Package ‘fsvm’

January 19, 2022

Title Fine Scale Vegetation Modeling

Version 2.0.0

Description This package contains functions supporting the creation and maintenance of Idaho Department of Fish and Game's Fine Scale Vegetation Modeling project. This includes loading and formatting field data as well as training and predicting from machine learning models. These functions are available as GUI in the IDFG Shiny Application 'Plant Species Locator'.

Depends R (>= 4.0.0)

SystemRequirements Python27, ArcGIS10.6, 32-bit R (for some functions)

License file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Imports reticulate (>= 1.18), caret (>= 6.0.86), data.table (>= 1.14.0), dplyr (>= 1.0.5), foreign (>= 0.8.81), rgdal (>= 1.5.23), stringr (>= 1.4.0), taxize (>= 0.9.99), sp (>= 1.4.5), sf (>= 0.9.8), pROC (>= 1.17.0.1), rpart (>= 4.1.15), ranger (>= 0.12.1), caTools (>= 1.18.2), geosphere (>= 1.5.10), rgeos (>= 0.5.5), DBI (>= 1.1.1), reshape2 (>= 1.4.4), purrr (>= 0.3.4), here (>= 1.0.1), skimr (>= 2.1.3), gridExtra (>= 2.3), grid (>= 4.0.0), rlang (>= 0.4.10), lubridate (>= 1.7.10), doParallel (>= 1.0.16), foreach (>= 1.5.1), ggplot2 (>= 3.3.3), readr (>= 1.4.0), tidyr (>= 1.1.3), terra, future, furrr, RPyGeo, progressr

Suggests nodeHarvest, HDclassif, knitr, rmarkdown, bartMachine, kohonen, gbm, glmnet, Matrix, e1071, xgboost, arm, kernlab, MASS, sda, nnet, naivebayes, ipred, earth, brnn, frbs, elasticnet, monomvn, evtree

VignetteBuilder knitr

NeedsCompilation no

Author Robert Ritson [aut, cre],
Erin Roche [aut],
Wildlife Management Institute [ctr],
Idaho Department of Fish and Game [own, cph]

Maintainer Robert Ritson <robert.ritson@idfg.idaho.gov>

R topics documented:

assignLPI *Assign GPS points along LPI Survey Interval*

Description

Assigns GPS points at each sampled point within a line point intercept survey. Accepts survey units in centimeters, meters, feet, or inches.

Usage

```
assignLPI(
  lpi = dat,
  x = "Longitude",
  y = "Latitude",
  ID = "TranKey",
  interval = "PointInterval",
  n = "PointsCompleted",
  units = "cm",
  azimuth = "Azimuth",
  direction = "Direction",
  datum = c("WGS84", "IDTM")
)
```

Arguments

`lpi`	A data frame containing a raw line point intercept survey
`x`	A character vector corresponding with the X coordiante of a GPS point.
`y`	A character string corresponding with the Y coordiante of a GPS point.
`ID`	A character string corresponding with the unique identifier of a survey.
`interval`	A character string corresponding with the interval used between points within the LPI survey.
`n`	A character string corresponding with the total number of points completed in an LPI survey.
`units`	A character string corresponding with the units of measurement used within the LPI survey. Options include centimeters ("cm"), meters ("m"), feet ("ft"), or inches ("in").
`azimuth`	A character string corresponding with the column containing the azimuth.
`direction`	A character string corresponding with the column containg the direction.
`datum`	A character string corresponding with the datum of the location coordinates."WGS84" and "IDTM" will automatically use the proper proj4strings for those datums. If your coordinates are in a different datum, you must provide the valid proj4string to this parameter.

Value

A data frame containing the assigned coordinates of each surveyed point along an LPI survey.

assign_ecoregions *Assign Bailey's Eco-Region to field data*

Description

Converts field data into a spatial points data frame and assigns eco-region based on intersection with an eco-region shapefile (internal). Uses Idaho Transverse Mercator projection.

Usage

```
assign_ecoregions(
  DAT = field_data,
  lib_loc = "//hqwildstat/C$/Program Files/R/R-4.0.5/library"
)
```

Arguments

DAT	A data frame containing at minimum two columns named "Easting" and "Northing". These are required for coercing the field data into a SpatialPointsDataFrame (package::sp) so a geometric intersection can be performed. Best behavior if coordinates already in Idaho Transverse Mercator projection.
lib_loc	A character string corresponding with the file path to the R library.

Value

Returns a new data frame containing original data provided with the addition of an 'EcoCode' column containing a string corresponding to an eco-region (Bailey's) for each coordinate.

assign_forage *Assign Forage Utility for Game Species*

Description

This function uses the 'ForageSpecies' object within `fsvm` to assign the forage utility of plant species for each target game species. Target game species currently include mule deer, elk, moose, and sage grouse. The forage species currently contains 821 genra, species, subspecies, and varieties of plants documented to be useful sources of forage for the four target game species.

Usage

```
assign_forage(
  fielddata = fielddata,
  target = "AcceptedName",
  lib_loc = "//hqwildstat/C$/Program Files/R/R-4.0.5/library",
  sel = "all"
)
```

Arguments

`fielddata` A data frame containing the plant species to be classified.

`target` A character string corresponding with the column name containing the plant species names. Recommended to use 'AcceptedName' since the forage species in the function will be selected by the accepted name assigned by `rectify_taxa`.

`lib_loc` A character string corresponding with the file path to the R library.

`sel` A character string for filtering the output by forage utility for target game species. Options currently include "all (default)" for no filtering, "forage only" to return only forage species for any target game species, "mule deer", "elk", "moose", and "sage grouse".

Value

A data frame containing the initial data with the appended columns 'MuleDeer','SageGrouse','Elk', and 'Moose' each containing a "Y" or "N" corresponding with their forage utility.

`assign_Lpi` *Assign GPS points along LPI Survey Interval*

Description

Assigns GPS points at each sampled point within a line point intercept survey. Accepts survey units in centimeters, meters, feet, or inches.

Usage

```
assign_Lpi(
  lpi = dat,
  x = "Longitude",
  y = "Latitude",
  ID = "ParentGlobalID",
  interval = "Point Interval (cm)",
  n = "Points Completed",
  units = "cm",
  azimuth = "NA",
  coord.type = c("Start", "End", "NA"),
  datum = c("WGS84", "IDTM")
)
```

Arguments

`lpi` A data frame containing a raw line point intercept survey

`x` A character vector corresponding with the X coordiante of a GPS point.

`y` A character string corresponding with the Y coordiante of a GPS point.

`ID` A character string corresponding with the unique identifier of a survey.

`interval` A character string corresponding with the interval used between points within the LPI survey.

`n` A character string corresponding with the total number of points completed in an LPI survey.

`units` A character string corresponding with the units of measurement used within the LPI survey. Options include centimeters ("cm"), meters ("m"), feet ("ft"), or inches ("in").

`azimuth` A number between 0-360 corresponding with the cardinal direction of the line (if only coordinate point is provided). If multiple points were recorded for the line (Start, Middle, and End), leave use default character string "NA".

`coord.type` A character string corresponding with the type of coordinate provided if only one location is recorded for an LPI survey. "NA" is used if multiple points have already been recorded for an LPI. "Start" uses the provided azimuth number unaltered while "End" calculates the reverse direction (to coincide with the sampled line) to calculate the starting point based on line length.

`datum` A character string corresponding with the datum of the location coordinates."WGS84" and "IDTM" will automatically use the proper proj4strings for those datums. If your coordinates are in a different datum, you must provide the valid proj4string to this parameter.

Value

A data frame containing the assigned coordinates of each surveyed point along an LPI survey.

`assign_Plot` *Assign polygon dimensions to a sample plot*

Description

Creates a polygon for a sampled plot. Two types are currently accepted: Cover plots ("Macro") and Simple Veg Survey location ("SVS"). Macro cover plots are assumed to follow standard dimensions of a 60 meter baseline and 30 meter transects, assumes provided coordinates correspond with the baseline start and must include records for the baseline heading and transect headings; plot is drawn off of the first coordinate with the specified baseline heading and length. NAs for locations or headings are not tolerated in the function and should be removed beforehand. SVS assumes a circular region of a specified radius is sampled, drawing the specified buffer from plot center (assumed to be the provedid coordinates). Also performs unit conversions to meters "m" from centimeter, feet, or inches.

Usage

```
assign_Plot(
  dat = dat,
  x = "Longitude",
  y = "Latitude",
  ID = "GlobalID",
  size = "Radius",
  units = "m",
  type = c("SVS", "Macro"),
  proj = "WGS84",
  base_dir = "BaseHeading",
  tran_dir = "TransectHeading",
  base_len = 60,
  tran_len = 30
)
```

Arguments

`dat` A data frame containing field plot coordinates and information.

`x` A character string corresponding with the x coordinate in the data frame.

`y` A character string corresponding with the y coordiante in the data frame.

`ID` A character string corresponding with the unique identifier of each plot in the data frame.

`size` Only required for type "SVS". A character string corresponding with the plot size of the buffer in the data frame.

`units` A character string corresponding with the measurement units of the plot size or baseline and transect lengths. Default is meters ("m"), but arguments also accepted include centimeters ("cm"), feet ("ft"), and inches ("in").

`type` A character string corresponding with the type of plot (polygon) to be drawn. Currently, only accepted arguments are "SVS" for circular buffers around a provided coordinate or "Macro" for rectangular macro cover plots beginning at the provided coordinate with and valid values for baseline heading (base_dir), transect heading (tran_dir), baseline length (base_len), and transect length (tran_len).

`proj` A character string corresponding with current projection of the provided coordinates. "WGS84" and "IDTM" will automatically provide thier valid proj4strings. If a different projection is needed, a valid proj4string corresponding with with the projection must be provided.

`base_dir` Only required for type "Macro". A character string corresponding with the baseline heading (in degrees) in the dataframe.

`tran_dir` Only required for type "Macro". A character string corresponding with the transect heading (in degrees) in the dataframe.

`base_len` Only required for type "Macro". Default value of 60 meters for a typical macro cover plot. Other numeric values may be used if a different macro protocol has been used.

`tran_len` Only required for type "Macro". Default value of 30 meters for a typical macro cover plot. Other numeric values may be used if a different macro protocol has been used

Value

A 'sf' data frame containing the original data frame with geometery of the created polygons and projected IDTM coordinates appended.

`as_fsvm` *Format field data for machine learning analysis*

Description

Creates a response variable for machine learning models based on whether data was derrived from cover plot methodology or line point intercept methodology. The field data frame should contain a column indicating methodology ("COVER" or "LPI") used to obtain each observation. This column name should be provided to the "DataType" parameter. Observations obtained using the "COVER" methodology should also contain values in a separate column corresponding with percent cover. The column name containing this information should be provdied to the "Prcnt_C" parameter.

Usage

```
as_fsvm(
  fielddata = fielddata_quadpolyID,
  covariates = covariates_quadpolyID,
  group = "G2",
  tax.list = "Artemisia tridentata"
)
```

Arguments

`fielddata`	A data frame containing field data of plant species observations.
`covariates`	A data frame containing covariate data of field observations.
`group`	A character string corresponding with the type of model you want. Options are "G1", "G2", "G3", or "G4" for genus, species, subspecies or variety.
`tax.list`	A character string corresponding with the taxonomic name for which you want to model.

Value

Returns an 'fsvm data' object, a list of five data frames: LpiCov, ObsData_plots, ObsData_pts, PLOTS, and PTS. The function 'fsvm_train()' uses the outputs 'PLOTS' AND 'PTS' for training the machine learning models. 'LpiCov' contains fielddata with a populated 'ResponseVariable' column based on the provided taxonomy inforation and 'DataType'. 'ObsData_plots' and 'ObsData_pts' correspond respectively with 'LpiCov' split by 'COVER' and 'LPI' DataTypes. 'PLOTS' and 'PTS' are the preppared data frames for machine learning analysis with 'PLOTS' containing 'Total', 'Prop', and 'Present' columns while 'PTS' contains 'Total','Hit', 'Present', and 'Prop' columns.

`as_fsvm2` *Format field data for random forest modeling*

Description

Creates response variable for modeling based on survey methodology ("COVER" or "LPI" for 'DataType'). "COVER" type data uses column 'PercentCover' as response variable while "LPI" type data is given '1' (for present).

Usage

```
as_fsvm2(
  fielddata = fielddata_quadpolyID,
  covariates = covariates_quadpolyID,
  group = "G2",
  tax.list = "Artemisia tridentata"
)
```

Arguments

`fielddata` A data frame containing field data of plant species observations.

`covariates` A data frame containing covariate data of field observations.

`group` A character string corresponding with the type of model you want. Options are "G1", "G2", "G3", or "G4" for genus, species, subspecies or variety.

`tax.list` A character string corresponding with the taxonomic name for which you want to model.

Value

Returns an 'fsvm data' object, a list of five data frames: LpiCov, ObsData_plots, ObsData_pts, PLOTS, and PTS. The `fsvm_train_` functions use the outputs 'PLOTS' AND 'PTS' for training the machine learning models. 'LpiCov' contains fielddata with a populated 'ResponseVariable' column based on the provided taxonomy inforation and 'DataType'. 'ObsData_plots' and 'ObsData_pts' correspond respectively with 'LpiCov' split by 'COVER' and 'LPI' DataTypes. 'PLOTS' and 'PTS' are the preppared data frames for machine learning analysis with 'PLOTS' containing 'Total', 'Prop', and 'Present' columns while 'PTS' contains 'Total','Hit', 'Present', and 'Prop' columns.

`finalizeFSVM` *Finalize FSVM SDM predictions*

Description

Add QuadPolyID and Shape information back into outfile of `predictFSVM`

Usage

```
finalizeFSVM(
  outpath = "outpath from predictFSVM",
  shppath = "D:/Fine scale vegetation analysis/dbases_4modeling/24kpolys",
  ncore = 2
)
```

Arguments

`outpath` The file path of saved files from `predictFSVM`

`shppath` The file path to the 24k Quad Covariate RDS files (dbases_4modeling/24kpolys)

`ncores` The number of cores to utilize

Value

Resaves original files from `predictFSVM` with QuadPolyID and Shape info added.

`ForageSpecies` *Forage utility of plant species for target game species.*

Description

A dataset containing the utility of over 3,000 Idaho plant species to four important game species.

Usage

```
ForageSpecies
```

Format

A data frame with 3142 rows and 8 variables

Genus.corrected corrected plant genus name

species.corrected corrected plant species name, specific epithet only

subspecies.variant.corrected corrected plant subspecies or variant name, infraspecific epithet only

SpName full taxonomic name of plant species, Genus specific epithet infraspecific epithet

MuleDeer whether mule deer use the plant taxa as forage, Y or blank

SageGrouse whether sage grouse use the plant taxa as forage, Y or blank

Elk whether elk use the plant taxa as forage, Y or blank

Moose whether moose use the plant taxa as forage, Y or blank

Source

Idaho Department of Fish and Game

`fsvm_dummy` *Dummy data of fine scale vegetation modeling field data.*

Description

A data set containg selections of field data for illustrating fsvm workflow.

Usage

```
fsvm_dummy
```

Format

A data frame containing 126 rows and 10 variables

TranKey Unique transect identifier

PlotKey Unique plot identier along a transect

Source Where the data originally came from

DataType The methodology used to collect the data

Smpl_Yr The year in which the data was collected

Easting The X coordiante of the location

Northing The Y coordinate of the location

Scientific_Name_Complete The full latin binomial or trinomial

Prcnt_C Percent cover of an observation

Count Presence count of an observation

Source

Idaho Department of Fish and Game

fsvm_map *Create a map of species distribution predictions*

Description

Create a map of species distribution predictions

Usage

```
fsvm_map(
  folder = "G2",
  type = "presence",
  file_path = here::here(),
 centroid_filepath = "A:/Fine scale vegetation analysis/fsvm_package/Centroids.RDS",
  file_name = "56_73A_Pseudoroegneria_spicata_NA.csv",
  title
)
```

Arguments

`folder` A character string of the folder group containing the target species model. Options are "G1", "G2", "G3", or "G4".

`type` A character string of the model type. Options are either "presence" or "percent_cover".

`file_path` A character string corresponding with the file path leading to the 'results' folder containing the target models.

`centroid_filepath` A character string corresponding with the file path leading to the 'Centroids.RDS' folder.

`file_name` A character string of the target species model. Either a '.csv' file located in the specified file paths or an R 'fsvm_predict' object.

`title` An optional character string containing the title of the output plot, used if `file_name` is an R object.

Value

A ggplot grid object containing two maps. For "presence" data its probability of presence (not discrete) and presence (discrete) and for "percent_cover" its probability with and without limits

fsvm_nutrition *Calculate nutritional index for a species*

Description

Gathers Nutritional Content database, RAP Biomass prediction, and Presence SDM for the specified plant species and calculates the nutritional index for each eCognition polygon within the extent of the SDM. NI = Prob.pres x Biomass x NutritionContent

Usage

```
fsvm_nutrition(
  sp = "Artemisia tridentata",
  title = "EOE_",
  folder = "G2",
  nu.val = "Mean_CrudeProtein",
  bio.val = "Mean_kgha",
 filepath = "A:/Fine scale vegetation analysis/understory_veg_model/ranger/results/",

   nutrition_dir = "A:/Fine scale vegetation analysis/fsvm_package/NutritionContent_bySpecies.rds"
  biomass_dir = "A:/Fine scale vegetation analysis/fsvm_package/Biomass.RDS"
)
```

Arguments

`sp` A character string corresponding with a species scientific name.

`title` A character string corresponding with the prediction title (typically the predicition extent).

`folder` A character string corresponding with the folder (species name type) containing the prediction SDM results.

`nu.val` A character string corresponding with the nutrtitional content value to use for the nutritional index calculation. Options include "Mean_CrudeProtein","Mean_ODM","Mean_IVDMD Default is "Mean_CrudeProtein".

`bio.val` A character string corresponding with the biomass value to use for the nutritional index calculation. Options include "Mean_kgha","Mean_lbsac","Mean_kg". Default is "Mean_kgha".

`filepath` A character string corresponding with the file path containing SDM prediction RData file.

`nutrition_dir` A character string corresponding with the file path to the NutritionContent.rds file.

`biomass_dir` A character string corresponding with the file path to the folder containing the Biomass rds files.

Value

A data frame containing QuadPolyID, Presence probability, biomass, nutritional content, and nutritional index for the selected plant species.

`fsvm_predict` *Get predictions from trained machine learning models*

Description

Selects best model from each output folder of 'fsvm_train()' and refits it with the entire data set. Function uses output list from 'getPredCovs()'

Usage

```
fsvm_predict(
  pred_covs = predcovs,
  model = mod_file,
  fsvm.dat = NA,
  fsvm.res = NA,
  file_path = "O:/Research",
  folder = "G2",
  type = "presence"
)
```

Arguments

`model`	A character sting corresponding to the file path containing a model output from either 'fsvm_train()' or 'getModels()'. File is a *.RData corresponding to a modeled taxonomic group (ex."Genus_species_subspecies.RData") and contains two objects: "fsvm.dat" and "fsvm.res" which contain the training data and results of model training respectively. If using objects already within R global environment, use the parameters 'fsvm.dat' and 'fsvm.res'.
`fsvm.dat`	The data output object from `as_fsvm`. An optional input to pass the object directly from R to the function instead of loading from a file.
`fsvm.res`	The result output object from `fsvm_train`. An optional input to pass the object directly from R to the function instead of loading from a file.
`file_path`	A character string corresponding with the file path containing the trained models.
`folder`	A character string corresponding with folder containing the model training file. ("G1","G2","G3","G4")
`type`	A character string corresponding with modeling type ("presence" or "percent_cover").
`predcovs`	A list of data frames returned from 'getPredCovs()'

Value

A list containing prediction model ('pred_model'), the prediction ('pred') and a character string of the best model ('best_model').

See Also

'fsvm_train', 'getModels', and 'getPredCovs'

`fsvm_train` *Train fine scale vegetation models with machine learning*

Description

Trains the machine learning models on the provided formatted field data for 29 unique algorithms. Presence data is trained on glm, gbm, glmnet, rf, glmnet_class, xgboost, som, nodeHarvest, bart, bayesian glm, svmPoly, svmRadial, lda, sda, nnet, naive_bayes, and hdda. Percent cover data is trained on lm, glm, gbm, glmnet, rf, xgboost, rpart, rpart2, treebag, bagEarth, som, nodeHarvest, bart, bayesian glm, svmPoly, svmRadial, nnet, brnn, denfis, lasso, blasso, bridge, avnnet. This function relies on the 'caret' package.

Usage

```
fsvm_train(DAT = DAT, type = c("presence", "percent_cover"))
```

Arguments

`DAT`	An 'fsvm data' object produced from `as_fsvm`
`type`	A character string corresponding with model type. "presence" corresponds with presence/absence oriented machine learning models and "percent_cover" corresponds with proportion oriented machine learning models.

Value

An 'fsvm model' object, a list containing five lists: DATA, CONTROL, MODELS, COMPARE, and BEST. DATA contains the training data set, CONTROL contains the definitions of how the models were defined to function, MODELS contains the list of machine learning models fitted, COMPARE contains the model summaries and plots, and BEST contains the best fitted model

---

`fsvm_train_lite` *Train fine scale vegetation models with machine learning - LITE*

Description

Trains the machine learning models on the provided formatted field data for 25 unique algorithms*. Presence data is trained on glm, gbm, glmnet, rf, glmnet_class, xgboost, som, bayesian glm, svmPoly, svmRadial, lda, sda, and naivebayes. Percent cover data is trained on lm, glm, gbm, glmnet, rf, xgboost, rpart, rpart2, treebag, bagEarth, som, bayesian glm, svmPoly, svmRadial, nnet, brnn, denfis, lasso, blasso, bridge, and evtree. Algorithms are implemented using the 'caret' package. *Same as `fsvm_train` but without 'BART', 'nodeHarvest', 'hdda', and 'avnnet'

Usage

```
fsvm_train_lite(DAT = DAT, type = c("presence", "percent_cover"))
```

Arguments

`DAT`	An 'fsvm data' object produced from `as_fsvm`
`type`	A character string corresponding with model type. "presence" corresponds with presence/absence oriented machine learning models and "percent_cover" corresponds with proportion oriented machine learning models.

Value

An 'fsvm model' object, a list containing five lists: DATA, CONTROL, MODELS, COMPARE, and BEST. DATA contains the training data set, CONTROL contains the definitions of how the models were defined to function, MODELS contains the list of machine learning models fitted, COMPARE contains the model summaries and plots, and BEST contains the best fitted model

`fsvm_train_ranger` *Train random forest models for fsvm*

Description

Trains random forest models using 'ranger' and 'caret' with a custom subsampling function for imbalanced data (see Evans and Cushman 2009)

Usage

```
fsvm_train_ranger(DAT = DAT, type = "presence")
```

Arguments

`DAT`	An 'fsvm data' object produced from `as_fsvm`
`type`	A character string corresponding with model type. Options are "presence" for presence/absence data and "percent_cover" for proportion data.

Value

An 'fsvm model' object, containing 4 lists: DATA, CONTROL, MODEL, and SUMMARY. DATA contains the training data set, CONTROL contains the definitions of how the models were defined to function, MODEL contains the details of the trained model, and SUMMARY contains the model summary statistics.

fsvm_train_ranger2 *Train random forest models with 24k Quad covariates*

Description

Trains random forest models using ’ranger’ and ’caret’ with a custom subsampling function for imbalanced data (see Evans and Cushman 2009)

Usage

```
fsvm_train_ranger2(DAT = DAT, type = "presence")
```

Arguments

`DAT`	An ’fsvm data’ object produced from `as_fsvm`
`type`	A character string corresponding with model type. Options are "presence" for presence/absence data and "percent_cover" for proportion data.

Value

An ’fsvm model’ object, containing 4 lists: DATA, CONTROL, MODEL, and SUMMARY. DATA contains the training data set, CONTROL contains the definitions of how the models were defined to function, MODEL contains the details of the trained model, and SUMMARY contains the model summary statistics.

getBiomass *Get Biomass Predictions*

Description

Extracts biomass predictions from RAP 2021 raster

Usage

```
getBiomass(
  quads = unique(preds$pred[["QuadPolyID"]]),
  bio.file.path = "A:/Fine scale vegetation analysis/fsvm_package/Biomass.RDS"
)
```

Arguments

`quads`	A list of QuadPolyIDs to query from the Biomass RDS files.
`bio.file.path`	A character string corresponding with the file path to the Biomass RDS files.

Value

A data frame containing biomass predictions for each QuadPolyID in the ’quads’ query list.

getCentroidMap *Get centroid maps of model predictions by spatial extent*

Description

Creates maps of model predictions by defined spatial extent

Usage

```
getCentroidMap(
  folder = "G2",
  type = "presence",
  files = NULL,
  file_path = here::here(),
 centroid_filepath = "A:/Fine scale vegetation analysis/fsvm_package/Covariates.RDS"
)
```

Arguments

folder	Leave default 'c("genus","species","subspecies")' in order to loop through all the model groups. If a different loop length is desired, then can define any of the options as single character string.
type	Leave default 'c("presence","perecent_cover") in order to loop through all model types. If a different loop length is desired, then can define either as a single character string.
files	Leave default of 'NULL' in order to loop through all available files for a model group folder. Define a vector of character strings here if a different loop length is desired.
pred_covs	A character string corresponding with the file path for an output from 'prep_pred_covs()'.
title	A character sting corresponding with the name of the spatial extent used to create 'pred_covs' parameter.

Value

Maps are saved as *.tif files to the folder path here(file.path("results",folder,type,"maps",paste(title,".tif",sep=""))).

getCovariates *Extract eCognition QuadPolygon Covariates*

Description

This function uses the QuadPoly_IDs extracted for field data to query eCognition quadpolygon covariate data tables and return a formatted data frame of covariates ready for machine learning modeling

Usage

```
getCovariates(
  dat = FieldDataPoints_QuadPolyID,
 cov.file.path = "A:/Fine scale vegetation analysis/fsvm_package/Covariates.RDS",
  rm.na = T,
  export = F
)
```

Arguments

`dat`	A data frame containing the input data set (an output of 'py_extract_quadpolyID')
`cov.file.path`	File path to covariate files on network drive. Default uses 'A:' for HQWILD-STAT.
`rm.na`	A boolean statement indicating whether or not to remove NAs from the output
`export`	A boolean statement indicating whether or not to export the output as an '.RData' file.Outputs are saved to the file path 'here::here()' with the file name "Output-Covariates_QuadPolyID.RData".

Value

A data frame containing formatted eCognition polygon covariate data ready for machine learning modeling (see 'as.fsvm'). The output is also saved as an *.RData file in the current working directory.

getCovariates_24k *Load Covariates from 24k Quads (eCognition polygons)*

Description

This function loads covariate data corresponding with field data of observations (by 'QuadPolyID') to train species distribution models.

Usage

```
getCovariates_24k(
  dat = fielddata,
  cov.file.path = "A:/Fine scale vegetation analysis/dbases_4modeling/24kpolys",
  lvls.path = "O:/Research/factor_levels.rds",
  ncore = 1,
  export = F
)
```

Arguments

`dat`	A data frame of field data (output of `getSQLData`). File must contain 'QuadPolyID' and 'Quad100k' columns indicating which 24k quad tile it is located in
`cov.file.path`	File path to covariate files on network drive. Default uses 'A:' for HQWILD-STAT

`lvls.path`	File path to factor levels of covariates (for predictions)
`ncore`	Number of cores to process with. Default of 1 triggers single thread execution
`export`	A boolean statement indicating whether or not to export the output as an '.RData' file.Outputs are saved to the file path here::here()' with the file name "Output-Covariates_QuadPolyID.RData".

Value

A data frame containing formatted eCognition polygon covariate data ready for machine learning modeling (see 'as.fsvm')

`getGMUExtent` *Get 24k Quad Extent for GMU(s)*

Description

Get file list of all 24k Quads which intersect selected GMU(s)

Usage

```
getGMUExtent(
  gmu_sel = c("6", "10A"),
  lib_loc = "//hqwildstat/C$/Program Files/R/R-4.0.5/library"
)
```

Arguments

`gmu_sel`	A character vector of GMU names
`lib_loc`	Location in which `fsvm` package is installed (accessing internal files)

Value

A list of 24k quad files (paste0('q',UID,'.rds')) intersecting the selected GMU(s)

`getModels` *Get SDM models from field data*

Description

Formats field data and trains machine learning species distribution models. Function performs data formatting, trains models, and exports the results.

Usage

```
getModels(
  fielddata = fielddata,
  file_path = "O:/Research",
  iterator = "Forage",
  group = "G2",
  type = "presence",
  train = "ranger",
  parallel = TRUE,
  covs = covariates,
 cov.file.path = "D:/Fine scale vegetation analysis/fsvm_package/Covariates.RDS"
)
```

Arguments

`fielddata` A data frame containing formatted field data (with quadpoly IDs and rectified taxa)

`file_path` A character sting corresponding with the file path to store the trained models.

`iterator` A character string corresponding with the subset of taxa you would like to model. Options include "ALL" for no subset (recommended when group parameter is "G1"), "Forage" for only species in 'ForageSpecies' labeled yes, or an object containing a custom species list.

`group` A character string corresponding with modeling group you want to evaluate. Must be one of "G1","G2","G3", or "G4". This selection also prompts which forage model data frame to use (a system file generated from 'ForageSpecies' data). This parameter also populates the subfolder name of the output.

`type` A character string corresponding with the type of model you want to execute. Must be one of "presence" or "percent_cover". This also generates a subfolder name to store the output (within subfolder created by 'model' parameter).

`train` A character string indicating which training function to use, "full" for 'fsvm_train', "lite" for 'fsvm_train_lite', or "ranger" for 'fsvm_train_ranger'.

`parallel` A boolean statement whether or not to execute function in parallel (0.5 * n. cores)

`covs` A data frame containing the covariates corresponding to the field data. If NA, `getCovariates` will be executed internally on 'fielddata'.

`cov.file.path` File path to covariate files on network drive. Default uses 'A:' for HQWILDSTAT.

Value

Model outputs are saved to the folder path defined as here(file.path("models","forage.model","type","file.RDATA")) where "forage.model" corresponds with the provided 'model' parameter, "type" corresponds with the provided 'type' parameter and "file.RData" is generated from the taxonomic group modeled (ex."Genus_species_subspecies.RData").

getModels_ranger *Get Random Forest SDMs for field data*

Description

Formats field data and trains random forest species distribution models, exporting trained models.

Usage

```
getModels_ranger(
  fielddata = fielddata,
  file_path = "O:/Research",
  iterator = c("Species1", "Species2", "Species3"),
  group = "G2",
  type = "presence",
  ncores = 1,
  covs = covariates,
  cov.file.path = "D:/Fine scale vegetation analysis/dbases_4modeling/24kpolys"
)
```

Arguments

`fielddata` A data frame containing cleaned, formatted field data (with QuadPolyID)

`file_path` A character sting corresponding with the file path to store the trained models.

`iterator` A character string corresponding with the subset of taxa you would like to model. If less than 10 items in list, single thread execution automatically triggered.

`group` A character string corresponding with modeling group you want to evaluate. Must be one of "G1","G2","G3", or "G4". This parameter also populates the subfolder name of the output.

`type` A character string corresponding with the type of model you want to execute. Must be one of "presence" or "percent_cover". This also generates a subfolder within the `group` subfolder. 'model' parameter).

`ncores` The number of cores to process with. Default = 1 triggers single thread execution.

`covs` A data frame containing the covariates corresponding to the field data. If NA, `getCovariates_24k` will be executed internally on 'fielddata'.

`cov.file.path` File path to covariate files on network drive. Default uses 'A:' for HQWILDSTAT.

Value

Two outputs are saved with this function. The first to the folder path defined as file.path(file_path,"models",group,type,"fil where "file.rds" is generated from the taxonomic group modeled (ex."Genus species.rds") and contains the model and summary/supplemental information from training. The second file is saved to file.path(file_path,"models","justModels",group,type,"file.rds") which contains only the model object for predictions.

getNutrition *Get Composite Nutrition Index*

Description

Gathers Nutritional Content database, RAP Biomass prediction, and Presence SDM for the specified plant species and calculates the nutritional index for each eCognition polygon within the extent of the SDM. NI = Prob.pres x Biomass x NutritionContent

Usage

```
getNutrition(
  forage_list = forage_list,
  qpid = temp$QuadPolyID,
  title = "EOE_",
  folder = "G2",
  nu.val = "Mean_CrudeProtein",
  bio.val = "Mean_kgha",
 filepath = "A:/Fine scale vegetation analysis/understory_veg_model/ranger/results/",

   nutrition_dir = "A:/Fine scale vegetation analysis/fsvm_package/NutritionContent_bySpecies.rds"
  biomass_dir = "A:/Fine scale vegetation analysis/fsvm_package/Biomass.RDS"
)
```

Arguments

`forage_list` A list of plant species scientific names.

`qpid` A list of QuadPolyIDs (prediction extent).

`title` A character string corresponding with the prediction title (typically the predicition extent).

`folder` A character string corresponding with the folder (species name type) containing the prediction SDM results.

`nu.val` A character string corresponding with the nutrtitional content value to use for the nutritional index calculation. Options include "Mean_CrudeProtein","Mean_ODM","Mean_IVDMD Default is "Mean_CrudeProtein".

`bio.val` A character string corresponding with the biomass value to use for the nutritional index calculation. Options include "Mean_kgha","Mean_lbsac","Mean_kg". Default is "Mean_kgha".

`filepath` A character string corresponding with the file path containing SDM prediction RData file.

`nutrition_dir` A character string corresponding with the file path to the NutritionContent.rds file.

`biomass_dir` A character string corresponding with the file path to the folder containing the Biomass rds files.

Value

A list of two objects, species_indices and composite_nutrition_index which contain

getPredCovs *Prepare Covariates for Model Predictions*

Description

Formats covariates extracted for the entire spatial extent of the field data for use in making predictions from the trained machine learning models

Usage

```
getPredCovs(
  extent = NA,
  gmu.sel = NA,
 cov.file.path = "A:/Fine scale vegetation analysis/fsvm_package/Covariates.RDS",
  quad.sel = NA,
  export = F
)
```

Arguments

`extent`	A data frame containing the output of 'py_extract_quadpolyID' for an extent shapefile. Use `NA` if providing GMU(s).
`gmu.sel`	A list of valid IDFG Game Management Units (GMUs) to query in GMU look-up table. All covariates within the provided GMUs will be returned. Use `NA` if providing an extent.
`cov.file.path`	File path to covariate files on network drive. Default uses 'A:' for HQWILDSTAT.
`quad.sel`	A vector output of `set_extent_manual` containing the names of the 100k quad polygons needed. This optional parameter (`NA` default) can help to speed up the function by restricting the number of 100k quads that need to be sorted through.
`export`	A boolean statement indicating whether or not to export the output as an '.RData' file.Outputs are saved to the file path 'here::here()' with the file name "Prepare_Covariates_Prediction.RData".

Value

A list of three data frames: 'pred_data', 'pred_data_ids', and 'miss_rows'. 'pred_data' contains the covariate data to be used in model predictions 'pred_data_ids' contains the QuadPolyIDs corresponding with prediction covariates. 'miss_rows' contains all observations of 'NA'.

getPredCovs2 *Prepare 24k Quad Covariates for Model Predictions*

Description

Format covariates for all 24k quads within specified extent to be predicted with trained SDMs.

Usage

```
getPredCovs2(
  cov.file.path = "A:/Fine scale vegetation analysis/dbases_4modeling/24kpolys",
  lvls.path = "O:/Research/factor_levels.rds",
  ncore = 1,
  outpath = "A:/Fine scale vegetation analysis/dbases_4modeling/24kpolys_predcovs"
)
```

Arguments

`cov.file.path` File path to covariate files on network drive. Default uses 'A:' for HQWILDSTAT.

`lvls.path` File path to factor levels of covariates

`ncore` Number of cores to process with. Default of 1 triggers conventional execution.

`outpath` File path where to save formatted prediction covariates.

Value

Saves outputs as RDS files in specified outpath

`getPredictions` *Get predictions from all models*

Description

Uses parallel processing framework to loop through all model outputs (from 'fsvm_train') with 'fsvm_predict'. Iterates predictions by model type ("presence" or "percent_cover"), folders (modeling group; "genus", "species" or "subspecies"), and model files (ex. "Genus_species_subspecies.RData"). If this too slow or ineffective, may need to examine additional batch processing or command-line options.

Usage

```
getPredictions(
  pred_covs = pred_covs,
  title = "Name of a Spatial Extent",
  file_path = "O:/Research",
  type = "presence",
  folder = "G2",
  ranger = T,
  parallel = T,
  mod_files = NULL
)
```

Arguments

`pred_covs` An object which is output from `getPredCovs()`.

`title` A character string corresponding with the name of the spatial extent used for `getPredCovs()`.

`file_path` A character string defining file path location of the folder containing the trained models and destination of the prediction results.

`type` A character vector corresponding with the model type subfolder name (either ’presence’ or ’percent_cover).

`folder` A character string corresponding with the group type subfolder name (either "G1","G2","G3", or "G4").

`ranger` A boolean statement, whether or not the file path leads to a model output of ’fsvm_train_ranger’.

`parallel` A boolean statement, whether or not to execute the function in parallel (0.5 * n. cores)

`mod_files` Leave default of ’NULL’ in order to loop through all available files for a model group folder. Define a vector of character strings here if a different loop length is desired.

Value

Model predictions are saved to the folder path file.path(file_path,"results",folder,type).

`getSQLData` *Get up-to-date field data from IFWIS SQL Server*

Description

Get up-to-date field data from IFWIS SQL Server

Usage

```
getSQLData(sqltable = "Veg_fsvm_understory_model_data")
```

Arguments

`sqltable` A character string naming the SQL Table within the server containing the updated field data.

Value

An object containing up-to-date field data

getSummary *Get summary of FSVM predictions*

Description

Iterates through output files from 'GetPredictions' to report model summaries. Parameters can be adjusted to restrict loop lengths.

Usage

```
getSummary(
  file_path = "O:/Research",
  folder = "G2",
  type = "presence",
  mod_files = NULL
)
```

Arguments

file_path	A character string defining the file path to the 'model' folder containing the trained models.
folder	A character string corresponding with a model subgroup (either "G1","G2","G3","G4").
type	A character string corresponding to a model type (either "presence" or "percent_cover").
mod_files	A character string or vector of character strings corresponding with full file names of outputs from 'getModels'. Default (NULL) triggers loop for all files within destination file path.

Value

A data frame containing the summary of best models for each specis, saved to the 'results' folder of the provided file path.

See Also

'getPredictions'

getSurvey123 *Download Survey123 Data from ArcGIS Online*

Description

Accesses ArcGIS Online (AGOL) for a permitted IDFG account user with a python script (downloadSVS.py) which downloads data from a specified survey. Requires 32-bit R unless a valid alternative py.path is specified (may be able to use default available Python with 64-bit). A valid AGOL username and password, the featureServiceID, the url of the survey, and download token must all be specified. Also requires a file path to a download folder and filename. Output format is recommended to remain "CSV", however "Shapefile" and "File Geodatabase" are valid alternatives. Only an output parameter of "CSV" will trigger the function to format the output, other formats will be returned as is.

Usage

```
getSurvey123(
  py.path = "C:/Python27/ArcGIS10.6/python.exe",
  featureService_ID,
  output_format = "CSV",
  download_folder = "C:/Temp/",
  agol.username,
  agol.password
)
```

Arguments

`py.path` Character string of file path to a valid installation of Python. If using default path, must be running R in 32-bit session with a valid installation of ArcGIS.

`featureService_ID` A character string containing a valid feature service id for the desired survey. See vignette for acquiring the feature service id for a desired survey.

`output_format` A character string corresponding with the desired output format. Must be either "CSV", "Excel", "KML", "Shapefile", or "File Geodatabase".

`download_folder` A character string corresponding with the file path of the folder you want the downloads to be saved to.

`agol.username` A character string containing a valid ArcGIS Online IDFG account (firstname.lastname_IDFGgis) with AGOL license. **MUST HAVE DEVELOPER PERMISSIONS ACTIVATED**

`agol.password` A character string containing a valid password corresponding to specified AGOL username.

`ifwis_resolve` *Get Idaho Fish and Wildlife Information System (IFWIS) Taxonomic ID*

Description

Gets IFWIS TaxonID number from scientific names

Usage

```
ifwis_resolve(dat = fielddata)
```

Arguments

`dat` A data frame containing scientific names of plants (or fungi)

`sci.name` A character string corresponding with the column which contains the latin binomial of plant species.

`kingdom` A character sting corresponding with the taxonomic kingdom of interest. Usually should use "Plantae" for only vascular plants but can also include "Fungi" for including fungus as well (ex. c("Plantae","Fungi")).

Value

A data frame with an appended 'TaxonID' column containg the IFWIS TaxonID for the species.

`instaload` *Install and load R packages with one function*

Description

Loads a list of packages into workspace, installing those not detected on machine first.

Usage

```
instaload(rPack, update = F)
```

Arguments

`rPack` a list of valid R packages (CRAN only; `remotes` not yet supported)

`update` a boolean statement, default of FALSE. If true, currently loaded packages are detached then all are re-installed and loaded into the session

`lpi2fsvm` *Convert an IDFG LPI Survey to 'fsvm' format*

Description

Convert an IDFG LPI Survey to 'fsvm' format

Usage

```
lpi2fsvm(

    lpi_path = "A:/Fine scale vegetation analysis/understory_veg_model/data/FieldData/DataFormattin
   lib_loc = "C:/Users/rritson/Documents/R/win-library/4.0",
   export = TRUE
)
```

Arguments

`lpi_path` The file path to the folder containing the component csv files of an IDFG LPI survey download (frmLPI.csv, tblLPI.csv, and LowerLevels.csv).

`lib_loc` The file path to the R library on your machine.

`export` A boolean statement indicating whether to save the output file as a csv.

Value

A formatted data frame in proper 'fsvm' format with correct column headings including rectified taxa, modeling groups, ecocodes, and forage

mapFSVM *Visualize FSVM SDM predictions*

Description

Rasterize SDM predictions for each 24k quad in extent, then mosaics tiles into one *.tif raster

Usage

```
mapFSVM(
  res,
  species,
  filepath = "K:/Wildlife/Wildlife Research/Fine_scale_vegetation_2017/Ritson",
  ncore = 1,
  outname = "GMU6_10A_pred.tif"
)
```

Arguments

`res`	Results file path (from `predictFSVM`), only 24k quads corresponding with desired extent (see `getGMUExtent`)
`species`	A list of species for which to visualize predictions; corresponds with column names of RDS files from `predictFSVM`
`filepath`	File path to store temporary files and save final mosaic raster; each species will be saved into a named folder in this path.
`ncore`	Number of cores to process with
`outname`	Name of final mosaic prediction raster, must include '.tif' at the end; file will appear in a new folder in 'filepath' with the species name of the prediction

Value

An SDM prediction mosaic raster for each species 's' in the extent 'res'

predictFSVM *Get predictions from fine-scale vegetation SDMs*

Description

Get predictions from FSVM species distribution models for all specified species (`rangerModel`) in the specified extent (`quads`).

Usage

```
predictFSVM(
  quads,
  rangerModel,
  N_Sample_max = NULL,
  Type_Sample_pred = "Cook",
  ncore = 1,

   outpath = "A:/Fine scale vegetation analysis/understory_veg_model/ranger2/results/RDS"
)
```

Arguments

`quads` List of all 24k quad files (from `getPredCovs2`) within spatial extent of where you want predictions

`rangerModel` List of all species with random forest model files from `getModels_ranger` for which you want SDM predictions

`N_Sample_max` Specify parameter of number of samples to use (from field data, maximized with default NULL, which triggers max sample number detected from field data)

`Type_Sample_pred`
Specify parameter of sample type to use (from field data, maximized by default 'Cook')

`ncore` Number of processors to use for function evaluation.

`outpath` File path where to save output files

Value

Saves RDS files for each 24k Quad which contain predictions for all specified species.

`py_extract_quadpolyID` *Extract eCognition quadpolygon IDs for field data*

Description

Leverages the power of Python and ArcGIS using interpreter 'reticulate' to call a Python script. Creates a data frame containing merged and formatted field data associated with eCognition QuadPoly_IDs.

Usage

```
py_extract_quadpolyID(
  py.path = "//hqwildstat/C$/Python27/ArcGIS10.6/python.exe",

   fielddata.path = "//hqwildstat/D$/Fine scale vegetation analysis/understory_veg_model/PlantSpec

   quad.path = "//hqwildstat/D$/Fine scale vegetation analysis/dbases_4modeling/blank_polys100k.gd

   output.gdb.path = "//hqwildstat/D$/Fine scale vegetation analysis/understory_veg_model/PlantSpe
```

```
     output.folder.path = "//hqwildstat/D$/Fine scale vegetation analysis/understory_veg_model/Plant
    output.RData = "None",
    intercept.feature = "None",
    newgdb.name = "Output.gdb",
    quad.sel = quad.sel,
    export = T,
    survey = T
  )
```

Arguments

`py.path` Character string of file path to ArcGIS installed version of Python (required for running 'arcpy' module). Default is the file path where this is typically stored by ESRI. Function will not run without a valid version of ArcGIS installed before hand and must be run in a 32-bit R session.

`fielddata.path` Character string of file path to geodatabase or shapefile containing coordinates of field data or a shapefile of the target spatial extent in the case of prediction covariates (Must be transformed to IDTM before running function). If a geodatabase, then must include an argument for the parameter 'intercept.feature'.

`quad.path` A character string corresponding with the file path to the folder containing the 100k USGS quads.Typically is "A://Fine scale vegetation analysis/dbases_4modeling/blank_polys10(but needs to be changed for use in SHINY.

`output.gdb.path`

Character string of file path to a valid geodatabase or folder. If path is to a folder, then must provide an argument for newgdb.name. This will prompt the python script to create a new *.gdb for storing output files.

`output.folder.path`

Character string of file path to a valid folder for containing output *.dbf tables. This function now includes the ability now to create the specified folder along the file path if it does not already exist.

`output.RData` Character string containing the name of an *.RData file for storing the formatted and merged data table. String must contain '.RData'. The default ("None") autogenerates a name based on the choice of quad.path geodatabase, either "FieldDataPoints_QuadPolyID.RData" if "blank_polys100k.gdb" or "Covariates_QuadPolyID.RData" if "poly_merges.gdb".

`intercept.feature`

Character string containing the name of a file within a fielddata.path geodatabase. Default ("None") assumes the parameter fielddata.path leads to a shapefile.

`newgdb.name` Character string containing the name of a new geodatabase for storing output shapefile. Default ("None") assumes the parameter output.gdb.path leads to a valid geodatabase.

`quad.sel` Character string or vector containing the names of eCognition quadpolygons, limits the function to looping through only the selected polygons. Default ("None") will loop through all polygons. To select a list of quadpolygons, use the function 'fsvm::set_extent_manual()'.

`export` A boolean statement. If TRUE, the function will save the output as an .RData file to the output.folder.path. If FALSE, object will not be automatically saved.

`survey` A boolean statement. If TRUE, the function will treat the output as a properly formatted file containing survey data. If FALSE, only 'QuadPolyID' will be formatted.

Value

An R object containing merged and formatted output tables containing field data coordinates associated with QuadPoly_IDs for eCognition polygons. These will be used later for running machine learning suite of models for predicting fine scale vegetation presence/absence and percent cover. Object is saved in the provided file path (output.folder.path).

Troubleshooting

This function can be very finicky because of the Python-to-R interfacing. Crashing can be common if the parameters are not specified exactly. After a crash, the best thing to do is save your work and restart R (Ctrl + Shift + F10) and/or close R completely and reopen it. This is to make sure the connection to the file geodatabase is completely closed before trying to proceed. In the current version, it is best to leave the parameter 'quad.sel' at its default of 'None'. Otherwise, a character string with the exact name of a single specific quadpolygon may also work. If function is being particularly finicky, try with 'export' parameter as FALSE.

py_get_centroids *Get centroid locations for eCognition polygons*

Description

Get centroid locations for eCognition polygons

Usage

```
py_get_centroids(
  py.path = "C:/Python27/ArcGIS10.6/python.exe",
  quad.path = "A:/Fine scale vegetation analysis/dbases_4modeling/blank_polys100k.gdb",
  output.folder.path = "C:/Users/rritson/Documents/Output",
  quad.sel = "None"
)
```

Arguments

`py.path` Character string of file path to ArcGIS installed version of Python (required for running 'arcpy' module). Default is the file path where this is typically stored by ESRI. Function will not run without a valid version of ArcGIS installed before hand and must be run in a 32-bit R session.

`quad.path` A character string corresponding with the file path to the folder containing the 100k USGS quads.

`output.folder.path` Character string of file path to a valid folder for containing output *.dbf tables.

`quad.sel` Character string or vector containing the names of eCognition quadpolygons, limits the function to looping through only the selected polygons.

rateROC *Rate and Summarize Random Forest ROC values*

Description

Rate and Summarize Random Forest ROC values

Usage

```
rateROC(

  filepath = "D:/Fine scale vegetation analysis/understory_veg_model/ranger2/models/G2/presence"
)
```

Arguments

`filepath`	File path to RDS full model files (random forest)

Value

A data frame containing the file name (species model), ROC value, and Rating (A-F)

rectify_taxa *Rectify taxonomic names and assign modeling groups*

Description

Uses Integrated Taxonomic Information System to assign current accepted name and taxonomic serial number (TSN) to field data by modeling group ("G1","G2","G3",and "G4").

Usage

```
rectify_taxa(dat, scientific.name = "SpeciesName")
```

Arguments

`dat`	A data frame with at least one column containing taxonomic names of plants
`scientific.name`	A character string corresponding with the column in the data frame which contains the taxonomic plant names.

Value

Returns the input data frame appended with 8 additional columns: "G1", "G2", "G3", "G4", "G1TSN", "G2TSN", "G3TSN", and "G4TSN". "G1" corresponds with the accepted genera name and "G1TSN" the corresponding taxonomic serial number. "G2" corresponds with species (Genus and specific epithet only), "G3" corresponds with subspecies only (Genus specific epithet ssp. subspecies), and "G4" corresponds with varieties only (Genus specific epithet var. variety). Note, for example that "G2" and "G2TSN" would be identical for subspecies and varieties with the same taxonomic parent.

Troubleshooting

This function does not work well with very large data sets. If you need to rectify >1000 names, it will be best to break it into batches.

---

select_quads *Select 100k eCognition Quadpolygons*

Description

Lists the 100k eCognition Quadpolygons containing field data. Used with `set_extent_manual` and `py_extract_quadpolyID`

Usage

```
select_quads(
  dat = fielddata,
  x = "X_Coord",
  y = "Y_Coord",
  lib_loc = "//hqwildstat/C$/Program Files/R/R-4.0.5/library"
)
```

Arguments

`dat`	A data frame containing the coordinates of field data. Must be in IDTM projection.
`x`	A character string corresponding with the column in the data frame containing the x coordinate.
`y`	A character string corresponding with the column in the data frame containing the y coordinate.
`lib_loc`	A character string corresponding with the file path to the R library.

Value

A vector containing the names of the 100k quadpolygon IDs which intersect the field data.

---

set_extent_manual *Set Manual Spatial Extent*

Description

Returns a list of quadpolygons which correspond with a selected extent. This includes choices of specific IDFG Regions (1-7), IDFG Game Management Units, or UIDs of 100k or 24k quadpolygons.

Usage

```
set_extent_manual(
  extent = c("Region", "GMU", "UID_100k", "UID_24k"),
  selections = choices,
  lib_loc = "//hqwildstat/C$/Program Files/R/R-4.0.5/library"
)
```

Arguments

`extent`	A character string corresponding with the extent type you want to choose from. Choices are "Region", "GMU", "UID_110k", or "UID_24k". Multiple choices can be selected and stored as character vector, however this is probably less useful.
`selections`	A character vector corresponding with the selection(s) within the extent type.
`lib_loc`	A character string corresponding with the location of the R library.

Value

A character vector containing the names of 100k quadpolygons corresponding with the selected location extents. These name should be able to be passed to 'py_extract_quadpolyID' as quad selections.

See Also

py_extract_quadpolyID

`usda_resolve` *Rectify USDA Plant Codes*

Description

Assign scientific name based on a valid USDA plant code or assign a USDA code based on a valid scientific name. LPI codes ("HL","WL","AF", etc.) are only translated to modeling designations ("No Veg","UNKNOWN") when 'resolve = "c2n".

Usage

```
usda_resolve(dat = dat, target = "Code", resolve = c("c2n", "n2c"))
```

Arguments

`dat`	A data frame containing raw field data with valid USDA Plant codes
`target`	A character string corresponding with the column in the data frame to be translated. Either the USDA code when 'resolve' parameter is "c2n" or the scientific name when 'resolve' is "n2c".
`resolve`	A character string corresponding with type of the direction of translation. Argument may either be "c2n" to translate valid USDA codes to scientific names or "n2c" to translate valid scientific names to USDA codes.

Value

A data frame containing the original data with an additional column corresponding with the translation. Either "Scientific.Name" if resolve parameter is "c2n" or "Symbol" if resolve is "n2c".

Index