Package: labelnoise (via r-universe)

November 14, 2026

Title Find Label Errors in Classification Data with Confident Learning

Version 0.1.0

Description Finds examples whose given class label is likely wrong, using out-of-sample predicted probabilities from any classifier. Implements confident learning (Northcutt, Jiang and Chuang, 2021) <doi:10.1613/jair.1.12125>: per-class confidence thresholds, the confident joint, calibrated estimates of the joint distribution of given and true labels, noise transition matrices, and five rank-and-prune rules for flagging label issues. Also provides model-agnostic label quality scores (Kuan and Mueller, 2022) <doi:10.5281/zenodo.18168246> and a per-class label health summary. Inputs can be a label vector with a probability matrix, a data frame of predictions, resampling results from 'tune', predictions from 'mlr3', or a fitted 'caret' model.

License MIT + file LICENSE

URL https://ridsonap.is-a.dev/labelnoise/, https://github.com/ridsonap/labelnoise

BugReports https://github.com/ridsonap/labelnoise/issues

Encoding UTF-8

Depends R (>= 4.1.0)

Imports cli, rlang (>= 1.1.0), stats, utils, tibble, tidyselect

Suggests caret, knitr, lgr, mlr3, nnet, parsnip, rmarkdown, rsample, testthat (>= 3.0.0), tune, withr, workflows, yardstick

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Ridson Al Farizal P [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-0617-0214>)

Maintainer Ridson Al Farizal P <ridsonalfarizal15@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-11-13 17:10:08 UTC

RemoteUrl https://github.com/cran/labelnoise

RemoteRef HEAD

RemoteSha 1bae123c1d1650db1fbe1491b84b5e4254faed3c

Contents

class_thresholds 2
confident_joint 3
estimate_joint 4
find_label_issues 5
health_summary 6
label_inputs 8
label_quality_scores 9
noise_matrices 11
num_label_issues 12
simulate_label_noise 13

Index **14**

`class_thresholds` *Per-class confidence thresholds*

Description

The threshold of class `j` is the mean predicted probability of class `j` among examples whose given label is `j` (Northcutt et al., 2021, Eq. 2). An example is a confident member of class `j` when its predicted probability for `j` reaches this threshold.

Usage

```
class_thresholds(x, ...)
```

Arguments

`x`	Labels or an object holding labels and predicted probabilities. See Description.
`...`	Passed to `label_inputs()`: for example `pred_probs` when `x` is a label vector, or `truth` when `x` is a data frame.

Value

A named numeric vector with one threshold per class. Classes without any given label get `Inf`, so no example is counted as a confident member of them.

References

Northcutt, C., Jiang, L., & Chuang, I. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 1373–1411. doi:10.1613/jair.1.12125

Examples

```
sim <- simulate_label_noise(n = 200, seed = 1)
class_thresholds(sim, truth = given)
```

`confident_joint` *The confident joint*

Description

`confident_joint()` counts, for every pair of given label `i` and (estimated) true label `j`, the examples labelled `i` that are confident members of class `j` (Northcutt et al., 2021, Eq. 1 and Algorithm 1). An example is a confident member of class `j` when its predicted probability for `j` is at least the threshold of `j` (see `class_thresholds()`). When it is a confident member of several classes, the one with the largest predicted probability among them is used. Examples that are confident members of no class are not counted.

With `calibrate = TRUE`, each row is rescaled so that it sums to the number of examples with that given label (the numerator of Eq. 3). Rows with no confident examples are treated as noise free.

Off-diagonal cells are label errors: examples given label `i` that look like they belong to class `j`.

Usage

```
confident_joint(x, ..., calibrate = TRUE)
```

Arguments

`x`	Labels or an object holding labels and predicted probabilities. See Description.
`...`	Passed to `label_inputs()`: for example `pred_probs` when `x` is a label vector, or `truth` when `x` is a data frame.
`calibrate`	Rescale rows to the observed class counts?

Value

A square matrix with dimension names `given` (rows) and `true` (columns). Integer counts when `calibrate = FALSE`; real-valued calibrated counts otherwise.

References

Northcutt, C., Jiang, L., & Chuang, I. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 1373–1411. doi:10.1613/jair.1.12125

See Also

`estimate_joint()`, `noise_matrices()`, `num_label_issues()`

Examples

```
sim <- simulate_label_noise(n = 500, seed = 1)
confident_joint(sim, truth = given, calibrate = FALSE)
confident_joint(sim, truth = given)
```

---

estimate_joint *Estimate the joint distribution of given and true labels*

Description

Normalises the calibrated confident joint so that it sums to one (Northcutt et al., 2021, Eq. 3 and Algorithm 2). Entry `[i, j]` estimates the proportion of examples that have given label `i` and true label `j`.

Usage

```
estimate_joint(x, ...)
```

Arguments

`x`	Labels or an object holding labels and predicted probabilities. See Description.
`...`	Passed to `label_inputs()`: for example `pred_probs` when `x` is a label vector, or `truth` when `x` is a data frame.

Value

A square matrix with dimension names `given` and `true` that sums to one. Its row sums equal the observed class proportions.

References

Northcutt, C., Jiang, L., & Chuang, I. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 1373–1411. doi:10.1613/jair.1.12125

Examples

```
sim <- simulate_label_noise(n = 500, seed = 1)
round(estimate_joint(sim, truth = given), 3)
```

`find_label_issues` *Find examples with likely label errors*

Description

Flags examples whose given label is likely wrong, using one of the rank and prune rules of confident learning (Northcutt et al., 2021, Sec. 3.2). Every rule needs out-of-sample predicted probabilities, for example from cross-validation.

- `"prune_by_noise_rate"` (default): for every pair of given label `i` and true label `j`, the estimated number of examples in that cell of the joint (see `estimate_joint()`) is flagged among the examples labelled `i`, choosing those where `p[j] - p[i]` is largest.
- `"prune_by_class"`: for every given label `i`, the estimated number of mislabelled examples is flagged among the examples labelled `i`, choosing those with the lowest self-confidence `p[i]`.
- `"both"`: examples flagged by both rules above.
- `"confident_learning"`: examples in the off-diagonal cells of the confident joint (see `confident_joint()`).
- `"predicted_neq_given"`: examples whose most likely class differs from the given label. A simple baseline that ignores class imbalance.

As in the cleanlab Python package, an example whose most likely class is its given label is never flagged, whichever rule is used: the classifier then offers no evidence against the label.

Flagged examples are candidates for review, not proven errors: ambiguous examples that genuinely fit several classes are flagged too.

Usage

```
find_label_issues(
  x,
  ...,
  filter_by = c("prune_by_noise_rate", "prune_by_class", "both", "confident_learning",
    "predicted_neq_given"),
  rank_by = c("self_confidence", "normalized_margin", "confidence_weighted_entropy"),
  frac_noise = 1,
  min_examples_per_class = 1
)
```

Arguments

`x`	Labels or an object holding labels and predicted probabilities. See Description.
`...`	Passed to `label_inputs()`: for example `pred_probs` when `x` is a label vector, or `truth` when `x` is a data frame.
`filter_by`	Rule used to flag examples, see Description.
`rank_by`	Label quality score reported in `label_quality`, see `label_quality_scores()`. Sort by it to review the most suspicious examples first.

`frac_noise` Fraction of the estimated number of label errors to flag, between 0 and 1. Only used by the `"prune_by_*"` and `"both"` rules. Lower values flag fewer, more certain examples.

`min_examples_per_class`
The rules never flag so many examples of a given label that fewer than this number would remain unflagged.

Value

A tibble with one row per example, in the order of the input, and columns

- `.row`: the row identifier (the `.row` of resampled predictions, or the position in the input);
- `given`: the given label;
- `predicted`: the class with the largest predicted probability;
- `label_quality`: the label quality score chosen by `rank_by`;
- `is_label_issue`: whether the example is flagged.

References

Northcutt, C., Jiang, L., & Chuang, I. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 1373–1411. doi:10.1613/jair.1.12125

See Also

`health_summary()` for a class-level overview.

Examples

```
sim <- simulate_label_noise(n = 500, noise_rate = 0.1, seed = 1)
issues <- find_label_issues(sim, truth = given)
issues

# Review the most suspicious examples first
flagged <- issues[issues$is_label_issue, ]
head(flagged[order(flagged$label_quality), ])

# How well did it do on this simulation?
table(flagged = issues$is_label_issue, mislabelled = sim$given != sim$true)
```

`health_summary` *Summarise label health by class*

Description

Gives a dataset-level and class-level overview of label quality from the estimated joint distribution of given and true labels (see `noise_matrices()`):

- an overall label health score, one minus the estimated share of mislabelled examples;
- for every class, the estimated number and share of mislabelled examples, and the noise rates;
- the pairs of classes that are most often confused with each other, which point to unclear class definitions or coding guidelines.

Usage

```
health_summary(x, ..., method = c("confident_joint", "calibrated"))
```

Arguments

`x`	Labels or an object holding labels and predicted probabilities. See Description.
`...`	Passed to `label_inputs()`: for example `pred_probs` when `x` is a label vector, or `truth` when `x` is a data frame.
`method`	Estimator of the number of label errors, see `num_label_issues()`.

Value

An object of class `label_health`, a list with

- `score`: the overall label health score;
- `n`, `num_label_issues`: the number of examples and the estimated number of mislabelled examples;
- `classes`: a tibble with one row per class, sorted from worst to best `label_quality`. `label_noise` is the estimated share of examples truly in the class that were given another label; `inverse_label_noise` the estimated share of examples given the label that truly belong to another class; `label_quality` is `1 - inverse_label_noise`.
- `overlapping_classes`: a tibble of class pairs with the estimated number of examples confused in either direction, sorted from most to least confused. Pairs that are never confused are left out.

References

Northcutt, C., Jiang, L., & Chuang, I. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 1373–1411. doi:10.1613/jair.1.12125

Examples

```
sim <- simulate_label_noise(n = 1000, noise_rate = 0.15, seed = 1)
health <- health_summary(sim, truth = given)
health
health$classes
```

label_inputs *Standardise labels and predicted probabilities*

Description

`label_inputs()` converts the many ways of supplying given labels and out-of-sample predicted probabilities into one validated object. Every other function in labelnoise calls it, so anything accepted here is accepted everywhere:

- a label vector (factor, character or integer) plus a probability matrix or data frame in `pred_probs`;
- a data frame of predictions in the yardstick convention: a `truth` column and one `.pred_<level>` column per class (as returned by `tune::collect_predictions()` or `augment()`);
- a `tune_results` object from `tune::fit_resamples()` or `tune::tune_grid()` fitted with `control_resamples(save_pred = TRUE)`;
- an mlr3 `PredictionClassif` or `ResampleResult` with `predict_type = "prob"`;
- a caret `train` object fitted with `classProbs = TRUE` and `savePredictions = "final"` (or `"all"`).

Predicted probabilities must be out-of-sample (for example from cross-validation). In-sample probabilities from a model that has seen the labels will hide most label errors.

When an example has several out-of-sample predictions (repeated cross-validation), its probabilities are averaged.

Other packages can add methods for their own prediction objects. A method should end by calling `label_inputs()` on a label vector and a probability matrix.

Usage

```
label_inputs(x, ...)

## S3 method for class 'labelnoise_inputs'
label_inputs(x, ...)

## Default S3 method:
label_inputs(x, pred_probs, ..., row_id = NULL)

## S3 method for class 'data.frame'
label_inputs(x, truth, ..., pred_probs = NULL)

## S3 method for class 'tune_results'
label_inputs(x, ..., parameters = NULL)

## S3 method for class 'PredictionClassif'
label_inputs(x, ...)

## S3 method for class 'ResampleResult'
```

```
label_inputs(x, ...)

## S3 method for class 'train'
label_inputs(x, ...)
```

Arguments

x	Labels or an object holding labels and predicted probabilities. See Description.
...	Passed on to methods. For data frames, a tidyselect specification of the probability columns.
pred_probs	A numeric matrix or data frame with one row per example and one column per class. Rows must sum to one. Columns are matched to labels by name when both have names, otherwise by position.
row_id	Integer identifiers returned in the `.row` column of `find_label_issues()`. Defaults to `seq_along(x)`.
truth	For data frames, the column holding the given labels.
parameters	For `tune_results` with several candidate models, a one-row data frame of tuning parameters, such as the output of `tune::select_best()`.

Value

An object of class `labelnoise_inputs`: a list with `labels` (factor), `pred_probs` (numeric matrix with one column per level of `labels`) and `row_id` (integer).

Examples

```
labels <- factor(c("a", "b", "b", "a"))
probs <- matrix(
  c(0.9, 0.1,
    0.2, 0.8,
    0.7, 0.3,
    0.6, 0.4),
  ncol = 2, byrow = TRUE, dimnames = list(NULL, c("a", "b"))
)
label_inputs(labels, probs)

# The yardstick convention
preds <- data.frame(class = labels, .pred_a = probs[, 1], .pred_b = probs[, 2])
label_inputs(preds, truth = class)
```

`label_quality_scores` *Label quality scores*

Description

Scores how likely each given label is to be correct, from 0 (probably wrong) to 1 (probably right). Scores depend only on the predicted probabilities `p` of an example and its given label `y` (Kuan and Mueller, 2022):

- `"self_confidence"`: `p[y]`, the predicted probability of the given label.
- `"normalized_margin"`: `(p[y] - max(p[-y]) + 1) / 2`, the gap between the given label and the most likely other class, rescaled to `[0, 1]`. Low when another class is clearly preferred.
- `"confidence_weighted_entropy"`: `ratio / (1 + ratio)` with `ratio = p[y] / H(p)`, where `H` is the entropy of `p` divided by `log(K)`. Also low for examples whose predictions are spread over many classes, such as examples that belong to none of the classes.

With `adjust = TRUE`, each class threshold `t` (see `class_thresholds()`) is subtracted from the predicted probabilities before scoring. This corrects for a classifier that systematically favours some classes. The adjusted probabilities are `(p - t + max(t)) / sum(p - t + max(t))`, which are non-negative and sum to one.

Usage

```
label_quality_scores(
  x,
  ...,
 method = c("self_confidence", "normalized_margin", "confidence_weighted_entropy"),
  adjust = FALSE
)
```

Arguments

`x`	Labels or an object holding labels and predicted probabilities. See Description.
`...`	Passed to `label_inputs()`: for example `pred_probs` when `x` is a label vector, or `truth` when `x` is a data frame.
`method`	Scoring method, see Description.
`adjust`	Adjust predicted probabilities by the class thresholds before scoring? Not available for `"confidence_weighted_entropy"`.

Value

A numeric vector with one score per example, in the order of `label_inputs(x, ...)$row_id`.

References

Kuan, J., & Mueller, J. (2022). Model-agnostic label quality scoring to detect real-world label errors. *ICML DataPerf Workshop*. doi:10.5281/zenodo.18168246

Examples

```
sim <- simulate_label_noise(n = 300, seed = 1)
scores <- label_quality_scores(sim, truth = given)
# Mislabelled examples get lower scores
tapply(scores, sim$given != sim$true, mean)
```

`noise_matrices` *Noise transition matrices and class priors*

Description

Derives from the estimated joint (see `estimate_joint()`) the quantities used to characterise label noise (Northcutt et al., 2021, Sec. 3.1). Both matrices are oriented with given labels in rows and true labels in columns.

Usage

```
noise_matrices(x, ...)
```

Arguments

`x`	Labels or an object holding labels and predicted probabilities. See Description.
`...`	Passed to `label_inputs()`: for example `pred_probs` when `x` is a label vector, or `truth` when `x` is a data frame.

Value

A list with

- `joint`: the estimated joint distribution `P(given = i, true = j)`.
- `prior_given`: observed class proportions `P(given = i)`.
- `prior_true`: estimated true class proportions `P(true = j)`.
- `noise_matrix`: `P(given = i | true = j)`; columns sum to one. The diagonal is the probability that an example of class `j` keeps its label.
- `inverse_noise_matrix`: `P(true = j | given = i)`; rows sum to one. The diagonal is the probability that a label `i` is correct.

A class with an estimated true prior of zero has `NA` in its column of `noise_matrix`.

References

Northcutt, C., Jiang, L., & Chuang, I. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 1373–1411. doi:10.1613/jair.1.12125

Examples

```
sim <- simulate_label_noise(n = 500, seed = 1)
nm <- noise_matrices(sim, truth = given)
round(nm$inverse_noise_matrix, 2)
```

`num_label_issues` *Estimate the number of label errors*

Description

Two estimators are available:

- `"confident_joint"` (default) counts the examples in the off-diagonal cells of the uncalibrated confident joint, that is examples that are confident members of a class other than their given label, leaving out those whose most likely class is still their given label. This matches `find_label_issues()` with `filter_by = "confident_learning"`.
- `"calibrated"` multiplies the off-diagonal mass of the estimated joint by the number of examples (Northcutt et al., 2021, Sec. 3.2). Calibration assumes that examples not counted in the confident joint are mislabelled at the same rate as those counted.

Usage

```
num_label_issues(x, ..., method = c("confident_joint", "calibrated"))
```

Arguments

`x`	Labels or an object holding labels and predicted probabilities. See Description.
`...`	Passed to `label_inputs()`: for example `pred_probs` when `x` is a label vector, or `truth` when `x` is a data frame.
`method`	Estimator, see Description.

Value

A single integer.

References

Northcutt, C., Jiang, L., & Chuang, I. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 1373–1411. doi:10.1613/jair.1.12125

Examples

```
sim <- simulate_label_noise(n = 500, noise_rate = 0.2, seed = 1)
num_label_issues(sim, truth = given)
sum(sim$given != sim$true)
```

`simulate_label_noise` *Simulate a classification dataset with label errors*

Description

Generates true classes, given labels in which a share of examples have been switched to another class at random, and predicted probabilities from a softmax of noisy logits that favour the true class. Useful for examples, teaching and checking how well the methods recover known errors.

Usage

```
simulate_label_noise(
  n = 1000,
  n_classes = 4,
  noise_rate = 0.1,
  signal = 2,
  prior = NULL,
  seed = NULL
)
```

Arguments

`n`	Number of examples.
`n_classes`	Number of classes, at least two.
`noise_rate`	Probability that an example's given label is switched to a different, uniformly chosen class.
`signal`	How much the logit of the true class is raised. Larger values give a more accurate classifier.
`prior`	Class probabilities of the true classes. Defaults to a moderately imbalanced distribution.
`seed`	Optional random seed. The global random number generator is restored afterwards.

Value

A tibble in the yardstick convention with columns `true`, `given` (factors) and one `.pred_<class>` probability column per class.

Examples

```
sim <- simulate_label_noise(n = 10, seed = 42)
sim
```

Index

class_thresholds, 2
class_thresholds(), *3*, *10*
confident_joint, 3
confident_joint(), *5*

estimate_joint, 4
estimate_joint(), *4, 5*, *11*

find_label_issues, 5
find_label_issues(), *9*, *12*

health_summary, 6
health_summary(), *6*

label_inputs, 8
label_inputs(), *2–5*, *7*, *10–12*
label_quality_scores, 9
label_quality_scores(), *5*

noise_matrices, 11
noise_matrices(), *4*, *7*
num_label_issues, 12
num_label_issues(), *4*, *7*

simulate_label_noise, 13

tibble, *6*