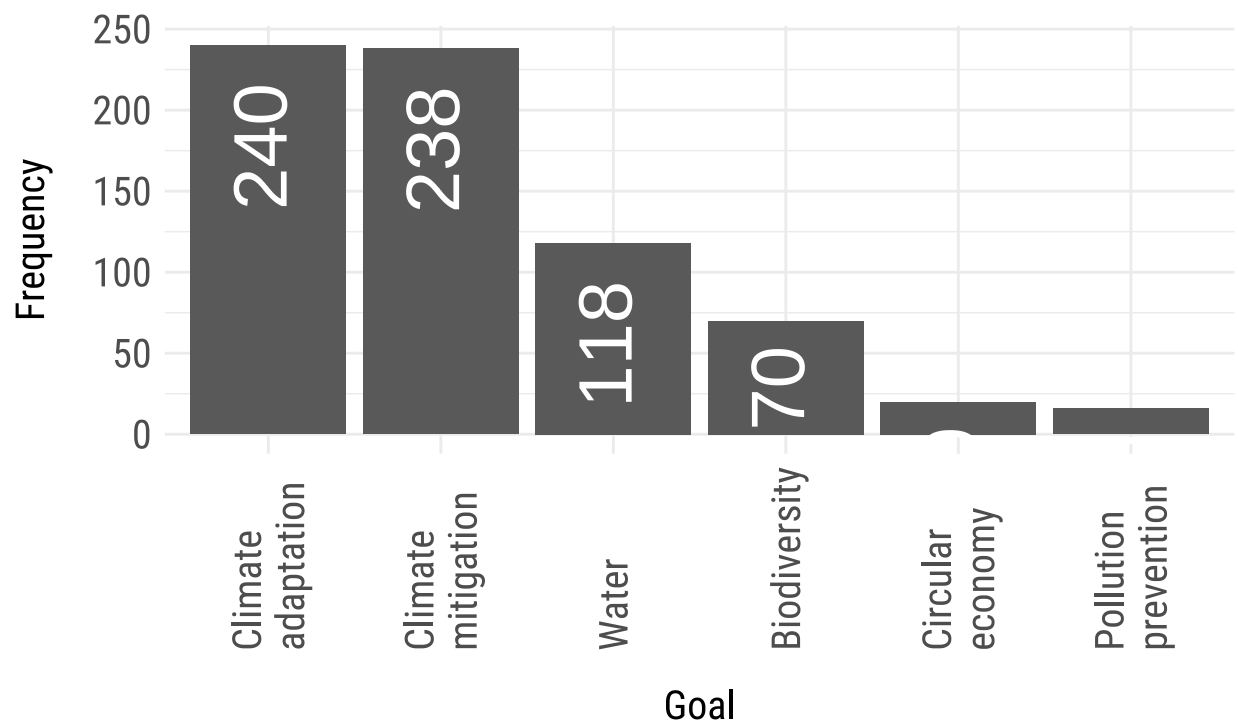

Network graph

This image shows the interconnection between the EUT Goals. We considered that two EUT Goals interact if they coexist in the same project. The network graph aggregates the coexistence of EUT Goals across the World Bank's portfolio.

```
plot_network('EUT',
             results,
             fontsize_base = 15,
             fontsize_title = 20,
             fontsize_subt = 15,
             dpi = 96,
             scale = 1,
             title = "Interactions between the EU Taxonomy Goals",
             subtitle = "In the World Bank's portfolio")
```

Interactions between the EU Taxonomy Goals
In the World Bank's portfolio