Package 'waves'

September 17, 2020

Title Vis-NIR Spectral Analysis Wrapper

Version 0.1.0

Description Originally designed application in the context of resource-limited plant research and breeding programs, 'waves' provides an open-source solution to spectral data processing and model development by bringing useful packages together into a streamlined pipeline. This package is wrapper for functions related to the analysis of point visible and near-infrared reflectance measurements. It includes visualization, filtering, aggregation, preprocessing, cross-validation set formation, model training, and prediction functions to enable open-source association of spectral and reference data.
Specialized cross-validation schemes are described in detail in Jarquín et al. (2017) <doi:10.3835/plantgenome2016.12.0130>. Example data is from Ikeogu et al. (2017) <doi:10.1371/journal.pone.0188918>.

URL https://github.com/GoreLab/waves

Maintainer Jenna Hershberger <jmh579@cornell.edu>

BugReports https://github.com/GoreLab/waves/issues

Depends R (>= 3.5)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports dplyr, prospectr, spectacles, caret, pls, randomForest, wesanderson, magrittr, tidyselect, ggplot2, tidyr (>= 1.0), stringr, rlang

RoxygenNote 6.1.1

Suggests testthat (>= 2.1.0)

NeedsCompilation no

Author Jenna Hershberger [aut, cre] (<https://orcid.org/0000-0002-3147-6867>), Michael Gore [ths], NSF BREAD IOS-1543958 [fnd]

Repository CRAN

Date/Publication 2020-09-17 12:10:02 UTC

R topics documented:

`AggregateSpectra` *Aggregate data based on grouping variables and a user-provided function*

Description

Use grouping variables to collapse spectral `data.frame` by mean or median. Recommended for use after `FilterSpectra`

Usage

```
AggregateSpectra(df, grouping.colnames, reference.value.colname,
  agg.function)
```

Arguments

`df`	`data.frame` object containing one or multiple columns of grouping variables (must be consistent within each group), column of reference values (optional), and columns of spectra. Spectral column names must start with "X".
`grouping.colnames`	Names of columns to be used as grouping variables. Minimum 2 variables required. Default is c("trial", "plot").
`reference.value.colname`	Name of reference column to be aggregated along with spectra. Default is "reference"
`agg.function`	Name of function (string format) to be used for sample aggregation. Must be either "mean" or "median". Default is "mean".

Value

`data.frame` object `df` aggregated based on grouping column by `agg.function`

Author(s)

Jenna Hershberger <jmh579@cornell.edu>

Examples

```
library(magrittr)
aggregated.test <- ikeogu.2017 %>%
  dplyr::select(-TCC) %>%
  na.omit() %>%
  AggregateSpectra(grouping.colnames = c("study.name"),
                   reference.value.colname = "DMC.oven",
                   agg.function = "mean")
aggregated.test[1:5, 1:5]
```

`DoPreprocessing` *Preprocess spectral data according to user-designated method*

Description

Preprocessing, also known as pretreatment, is often used to increase the signal to noise ratio in vis-NIR datasets. The *waves* function `DoPreprocessing` applies common spectral preprocessing methods such as standard normal variate and the Savitzky-Golay filter.

Usage

```
DoPreprocessing(df, test.data = NULL, preprocessing.method = 1,
  wavelengths = 740:1070)
```

Arguments

`df` `data.frame` object containing spectral data. First column(s) (optional) include metadata (with or without reference value column) followed by spectral columns. Spectral column names must be formatted as "X" followed by wavelength Include no other columns to right of spectra! No missing values permitted.

`test.data` `data.frame` object with same format as train.data. Will be appended to `df` during preprocessing so that the same transformations are applied to each row. Default is `NULL`.

`preprocessing.method`
Number or list of numbers 1:13 corresponding to desired pretreatment method(s):

- 1 = raw data (default)
- 2 = standard normal variate (SNV)
- 3 = SNV and first derivative
- 4 = SNV and second derivative
- 5 = first derivative
- 6 = second derivative
- 7 = Savitzky–Golay filter (SG)
- 8 = SNV and SG
- 9 = gap segment derivative (window size = 11)
- 10 = SG and first derivative (window size = 5)

- 11 = SG and first derivative (window size = 11)
- 12 = SG and second derivative (window size = 5)
- 13 = SG and second derivative (window size = 11)

`wavelengths` List of wavelengths represented by each column in `df`. Default is 740:1070.

Value

Preprocessed `df`‘ (or list of `data.frames`) with reference column intact

Author(s)

Jenna Hershberger <jmh579@cornell.edu>

Examples

```
DoPreprocessing(df = ikeogu.2017, wavelengths = 350:2500)[1:5,1:5]
```

`FilterSpectra` *Filter spectral data frame based on Mahalanobis distance*

Description

Determine Mahalanobis distances of observations (rows) within a given `data.frame` with spectral data. Option to filter out observations based on these distances.

Usage

```
FilterSpectra(df, filter, return.distances, num.col.before.spectra,
  window.size, verbose)
```

Arguments

`df` a `data.frame` object containing columns of spectra and rows of observations. May also contain columns of metadata to the left of the spectra.

`filter` boolean that determines whether or not the input `data.frame` will be filtered. If `TRUE`, `df` will be filtered according to squared Mahalanobis distance with a 95% cutoff from a chi-square distribution with degrees of freedom = number of spectral columns. If `FALSE`, a column of squared Mahalanobis distances `h.distance` will be added to the right side of df and all rows will be returned. Default is `TRUE`.

`return.distances` boolean that determines whether a column of squared Mahalanobis distances will be included in output `data.frame`. If `TRUE`, a column of Mahalanobis distances for each row will be added to the right side of `df`. Default is `FALSE`.

`num.col.before.spectra` number of columns to the left of the spectral matrix in `df`. Default is 4.

`window.size` number defining the size of window to use when calculating the covariance of the spectra (required to calculate Mahalanobis distance). Default is 10.

verbose — If TRUE, the number of rows removed through filtering will be printed to the console. Default is TRUE.

Details

This function uses a chi-square distribution with 95% cutoff where degrees of freedom = number of wavelengths (columns) in the input data.frame.

Value

If filter is TRUE, returns filtered data frame df and reports the number of rows removed. The Mahalanobis distance with a cutoff of 95% of chi-square distribution (degrees of freedom = number of wavelengths) is used as filtering criteria. If filter is FALSE, returns full input df with column h.distances containing the Mahalanobis distance for each row.

Author(s)

Jenna Hershberger <jmh579@cornell.edu>

References

Johnson, R.A., and D.W. Wichern. 2007. Applied Multivariate Statistical Analysis (6th Edition). pg 189

Examples

```
library(magrittr)
filtered.test <- ikeogu.2017 %>%
  dplyr::select(-TCC) %>%
  na.omit() %>%
  FilterSpectra(df = .,
                filter = TRUE,
                return.distances = TRUE,
                num.col.before.spectra = 5,
                window.size = 15)
filtered.test[1:5, c(1:5, (ncol(filtered.test)-5):ncol(filtered.test))]
```

FormatCV — *Format multiple trials with or without overlapping genotypes into training and test sets according to user-provided cross validation scheme*

Description

Standalone function that is also used within TrainSpectralModel to divide trials or studies into training and test sets based on overlap in trial environments and genotype entries

Usage

```
FormatCV(trial1, trial2, trial3 = NULL, cv.scheme, seed = NULL,
  remove.genotype = FALSE)
```

Arguments

trial1	data.frame object that is for use only when cv.scheme is provided. Contains the trial to be tested in subsequent model training functions. The first column contains unique identifiers, second contains genotypes, third contains reference values, followed by spectral columns. Include no other columns to right of spectra! Column names of spectra must start with "X", reference column must be named "reference", and genotype column must be named "genotype".
trial2	data.frame object that is for use only when cv.scheme is provided. This data.frame contains a trial that has overlapping genotypes with trial1 but that were grown in a different site/year (different environment). Formatting must be consistent with trial1.
trial3	data.frame object that is for use only when cv.scheme is provided. This data.frame contains a trial that may or may not contain genotypes that overlap with trial1. Formatting must be consistent with trial1.
cv.scheme	A cross validation (CV) scheme from Jarquín et al., 2017. Options for cv.scheme include: • "CV1": untested lines in tested environments • "CV2": tested lines in tested environments • "CV0": tested lines in untested environments • "CV00": untested lines in untested environments
seed	Number used in the function set.seed() for reproducible randomization. If NULL, no seed is set. Default is NULL.
remove.genotype	boolean that, if TRUE, removes the "genotype" column is removed from the output data.frame. Default is FALSE.

Details

Use of a cross-validation scheme requires a column in the input data.frame named "genotype" to ensure proper sorting of training and test sets. Variables trial1 and trial2 are required, while trial 3 is optional.

Value

List of data.frames (training set, test set) compiled according to user-provided cross validation scheme.

Author(s)

Jenna Hershberger <jmh579@cornell.edu>

References

Jarquín, D., C. Lemes da Silva, R. C. Gaynor, J. Poland, A. Fritz, R. Howard, S. Battenfield, and J. Crossa. 2017. Increasing genomic-enabled prediction accuracy by modeling genotype × environment interactions in Kansas wheat. Plant Genome 10(2):1-15. <doi:10.3835/plantgenome2016.12.0130>

Examples

```
# Must have a column called "genotype", so we'll create a fake one for now
# We will use CV00, which does not require any overlap in genotypes
# In real scenarios, CV schemes that rely on genotypes should not be applied when
# genotypes are unknown, as in this case.
library(magrittr)
trials <- ikeogu.2017 %>%
    dplyr::mutate(genotype = 1:nrow(ikeogu.2017)) %>% # fake for this example
    dplyr::rename(reference = DMC.oven) %>%
    dplyr::select(study.name, sample.id, genotype, reference,
                  dplyr::starts_with("X"))
trial1 <- trials %>%
  dplyr::filter(study.name == "C16Mcal") %>%
  dplyr::select(-study.name)
trial2 <- trials %>%
  dplyr::filter(study.name == "C16Mval") %>%
  dplyr::select(-study.name)
cv.list <- FormatCV(trial1 = trial1, trial2 = trial2, cv.scheme = "CV00",
                    remove.genotype = TRUE)
cv.list[[1]][1:5, 1:5]
```

`ikeogu.2017` *Example vis-NIRS and reference dataset*

Description

The 'ikeogu.2017' data set contains raw vis-NIRS scans, total carotenoid content, and cassava root dry matter content (using the oven method) from the 2017 PLOS One paper by Ikeogu et al. This dataset contains a subset of the original scans and reference values from the supplementary files of the paper. 'ikeogu.2017' is a 'data.frame' that contains the following columns:

- study.name = Name of the study as described in Ikeogu et al. (2017).
- sample.id = Unique identifier for each individual root sample
- DMC.oven = Cassava root dry matter content, the percentage of dry weight relative to fresh weight of a sample after oven drying.
- TCC = Total carotenoid content ($\mu g/g$, unknown whether on a fresh or dry weight basis) as measured by high performance liquid chromatography
- X350:X2500 = spectral reflectance measured with the QualitySpec Trek: S-10016 vis-NIR spectrometer. Each cell represents the mean of 150 scans on a single root at a single wavelength.

Usage

```
ikeogu.2017
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 175 rows and 2155 columns.

Author(s)

Original authors: Ikeogu, U.N., F. Davrieux, D. Dufour, H. Ceballos, C.N. Egesi, and J. Jannink. Reformatted by Jenna Hershberger.

References

Ikeogu, U.N., F. Davrieux, D. Dufour, H. Ceballos, C.N. Egesi, et al. 2017. Rapid analyses of dry matter content and carotenoids in fresh cassava roots using a portable visible and near infrared spectrometer (Vis/NIRS). PLOS One 12(12): 1–17. doi: 10.1371/journal.pone.0188918.

Examples

```
library(magrittr)
library(ggplot2)
data(ikeogu.2017)
ikeogu.2017[1:10,1:10]
ikeogu.2017 %>%
    dplyr::select(-starts_with("X")) %>%
    dplyr::group_by(study.name) %>%
    tidyr::gather(trait, value, c(DMC.oven:TCC), na.rm = TRUE) %>%
    ggplot2::ggplot(aes(x = study.name, y = value, fill = study.name)) +
        facet_wrap(~ trait, scales = 'free_y', nrow = 2) +
        geom_boxplot()
```

PlotSpectra *Plot spectral data, highlighting outliers as identified using Mahalanobis distance*

Description

Generates a ggplot object of given spectra, with wavelength on the x axis and given spectral values on the y. Mahalanobis distance is used to calculate outliers, which are both identified on the plot. Rows from the original dataframe are printed to the console for each outlier that is identified.

Usage

```
PlotSpectra(input.df, wavelengths, num.col.before.spectra = 1,
  window.size = 10, verbose = TRUE)
```

Arguments

input.df	data.frame object containing columns of spectra. Spectral columns must be labeled with an "X" and then the wavelength (example: "X740" = 740nm). Left-most column must be unique ID. May also contain columns of metadata between the unique ID and spectral columns. Cannot contain any missing values
wavelengths	List of wavelengths (numerical format) represented by each spectral column in input.df
num.col.before.spectra	Number of columns to the left of the spectral matrix (including unique ID). Default is 1.
window.size	number defining the size of window to use when calculating the covariance of the spectra (required to calculate Mahalanobis distance). Default is 10.
verbose	If TRUE, the number of rows removed through filtering will be printed to the console. Default is TRUE.

Value

If verbose, prints unique ID and metadata for rows identified as outliers. Returns plot of spectral data with non-outliers in blue and outliers in red. X-axis is wavelengths and y-axis is spectral values.

Author(s)

Jenna Hershberger <jmh579@cornell.edu>

Examples

```
library(magrittr)
ikeogu.2017 %>%
  dplyr::rename(unique.id = sample.id) %>%
  dplyr::select(unique.id, dplyr::everything(), -TCC) %>%
  na.omit() %>%
  PlotSpectra(input.df = .,
              wavelengths = 350:2500,
              num.col.before.spectra = 5,
              window.size = 15)
```

PredictFromSavedModel *Use provided model object to predict trait values with input dataset*

Description

Loads an existing model and cross-validation performance statistics (created with SaveModel) and makes predictions based on new spectra.

Usage

```
PredictFromSavedModel(input.data, model.stats.location, model.location,
  wavelengths = 740:1070, model.method = "pls")
```

Arguments

`input.data`	`data.frame` object of spectral data for input into a spectral prediction model. First column contains unique identifiers followed by spectral columns. Include no other columns to right of spectra! Column names of spectra must start with "X".
`model.stats.location`	String containing file path (including file name) to save location of "(model.name)_stats.csv" as output from the SaveModel function.
`model.location`	String containing file path (including file name) to location where the trained model ("(model.name).Rds") was saved as output by the `SaveModel` function.
`wavelengths`	List of wavelengths represented by each column in `input.data`
`model.method`	Model type to use for training. Valid options include: • "pls": Partial least squares regression (Default) • "rf": Random forest • "svmLinear": Support vector machine with linear kernel • "svmRadial": Support vector machine with radial kernel

Value

`data.frame` object of predictions for each sample (row). First column is unique identifier supplied by `input.data` and second is predicted values

Author(s)

Jenna Hershberger <jmh579@cornell.edu>

Examples

```
## Not run:
ikeogu.2017 %>%
  dplyr::select(sample.id, dplyr::starts_with("X")) %>%
  PredictFromSavedModel(input.data = .,
                        model.stats.location = paste0(getwd(),
                                                      "/my_model_stats.csv"),
                        model.location = paste0(getwd(), "/my_model.Rds"),
                        wavelengths = 350:2500)

## End(Not run)
```

SaveModel *Save spectral prediction model and model performance statistics*

Description

Saves spectral prediction model and model statistics to `model.save.folder` as `model.name.Rds` and `model.name_stats.csv` respectively

Usage

```
SaveModel(df, save.model = TRUE, autoselect.preprocessing = TRUE,
  preprocessing.method = NULL, model.save.folder = NULL,
  model.name = "PredictionModel", best.model.metric = "RMSE",
  tune.length = 50, model.method = "pls", num.iterations = 10,
  wavelengths = 740:1070, stratified.sampling = TRUE,
  cv.scheme = NULL, trial1 = NULL, trial2 = NULL, trial3 = NULL,
  verbose = TRUE)
```

Arguments

`df` `data.frame` object. First column contains unique identifiers, second contains reference values, followed by spectral columns. Include no other columns to right of spectra! Column names of spectra must start with "X" and reference column must be named "reference"

`save.model` If `TRUE`, the trained model will be saved in .Rds format to the location specified by `model.save.folder`. If `FALSE`, model will be output by function but will not save to file. Default is `TRUE`.

`autoselect.preprocessing` Boolean that, if `TRUE`, will choose the preprocessing method for the saved model using the `best.model.metric`. If `FALSE`, the user must supply the preprocessing method (1-12, see `DoPreprocessing()` documentation for more information) of the saved model. Default is `TRUE`.

`preprocessing.method` Number or list of numbers 1:13 corresponding to desired pretreatment method(s):

- 1 = raw data (default)
- 2 = standard normal variate (SNV)
- 3 = SNV and first derivative
- 4 = SNV and second derivative
- 5 = first derivative
- 6 = second derivative
- 7 = Savitzky–Golay filter (SG)
- 8 = SNV and SG
- 9 = gap segment derivative (window size = 11)
- 10 = SG and first derivative (window size = 5)
- 11 = SG and first derivative (window size = 11)

- 12 = SG and second derivative (window size = 5)
- 13 = SG and second derivative (window size = 11)

`model.save.folder`
: Path to folder where model will be saved. If not provided, will save to working directory.

`model.name`
: Name that model will be saved as in `model.save.folder`. Default is "PredictionModel".

`best.model.metric`
: Metric used to decide which model is best. Must be either "RMSE" or "Rsquared"

`tune.length`
: Number delineating search space for tuning of the PLSR hyperparameter `ncomp`. Default is 50.

`model.method`
: Model type to use for training. Valid options include:

  - "pls": Partial least squares regression (Default)
  - "rf": Random forest
  - "svmLinear": Support vector machine with linear kernel
  - "svmRadial": Support vector machine with radial kernel

`num.iterations`
: Number of training iterations to perform

`wavelengths`
: List of wavelengths represented by each column in `df`

`stratified.sampling`
: If `TRUE`, training and test sets will be selected using stratified random sampling. This term is only used if `test.data == NULL`. Default is `TRUE`.

`cv.scheme`
: A cross validation (CV) scheme from Jarquín et al., 2017. Options for cv.scheme include:

  - "CV1": untested lines in tested environments
  - "CV2": tested lines in tested environments
  - "CV0": tested lines in untested environments
  - "CV00": untested lines in untested environments

`trial1`
: `data.frame` object that is for use only when `cv.scheme` is provided. Contains the trial to be tested in subsequent model training functions. The first column contains unique identifiers, second contains genotypes, third contains reference values, followed by spectral columns. Include no other columns to right of spectra! Column names of spectra must start with "X", reference column must be named "reference", and genotype column must be named "genotype".

`trial2`
: `data.frame` object that is for use only when `cv.scheme` is provided. This data.frame contains a trial that has overlapping genotypes with `trial1` but that were grown in a different site/year (different environment). Formatting must be consistent with `trial1`.

`trial3`
: `data.frame` object that is for use only when `cv.scheme` is provided. This data.frame contains a trial that may or may not contain genotypes that overlap with `trial1`. Formatting must be consistent with `trial1`.

`verbose`
: If `TRUE`, the number of rows removed through filtering will be printed to the console. Default is `TRUE`.

Details

Wrapper that uses DoPreprocessing, FormatCV, and TrainSpectralModel functions.

Value

List of model stats (in data.frame) and trained model object. Saves both to model.save.folder as well. To use optimally trained model for predictions, use tuned parameters from $bestTune

Author(s)

Jenna Hershberger <jmh579@cornell.edu>

Examples

```
library(magrittr)
test.model <- ikeogu.2017 %>%
  dplyr::filter(study.name == "C16Mcal") %>%
  dplyr::rename(reference = DMC.oven) %>%
  dplyr::select(sample.id, reference, dplyr::starts_with("X")) %>%
  na.omit() %>%
  SaveModel(df = ., save.model = FALSE,
            autoselect.preprocessing = TRUE,
            model.name = "my_prediction_model",
            tune.length = 50, num.iterations = 10,
            wavelengths = 350:2500)
summary(test.model[1])
test.model[2]
```

TestModelPerformance *Test the performance of spectral models*

Description

Wrapper that trains models based spectral data to predict reference values and reports model performance statistics

Usage

```
TestModelPerformance(train.data, num.iterations, test.data = NULL,
  preprocessing = TRUE, wavelengths = 740:1070, tune.length = 50,
  model.method = "pls", output.summary = TRUE,
  rf.variable.importance = FALSE, stratified.sampling = TRUE,
  cv.scheme = NULL, trial1 = NULL, trial2 = NULL, trial3 = NULL,
  split.test = FALSE, verbose = TRUE)
```

Arguments

train.data	data.frame object of spectral data for input into a spectral prediction model. First column contains unique identifiers, second contains reference values, followed by spectral columns. Include no other columns to right of spectra! Column names of spectra must start with "X" and reference column must be named "reference".
num.iterations	Number of training iterations to perform
test.data	data.frame with same specifications as df. Use if specific test set is desired for hyperparameter tuning. If NULL, function will automatically train with a stratified sample of 70%. Default is NULL.
preprocessing	If TRUE, 12 preprocessing methods will be applied and their performance analyzed. If FALSE, input data is analyzed as is (raw). Default is FALSE.
wavelengths	List of wavelengths represented by each column in train.data
tune.length	Number delineating search space for tuning of the PLSR hyperparameter ncomp. Default is 50.
model.method	Model type to use for training. Valid options include: • "pls": Partial least squares regression (Default) • "rf": Random forest • "svmLinear": Support vector machine with linear kernel • "svmRadial": Support vector machine with radial kernel
output.summary	boolean that controls function output. • If TRUE, a summary df will be output (1st row = means, 2nd row = standard deviations). Default is TRUE. • If FALSE, entire results data frame will be output
rf.variable.importance	boolean that: • If TRUE, model.method must be set to "rf". Returns a list with a model performance data.frame and a second data.frame with variable importance values for each wavelength for each training iteration. If return.model is also TRUE, returns list of three elements with trained model first, model performance second, and variable importance last. Dimensions are nrow = num.iterations, ncol = length(wavelengths). • If FALSE, no variable importance is returned. Default is FALSE.
stratified.sampling	If TRUE, training and test sets will be selected using stratified random sampling. This term is only used if test.data == NULL. Default is TRUE.
cv.scheme	A cross validation (CV) scheme from Jarquín et al., 2017. Options for cv.scheme include: • "CV1": untested lines in tested environments • "CV2": tested lines in tested environments • "CV0": tested lines in untested environments • "CV00": untested lines in untested environments

trial1	data.frame object that is for use only when cv.scheme is provided. Contains the trial to be tested in subsequent model training functions. The first column contains unique identifiers, second contains genotypes, third contains reference values, followed by spectral columns. Include no other columns to right of spectra! Column names of spectra must start with "X", reference column must be named "reference", and genotype column must be named "genotype".
trial2	data.frame object that is for use only when cv.scheme is provided. This data.frame contains a trial that has overlapping genotypes with trial1 but that were grown in a different site/year (different environment). Formatting must be consistent with trial1.
trial3	data.frame object that is for use only when cv.scheme is provided. This data.frame contains a trial that may or may not contain genotypes that overlap with trial1. Formatting must be consistent with trial1.
split.test	boolean that allows for a fixed training set and a split test set. Example// train model on data from two breeding programs and a stratified subset (70%) of a third and test on the remaining samples (30%) of the third. If FALSE, the entire provided test set test.data will remain as a testing set or if none is provided, 30% of the provided train.data will be used for testing. Default is FALSE.
verbose	If TRUE, the number of rows removed through filtering will be printed to the console. Default is TRUE.

Details

Calls DoPreprocessing, FormatCV, and TrainSpectralModel functions.

Value

data.frame with model performance statistics in summary format (2 rows, one with mean and one with standard deviation of all training iterations) or in long format (number of rows = num.iterations). **Note** if preprocessing = TRUE, only the first mean of summary statistics for all iterations of training are provided for each technique. Included summary statistics:

- Tuned parameters depending on the model algorithm:
  - **Best.n.comp**, the best number of components
  - **Best.ntree**, the best number of trees in an RF model
  - **Best.mtry**, the best number of variables to include at every decision point in an RF model
- **RMSECV**, the root mean squared error of cross-validation
- **R2cv**, the coefficient of multiple determination of cross-validation for PLSR models
- **RMSEP**, the root mean squared error of prediction
- **R2p**, the squared Pearson's correlation between predicted and observed test set values
- **RPD**, the ratio of standard deviation of observed test set values to RMSEP
- **RPIQ**, the ratio of performance to interquartile difference
- **CCC**, the concordance correlation coefficient
- **Bias**, the average difference between the predicted and observed values
- **SEP**, the standard error of prediction
- **R2sp**, the squared Spearman's rank correlation between predicted and observed test set values

Author(s)

Jenna Hershberger <jmh579@cornell.edu>

Examples

```
library(magrittr)
ikeogu.2017 %>%
  dplyr::rename(reference = DMC.oven) %>%
  dplyr::rename(unique.id = sample.id) %>%
  dplyr::select(unique.id, reference, dplyr::starts_with("X")) %>%
  na.omit() %>%
  TestModelPerformance(train.data = .,
                       tune.length = 3,
                       num.iterations = 3,
                       preprocessing = FALSE,
                       wavelengths = 350:2500)
```

Index