PPC_Pipline_Supplementary_Documentation

Johannes Nelson

2023-12-19

Overview

In addition to the documentation within the scripts in the form of comments, this file provides a higher level overview of each script, highlighting key points for functionality, as well as any key assumptions being made.

Extract_Main_Data.R

General

This scripts purpose is to take the data out of Kobo, process it and clean it, and produce organized, easily-legible files for further analysis.

Video Link

Click here for Extraction Overview Video

Inputs/Requirements

Since this is the first stage, there are no local inputs. However, the user will be prompted for the Kobo username and password, so these need to be handy and that user's permissions need to be such that they can extract data in this way.

Outputs

The outputs of this script are files for submission data, tree data, photo data, and geolocation data. They will be stored in a folder called 'Main_Data'.

Assumptions/Notes

There are some assumptions hard-coded into this script. * The script assumes that the name mappings in the 'Kobo_Key' document are accurate. * The script assumes naming conventions within Kobo fields does not change with future form versions (e.g. it searches for geolcation data by looking for the strings 'Corner' and 'Centroid'. If these fields are changed in future form versions, it will not be able to find them. The same goes for key data fields like tree count, species, etc.) Here is a coded example:

```
    names(df)[grep("number", names(df), ignore.case = TRUE)] <- "Tree_Count"
    names(df)[grep("species", names(df), ignore.case = TRUE)] <- "Species"
    names(df)[grep("type", names(df), ignore.case = TRUE)] <- "Tree_Type"
```

This chunk of code conforms column labels by searching for key words within the column labels as they are in Kobo. If these change, this will not find them. Also, if a new column is later added with the word number, species, or type as a part of its name, this will try to label that column with the uniform labels incorrectly.

- Removal of empty columns: There are a number of entirely empty columns in the dataset that mostly correspond to fields that are not inputs but descriptions of parts of the Kobo form. The assumption here is that if there is not a single piece of information from any of the submissions, it is likely one of these technical fields.
- This script labels key information that is used for analysis like monitoring plot sizes (which are summed to get total area monitored) and size class of trees. Both of these labels are created using logic that considers responses in the forms (e.g. If the SiteSize answer is "No" it is a 10x10 plot.) Trees marked as planted at Y0 are given their own size class of '<10cm planted', since they were ending up in many other tables.

Troubleshooting

- **'robotoolbox' R package issue:** At the time of creation, there was a bug in this package, and I needed the creator to go in and fix it, which he was kind enough to do. However, the fix is currently only applied to the development version, so installing this package in the conventional way will not install the version we need. If you get error that looks something like this:

Error in rbindlist(drop_nulls(survey), fill = TRUE): Column 7 of item 3 is length 3 inconsistent with column 1 which is length 5. Only length-1 columns are recycled.

Then you have the older version. Try this:

```
remove.packages("robotoolbox")
install.packages("remotes")
remotes::install_gitlab("dickoa/robotoolbox")
```

- **JSON error:** This happened rarely, but if you encounter an error that talks about JSON format/parsing, wait and try to run the script again. It has to do with a faulty connection/communication step between your machine and Kobo.
- **Table changes within Kobo:** While unlikely at this point in data collection, there is ever-present potential that Kobo will confusingly store things in opaque and mysterious ways that are slightly different than they were at the time I wrote the script. I have included some out puts int he console that should be able to help you see if this is happening, such as the number of expected and retreived tables. Printed statements are also created at each step of the script, so if an error does appear, you can locate which module created the error easily and address the issue.

Extract_Brazil_Data.R

General

This does the same thing as the above script, and everything above pertains to this one as well. I will talk only about the things that are unique to this script here.

Because of the relatively small amount of submissions present in the database at the time of creating the script, I feel less certain that the submissions (including my test submissions) cover all bases of what might get put into the database in the future. Since the main data is full of strange tables, I assume that with time, some of these peculiarities may appear in the Brazil dataset as well. If this happens, the script will need to be altered to accommodate the changes.

Video

Click here for Brazil Extraction Video

Assumptions/Notes

- The Brazil Kobo form was designed differently than the primary form. This was to accommodate collection of different data. There are also differences, however, even in the areas that overlap. The video will go into these differences in more detail and highlight areas of potential concern.

Correct_Species_Names.R

General

This script uses the 'taxize' package to facilitate connection the the Global Names Resolver while attempting to map species names present in the Kobo database with accpeted species names in known scientific databases.

Video Link

Click here for Species Name Correction Video

Inputs/Requirements

- It will take the extracted data from the previous scripts as input–primarily the file with "Tree_Data_Uncorrected" within its name.
- If one exists, which it always should since the repository contains one at the time of hand-over, a 'Taxonomic_Corrections' file in the 'Species_Data' folder will also be used to automatically make corrections that are already known. The script will work without this file, but it will take a very long time to run (20+ minutes just for the automated part–much longer for manual validation).

Outputs

- Tree_Data_Corrected_YYYY-MM-DD_HHMM.csv: tree data with corrected names.
- Taxonomic_Corrections_YYY-MM-DD_HHMM.csv: updated taxonomic corrections.
- Optionally, at the end the user can be prompted to get a file showing the species that need review.

Assumptions

- Extraction scripts have been run
- Two databases are used for name matching: Tropicos - Missouri Botanical Garden and The International Plant Names Index. Others are available if these are not desired.

Troubleshooting

- Runtime: as mentioned, there is a part of the script that takes a while to run, so if you think your R session is frozen, it probably has not.

- Manual validation: This can take a long time, since the script will always show you all the species that it was not able to resolve, and prompt you for the correct name. Hundreds of these are likely common names. If you've corrected all the species names recently and are running the script again, it will still show you all these common names one by one, which can be onerous. I included a 'hidden' option, where if you type the word 'save' as a response, it will move forward and assume you know no corrections for those names.

Add_Family_Names.R

General

This script takes the cleaned and corrected species names and attempts to find family names for them. It does this using the 'rgbif' package to interact with the GBIF database. There is also 'hidden' functionality to use the 'taxize' package to pull from the ncbi database. Results were only slightly different, and since the GBIF method was much faster and required no API token set up (which would have been an obnoxious step that each user would have to do before running the first time), I choose the GBIF route. It there is interest in the other way, I can facilitate that too.

Video Link

Click here for Add Family Names Video

Inputs

- Tree_Data_Corrected_YYYY-MM-DD_HHMM.csv: the output from the previous script with corrected species names.
- Family_Names_YYYY-MM-DD_HHMM.csv: the most up to date file with known family names for each species.

Outputs

- Final_Tree_Data_YYY-MM-DD_HHMM.csv: the final tree data file (before passing to analysis), with correct species names and family names added.
- Family_Names_YYYY-MM-DD_HHMM.csv: An up to date family names file.

Assumptions/Notes

- No major assumptions other than the reliability of the GBIF database.

Troubleshooting

- This script will ask the user if they want to manually add family names for those species which could not be automatically matched. If choosing yes, the user will then have to go through each unresolved species name and look up a family name for it. Only run this if you have time. Unlike species names, I did not have time to create a manually up-to-date family names file, so this will require some effort the first time it is run.

Analyze_Data.R

General

This script takes the final tree data, with corrected species names and added family names, and it uses logic based on table names, size class, and resampling to first scale the tree counts to a uniform 30x30m (900mˆ2) size, and then further scales the counts to the size of the total site (if that size information was available.)

Video Link

Click here for Analyze Data Video

Inputs

- Final_Tree_Data_YYYY-MM-DD_HHMM.csv: the output from the previous script with corrected species names.
- site_size_data.csv: this is a manually saved and located csv that contains site size information.
- Main_Data_YYYY-MM-DD.csv: the main data file created during extraction

Outputs

- Baseline Reports: A folder containing baseline counts of different tree types and different size classes at the site and the species level.
- Indicator Reports: A folder containing files showing planted tree survival rates, trees restored, and trees naturally regenerated–each indicator broken down to the site level and the species level.

Assumptions/Notes

- Trees at Y0 that were marked as planted but that were NOT in the planted tables are ignored.
- Trees at Y0 that were marked as naturally regenerating are included in the counts for 'already present' in both the baseline reports and the trees restored reports (where the Y0 count is subtracted)
- There were still some unresolved questions about how to deal with different tree types during later timeframes (e.g. What about trees marked as 'present' in later time frames? Are the monitors supposed to remember what trees were already there? What about naturally regenerating?) Questions like this will likely need to be addressed as the data comes in.
- Site IDs that are shared within countries and organizations are actually the same sites.

Troubleshooting

The outputs will appear short or blank (depending on which report), in the absence of data from various timeframes. The script has been tested with randomly generated data for Y25 and Y5, and it runs. Until organizations add data from these timeframes, though, the outputs will not be complete.

IMP_Invasive_Species_Scanner.R

General

This script scans the IMP planted tree and seed data for potentially invasive species.

Video Link

Click here for IMP Invasive Scanner Video

Inputs

- ppc_export_xxx. . . csv: The csv export with planting data from the IMP that is manually exported from the IMP and placed in the IMP_Data folder of this repo

Outputs

- Invasive_Species_Data_YYYY-MM-DD_HHMM.csv: a record of all species passed to the script for processing.
- Invasive_Species_Report_YYY-MM-DD_HHMM.csv: The report that contains all flagged species, organizations, countries, and relevant info from the GISD database.

Assumptions/Notes

- ppc_export file exists (this has to be manually downloaded and placed in the IMP_Data folder)
- Common names are not going to return any results. There is a built-in species correction module here as a part of the preprocessing step. There are still common names present in the IMP data, and these will not get flagged even if they are invasive.

Troubleshooting

No known issues.

IMP_Native_Alien_Classification.R

General

This script enables the labeling of species-country pairs as Native, Alien, or Needing further review. If more options are desired, they can be pretty easily coded in. This is a workflow that requires user input, and it can take a long time. If you don't want to do them all at once, you can type 'save' into the response, and this will break the loop and save what you've done. The subsequent run will then check the manually reviewed file and be able to pick up where you left off.

Video Link

Click here for IMP Native Alien Classification Video

Inputs

- IMP_planted_trees_YYYY-MM-DD_HHMM.csv: preprocessed IMP data output by the Invasive Scanner script
- IMP_planted_seeds_YYYY-MM-DD_HHMM.csv: preprocessed IMP data output by the Invasive Scanner script
- GBIF_dataset_keys.csv: file with GRIIS datasets and their relevant keys for lookup in the GBIF database
- Taxonomic_Corrections_YYYY-MM-DD_HHMM.csv: the most recent species corrections.

Outputs

- Checklist_Scan_Results_YYYY-MM-DD_HHMM.csv: results of checks against GRIIS databases. Though supplementary. While these results are mainly displayed during the manual validation process, this output could also be used as a potential introduced species flagged by checking for what species had matches.
- Taxonomic_Corrections_YYYY-MM-DD_HHMM.csv: an updated species correction file
- Manually_Reviewed_Planting_Data_YYYY-MM-DD_HHMM.csv: a dataset containing the reviewed and labelled data after manual validation.

Assumptions/Notes

- GBIF occurrence data is used for mapping–assumption here is about accuracy and validity

Troubleshooting

- Wikipedia is source for relevant info–sometimes the query might grab an irrelevant page