Package: leadeR (via r-universe)

May 19, 2026

Title Profiling Leaders at a Distance

Version 0.1.0

Description Profiles political leaders at a distance from text data such as speeches, interviews, press conferences, and other public statements. Computes Leadership Trait Analysis scores for seven personality traits -- including need for power, conceptual complexity, and self-confidence -- and classifies leaders into one of eight leadership styles. Also computes Operational Code Analysis scores summarising a leader's beliefs about politics and the use of power.

URL https://github.com/mmukaigawara/leadeR

License MIT + file LICENSE

LazyData true

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Imports spacyr, dplyr, stringr, tidyr, stringi, data.table, tibble, countries, countrycode, purrr

Suggests knitr, metafor, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libicu-dev libpng-dev libssl-dev python3

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

Date/Publication 2026-05-18 18:23:13 UTC

RemoteUrl https://github.com/mmukaigawara/leader

RemoteRef HEAD

RemoteSha ca84c59a6179a5e825c9aa1e7f7eb88f4786b1fb

Contents

`clean_text` *Clean text for analysis*

Description

Removes editorial annotations in square brackets, parentheses, and curly braces from political speech transcripts. Also normalizes whitespace and collapses double em-dashes left behind by removals.

Usage

```
clean_text(x)
```

Arguments

`x`	A character vector of text to clean.

Value

A character vector with annotations removed and whitespace normalized.

Examples

```
clean_text("We must act now [applause] for the future.")
clean_text("The president (speaking loudly) left the room.")
```

get_aff *Compute affiliation scores*

Description

Measures the affiliation orientation of a leader's speech by classifying verb-level actions as affiliative (A) or other-affiliative (OA) based on the speaker's own-entity references and sentence-level context.

Usage

```
get_aff(own_entity, text, bootstrap = FALSE, B = 1000)
```

Arguments

own_entity	A character vector of entity names representing the speaker's own country or group.
text	A character string containing the speech text to analyse.
bootstrap	Logical; if TRUE, return bootstrapped mean and variance estimates. Default is FALSE.
B	Integer; number of bootstrap replicates. Default is 1000.

Value

A one-row tibble. When bootstrap = FALSE, columns are A and OA. When bootstrap = TRUE, columns are meanA, meanOA, varA, varOA.

get_complex *Compute conceptual complexity scores*

Description

Counts high-complexity (HC) and low-complexity (LC) markers in the text, using word stems, phrases, and exact words with negation-aware counting.

Usage

```
get_complex(
  text,
  bootstrap = FALSE,
  B = 1000,
  quote_strip = TRUE,
  window_chars = 40
)
```

Arguments

text	A character string containing the speech text to analyse.
bootstrap	Logical; if TRUE, return bootstrapped mean and variance estimates. Default is FALSE.
B	Integer; number of bootstrap replicates. Default is 1000.
quote_strip	Logical; if TRUE, remove quoted text before counting. Default is TRUE.
window_chars	Integer; number of characters to look back for negation context. Default is 40.

Value

A one-row tibble. When bootstrap = FALSE, columns are HC and LC. When bootstrap = TRUE, columns are meanHC, varHC, meanLC, varLC.

---

get_conf *Compute self-confidence scores*

Description

Classifies first-person pronoun occurrences in the text as self-confident (SC) or other-self-confident (OSC) based on sentence-level context patterns covering three conditions: self as instigator, self as authority, and self as recipient of positive recognition.

Usage

```
get_conf(text, bootstrap = FALSE, B = 1000)
```

Arguments

text	A character string containing the speech text to analyse.
bootstrap	Logical; if TRUE, return bootstrapped mean and variance estimates. Default is FALSE.
B	Integer; number of bootstrap replicates. Default is 1000.

Value

A one-row tibble. When bootstrap = FALSE, columns are SC and OSC. When bootstrap = TRUE, columns are meanSC, meanOSC, varSC, varOSC.

get_ctrl *Compute control (need for power) scores*

Description

Classifies verbs associated with first-person subjects as instrumental-control (IC) or other-control (OC) using dependency parsing to link subjects to their governing verbs.

Usage

```
get_ctrl(own_entity, text, bootstrap = FALSE, B = 1000)
```

Arguments

own_entity	A character vector of entity names representing the speaker's own country or group.
text	A character string containing the speech text to analyse.
bootstrap	Logical; if TRUE, return bootstrapped mean and variance estimates. Default is FALSE.
B	Integer; number of bootstrap replicates. Default is 1000.

Value

A one-row tibble. When bootstrap = FALSE, columns are IC and OC. When bootstrap = TRUE, columns are meanIC, meanOC, varIC, varOC.

get_dist *Compute distrust scores*

Description

Performs per-entity classification of references in the text as suspicious (S) or other-suspicious (OS) based on distrust verbs, harmful nominals, and contextual modifiers.

Usage

```
get_dist(own_entity, text, bootstrap = FALSE, B = 1000)
```

Arguments

own_entity	A character vector of entity names representing the speaker's own country or group.
text	A character string containing the speech text to analyse.
bootstrap	Logical; if TRUE, return bootstrapped mean and variance estimates. Default is FALSE.
B	Integer; number of bootstrap replicates. Default is 1000.

Value

A one-row tibble. When bootstrap = FALSE, columns are S and OS. When bootstrap = TRUE, columns are meanS, varS, meanOS, varOS.

---

get_lta *Leadership Trait Analysis (LTA)*

Description

Runs all eight LTA trait functions at once and returns a single combined tibble. The eight traits are: power, affiliation, distrust, conceptual complexity, task orientation, self-confidence, nationalism, and control.

Usage

```
get_lta(own_entity, text, bootstrap = FALSE, B = 1000)
```

Arguments

own_entity	A character string identifying the speaker's country or entity (e.g., "United States").
text	A character string containing the speech text to analyse.
bootstrap	Logical. If TRUE, returns bootstrap means and variances instead of raw counts. Default is FALSE.
B	Number of bootstrap iterations. Default is 1000.

Value

A one-row tibble::tibble.

When bootstrap = FALSE, columns include raw counts (P, OP, A, OA, S, OS, HC, LC, TI, IP, SC, OSC, N, ON, IC, OC) plus trait proportions (Pp, D, C, Ta, Ss, Na, B).

When bootstrap = TRUE, columns include bootstrap means and variances for the raw counts (meanP, varP, meanOP, varOP, ...) plus trait proportions and their delta-method variances (Pp, varPp, D, varD, C, varC, Ta, varTa, Ss, varSs, Na, varNa, B, varB).

Examples

```
# Requires spaCy to be installed; see spacyr::spacy_install().
spacyr::spacy_initialize()
get_lta(own_entity = "United States", text = "We will defend our nation.")
get_lta(own_entity = "United States", text = "We will defend our nation.",
        bootstrap = TRUE, B = 500)
```

get_nat *Compute nationalism scores*

Description

Classifies entity references in the text as nationalistic (N) or other-nationalistic (ON) based on own/other entity detection and contextual modifiers at the referent level.

Usage

```
get_nat(own_entity, text, bootstrap = FALSE, B = 1000)
```

Arguments

own_entity	A character vector of entity names representing the speaker's own country or group.
text	A character string containing the speech text to analyse.
bootstrap	Logical; if TRUE, return bootstrapped mean and variance estimates. Default is FALSE.
B	Integer; number of bootstrap replicates. Default is 1000.

Value

A one-row tibble. When bootstrap = FALSE, columns are N and ON. When bootstrap = TRUE, columns are meanN, meanON, varN, varON.

get_oca *Compute operational code (VICS) scores*

Description

Performs full VICS (Verbs in Context System) operational code analysis on a speech text, classifying verb-entity pairs as Self/Other and scoring them on a -3 to +3 scale (Punish to Reward). Returns P1–P5 and I1–I5 indices.

Usage

```
get_oca(own_entity, text, bootstrap = FALSE, B = 1000)
```

Arguments

own_entity	A character vector of entity names representing the speaker's own country or group.
text	A character string containing the speech text to analyse.
bootstrap	Logical; if TRUE, return bootstrapped mean and variance estimates for all indices. Default is FALSE.
B	Integer; number of bootstrap replicates. Default is 1000.

Value

A one-row tibble with P1–P5, I1–I5, and raw Self/Other counts by score category.

`get_power` *Compute Power Score (LTA)*

Description

Classifies verbs in text as reflecting Power (P) or Other Power (OP) based on the Leadership Trait Analysis codebook.

Usage

```
get_power(own_entity, text, bootstrap = FALSE, B = 1000)
```

Arguments

`own_entity`	Character vector of the speaker's country/entity name(s).
`text`	Character string of the speech text to analyse.
`bootstrap`	Logical; if TRUE, return bootstrap means and variances.
`B`	Number of bootstrap replications (default 1000).

Value

A tibble with columns P and OP (or meanP, meanOP, varP, varOP if bootstrap = TRUE).

`get_task` *Compute task orientation scores*

Description

Classifies lemmas in the parsed text as task-instrumental (TI) or interpersonal (IP) based on codebook word lists, with token-level quote and negation handling.

Usage

```
get_task(text, bootstrap = FALSE, B = 1000)
```

Arguments

`text`	A character string containing the speech text to analyse.
`bootstrap`	Logical; if `TRUE`, return bootstrapped mean and variance estimates. Default is `FALSE`.
`B`	Integer; number of bootstrap replicates. Default is 1000.

Value

A one-row tibble. When bootstrap = FALSE, columns are TI and IP. When bootstrap = TRUE, columns are meanTI, meanIP, varTI, varIP.

---

jfk19610120 *JFK Speech: Inaugural Address (January 20, 1961)*

Description

Full text of JFK's Inaugural Address on January 20, 1961.

Usage

```
jfk19610120
```

Format

A character string containing the full text of the speech.

Source

https://www.govinfo.gov/app/details/PPP-1961-book1. These original source texts are U.S. government works and are not subject to copyright protection in the United States under 17 U.S.C. § 105. The package authors do not claim copyright in the original presidential speech texts.

---

jfk19610925 *JFK Speech: Address Before the UN General Assembly (September 25, 1961)*

Description

Full text of JFK's address before the UN General Assembly on September 25, 1961.

Usage

```
jfk19610925
```

Format

A character string containing the full text of the speech.

Source

https://www.govinfo.gov/app/details/PPP-1961-book1. These original source texts are U.S. government works and are not subject to copyright protection in the United States under 17 U.S.C. § 105. The package authors do not claim copyright in the original presidential speech texts.

jfk19630610 *JFK Speech: Commencement Address at American University (June 10, 1963)*

Description

Full text of Senator John F. Kennedy's commencement address at American University on June 10, 1963.

Usage

```
jfk19630610
```

Format

A character string containing the full text of the speech.

Source

https://www.govinfo.gov/app/details/PPP-1963-book1. These original source texts are U.S. government works and are not subject to copyright protection in the United States under 17 U.S.C. § 105. The package authors do not claim copyright in the original presidential speech texts.

type_lta *Classify LTA Traits into Hermann's Leadership Typology*

Description

Takes per-speech LTA output (from `get_lta()`) and aggregates trait scores across speeches, then classifies the leader along three dimensions (constraint, openness, motivation toward world) and maps the first two plus task orientation to one of eight leadership styles.

Usage

```
type_lta(
  lta,
  precision_weighted = FALSE,
  need_for_power = 0.5,
  control = 0.44,
  complex_high = 0.56,
  confidence_high = 0.81,
  task = 0.59,
  distrust = 0.41,
  ingroup = 0.42
)
```

Arguments

lta	A data frame with one row per speech, as returned by get_lta(). Must contain columns Pp, B, C, Ss, Ta, D, Na. When precision_weighted = TRUE, must also contain varPp, varB, varC, varSs, varTa, varD, varNa.
precision_weighted	Logical. If FALSE (default), aggregate traits with a simple mean. If TRUE, use inverse-variance (precision) weighting via random-effects meta-analysis (metafor::rma()).
need_for_power	Threshold for the need-for-power trait (Pp). Default 0.50.
control	Threshold for belief in ability to control events (B). Default 0.44.
complex_high	Threshold for high conceptual complexity (C). Default 0.56.
confidence_high	Threshold for high self-confidence (Ss). Default 0.81.
task	Threshold for task orientation (Ta). Default 0.59.
distrust	Threshold for distrust (D). Default 0.41.
ingroup	Threshold for in-group bias / nationalism (Na). Default 0.42.

Value

A one-row tibble::tibble with aggregated trait values, standard errors (when precision_weighted = TRUE), and classification columns: constraint, openness, motivation_toward_world, task_orientation, typology, and method.

Examples

```
# Simple-mean aggregation works on any data frame with the seven trait
# columns; here we use a small illustrative input.
example_lta <- data.frame(
  Pp = c(0.45, 0.52, 0.48),
  B  = c(0.40, 0.48, 0.43),
  C  = c(0.60, 0.55, 0.58),
  Ss = c(0.75, 0.82, 0.79),
  Ta = c(0.62, 0.58, 0.65),
  D  = c(0.35, 0.42, 0.38),
  Na = c(0.38, 0.45, 0.41)
)
type_lta(example_lta)


# Precision-weighted aggregation needs per-trait variances, which come
# from get_lta(..., bootstrap = TRUE). Requires spaCy to be installed.
spacyr::spacy_initialize()
res <- data.table::rbindlist(
  lapply(c(jfk19610120, jfk19610925, jfk19630610), function(x)
    get_lta(own_entity = "United States", text = clean_text(x),
            bootstrap = TRUE, B = 1000))
)
type_lta(res, precision_weighted = TRUE)
```

Index