Package ‘tada’

August 4, 2021

Title Tools for impact analysis in randomized trials with text outcomes

Version 0.0.0.9000

Description A flexible and user-friendly toolkit for performing impact analysis in randomized trials with outcomes generated through human, machine, and/or hybrid scoring of text data. Provides functionality for feature extraction and aggregation, applying supervised and unsupervised machine learning models for semi-automated text scoring, estimating model-assisted treatment impacts with respect to text outcomes under various randomized designs, visually representing found impacts on text outcomes, and additional functionality for performing text analysis using existing frameworks, especially quanteda.

Encoding UTF-8

Imports quanteda,
textreg,
sampling,

RdMacros Rdpack

RoxygenNote 7.1.1

R topics documented:

`estimate_impacts` *Estimate treatment impacts for hybrid-scored text outcomes*

Description

Given text from a randomized trial with a binary treatment, where a subset of the documents have been human-scored, this function computes model-assisted estimates for the average treatment effect with respect to the human-coded outcome.

Usage

```
estimate_impacts(
  y.obs,
  yhat,
  Z,
  wts = NULL,
  design = c("crd", "multi", "cluster", "rcbd"),
  siteID = NULL,
  clusterID = NULL,
  data,
  adjust = NULL
)
```

Arguments

`y.obs` A vector of human-coded scores (with NAs for unscored documents).

`yhat` A vector of predicted scores estimated via `predict_scores`.

`Z` Indicator for treatment assignment.

`wts` Sampling weights for which documents were human scored. Assumed uniform if null.

`design` Type of design used for random assignment (complete randomization, multisite randomized, cluster randomized, and blocked and cluster randomized).

`siteID` Vector of IDs for site, for multi-site randomized experiments.

`clusterID` Vector of IDs for cluster, for cluster-randomized experiments.

`data` A `data.frame` of subject-level identifiers, demographic variables, group membership, and/or other pre-treatment covariates.

`adjust` (optional) character vector or named list of variables in the data matrix to adjust for when estimating treatment impacts.

Value

A model object for estimating treatment impact across an array of features.

`extract_features` *Generate an array of text features*

Description

Generates a rich feature representation for documents provided as a character vector or quanteda::corpus() object by applying an array of linguistic and syntactic indices, available text analysis dictionaries, and pre-trained embedding models to all documents.

Usage

```
extract_features(x, p_max = NULL, glove = c(300, 200, 100, 50), ...)
```

Arguments

`x` A corpus object or character vector of text documents.

`p_max` The maximum number of features to compute. Defaults to `NULL` (no strict limit).

`glove` Number of Word2Vec components to compute using GloVe pre-trained embedding model (Pennington et al. 2014). Defaults to 300.

`...` (optional) additional arguments passed to tokens() for text pre-processing.

Value

A matrix of available text features, one row per document, one column per feature.

References

Pennington J, Socher R, Manning C (2014). "Glove: Global vectors for word representation." In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

`extract_taaco` *Manage and merge text features generated using TAACO*

Description

Tools to support feature extraction using TAACO. `prep_taaco()` prepares a corpus for analysis in TAACO. `extract_taaco()` reads output and log files produced by TAACO program and returns a `data.frame` that can be merged with other feature sets.

Usage

```
extract_taaco(file, data = NULL, idvar = NULL)

prep_taaco(x, dir, docnames = NULL)
```

Arguments

`file` Filename where TAACO results are stored

`data` Optional `data.frame` with additional document-level variables to include in output.

`idvar` If `data` is specified, character vector with name(s) of variables used for merging.

`x` A [quanteda::corpus()] object or character vector of text documents.

`dir` Name of directory where TAACO intermediate text files should be stored.

`docnames` Optional character string specifying file names for each document in `x`.

Value

Returns a `data.frame` of text features.

`plot_impacts` *Plot the results from an impact analysis with text outcomes*

Description

This function provides a visualization of the set of textual features found to differ systematically between treatment and control groups.

Usage

```
plot_impacts(x, alpha = 0.05, ...)
```

Arguments

`x`	a model object output from `estimate_impacts()`
`alpha`	the threshold for determining statistical significance
`...`	additional arguments passed to plotting method.

`predict_scores` *Extract predictions from a fitted text scoring model.*

Description

This function computes the predicted scores for a collection of documents based on the results of a trained ensemble learner.

Usage

```
predict_scores(fit, newdata, na.action = na.omit, ...)
```

Arguments

`fit`	a model or list of models to use for prediction
`newdata`	an optional data frame or matrix of predictors
`na.action`	the method for handling missing data
`...`	additional arguments to pass to `predict.train`

Value

A vector of predictions

`prep_external` *Prepare text documents for analysis using external programs*

Description

Text pre-processing and corpus management functions to provide compatibility with external text analysis programs and standalone software packages such as Linguistic Inquiry Word Count (LIWC), the Tool for Automated Analysis of Cohesion (TAACO) and the Sentiment Analysis and Social Cognition Engine (SEANCE).

Usage

```
prep_external(x, dir, docnames = NULL, preProc = NULL)
```

Arguments

`x`	A corpus object or character vector of text documents.
`dir`	Name of directory where TAACO intermediate text files should be stored.
`docnames`	Optional character string specifying file names for each document in `x`.
`preProc`	Optional text pre-processing function(s) (e.g., stemming) to apply prior to writing text files for analysis in external programs.

References

Pennebaker JW, Booth RJ, Boyd RL, Francis ME (2015). “Linguistic Inquiry and Word Count: LIWC 2015.” www.liwc.net. Crossley SA, Kyle K, McNamara DS (2016). “The tool for the automatic analysis of text cohesion (TAACO): Automatic assessment of local, global, and text cohesion.” *Behavior research methods*, **48**(4), 1227–1237. Crossley SA, Kyle K, McNamara DS (2017). “Sentiment Analysis and Social Cognition Engine (SEANCE): An automatic tool for sentiment, social cognition, and social-order analysis.” *Behavior research methods*, **49**(3), 803–821.

`run_ccs` *Perform Concise Comparative Summarization*

Description

Wrapper for textreg::textreg().

Determine the penalty C that will zero out the textreg model for a series of randomly permuted labelings with random assignment dictated by a blocked and cluster-randomized experiment.

Usage

```
run_ccs(x, z, ...)

cluster.threshold.C(
  x,
  z,
  design = c("crd", "multi", "cluster", "rcbd"),
  clusterID,
  siteID = NULL,
  R,
  ...
)
```

Arguments

`x`	a corpus, character vector of text documents, or set of text features.
`z`	an indicator for treatment assignment
`...`	additional arguments passed to textreg().
`design`	Type of design used for random assignment (complete randomization, multisite randomized, cluster randomized, and blocked and cluster randomized).
`clusterID`	vector of cluster ID's
`siteID`	vector of block ID's
`R`	Number of times to scramble treatment assignment labels
`C`	The regularization term. 0 is no regularization.

Details

Method repeatedly generates +1/-1 vectors within the given blocking structure with blocks of +1/-1 within the clustering vector, and then finds a threshold C for each permutation.

Value

a textreg.result() object.

List of numbers. First is the threshold C for the passed labeling. Remainder are the reference distribution based on the permutations.

---

textML *Model-assisted impact analysis through hybrid human/machine text scoring*

Description

A wrapper function for the multiple steps of generating features, training a scoring model on the human-coded data, predicting scores, and comparing human v. machine estimates.

Usage

```
textML(
  x,
  y,
  z = NULL,
  wts = NULL,
  design = c("crd", "multi", "cluster", "rcbd"),
  siteID = NULL,
  clusterID = NULL,
  max.features = NULL,
  ...
)
```

Arguments

`x` a corpus or character vector of text documents.

`y` a vector of human-coded scores. Set elements to ‘NA‘ for documents not previously scored.

`z` optional indicator for treatment assignment. If specified, separate ensembles will be trained for each treatment group;

`wts` Sampling weights for which documents were human scored. Assumed uniform if null.

`design` Type of design used for random assignment (complete randomization, multisite randomized, cluster randomized, and blocked and cluster randomized).

`siteID` Vector of IDs for site, for multi-site randomized experiments.

`clusterID` Vector of IDs for cluster, for cluster-randomized experiments.

`max.features` maximum number of text features to use for model training. Defaults to ‘NULL‘ (no strict limit)

`...` additional arguments passed to train.

Details

This function takes in a corpus of text documents (or a set of computed text features) along with a sample of human-coded outcome values, and trains an ensemble of machine learning models to predict the outcome as a function of the machine measures of text.

Value

a `textML` model object

`textsamp` *Select a random sample of documents*

Description

Functions to select random samples of documents using different sampling schemes and/or along different design criteria.

Usage

```
textsamp(
  x,
  size = length(x),
  prob = NULL,
  wt.fn = NULL,
  scheme = NULL,
  method = c("srswr", "srswor", "systematic", "poisson")
)

textsamp_strata(x, by = NULL, ...)

textsamp_cluster(x, by = NULL, ...)
```

Arguments

`x` A corpus object or character vector of text documents.

`size` a non-negative integer giving the number of documents to sample.

`prob` a vector of probability weights for each document.

`wt.fn` a function for generating probability weights; ignored when `prob` is used. See Details.

`scheme` optional sampling scheme to implement

`method` the following methods are implemented: simple random sampling without replacement (‘srswor‘), simple random sampling with replacement (‘srswr‘), Poisson sampling (‘poisson‘), systematic sampling (‘systematic‘); if `method` is missing, the default method is `srswor`.

`by` a `data.frame` with document-level grouping variable(s) or character vector with names of variables in ‘docvars(x)‘

`...` additional arguments passed on to ‘textsamp‘. Cannot include ‘scheme‘.

Value

Returns a `data.frame` containing identifiers for the selected documents.

`train_ensemble` *Train an ensemble learner for semi-supervised text scoring*

Description

This function takes in a corpus of text documents or a set of computed text features, along with a sample of human-coded outcome values and trains an ensemble of machine learning models to predict the outcome as a function of machine measures of text.

Usage

```
train_ensemble(
  x,
  y,
  z = NULL,
  n.tune = 3,
  cvf = 5,
  bounds = NULL,
  ...,
  return.all = TRUE
)
```

Arguments

`x` a `data.frame` or matrix of numeric text features.

`y` a vector of human-coded scores for the outcome of interest.

`z` optional indicator for treatment assignment. If specified, separate ensembles will be trained for each treatment group;

`n.tune` an integer denoting the amount of granularity in the tuning parameter grid. By default, this argument is the number of levels for each tuning parameters that should be generated by train.

`cvf` number of folds for cross validation

`bounds` a vector (y1, y2) specifying the lower and upper limits for prediction

`...` additional arguments passed to trainControl.

`return.all` should all component models be returned? If ‘FALSE‘, returns only the fitted ensemble(s).

Value

a fitted model object