Package: rdomains (via r-universe)

October 29, 2026

Title Get the Category of Content Hosted by a Domain

Version 0.6.1

Description Fetch website content and classify it through the Python 'piedomains' text model. Look up domains in 'UT-Capitole', Steven Black, university, archived 'Shallalist' and 'DMOZ' lists, or use 'VirusTotal' <https://www.virustotal.com/>, 'OpenAI' and 'Anthropic' APIs. Report collection outcomes and source provenance. Includes a historical domain-name classifier trained on 'Shallalist' data.

URL https://themains.github.io/rdomains/, https://github.com/themains/rdomains

BugReports https://github.com/themains/rdomains/issues

Depends R (>= 4.5.0)

Imports Matrix, urltools, glmnet, stats, methods, httr2, xml2, curl, virustotal (>= 0.7.0), jsonlite, R.utils, dplyr (>= 1.1.0), purrr (>= 1.0.0), tibble (>= 3.2.0), stringr (>= 1.5.0), rlang (>= 1.1.0), cli (>= 3.6.0), checkmate (>= 2.3.0), glue (>= 1.6.0), readr (>= 2.1.0), spiderbar, ipaddress, withr

Suggests testthat, rmarkdown, knitr (>= 1.11), webfakes, reticulate (>= 1.41.0)

VignetteBuilder knitr

License MIT + file LICENSE

Encoding UTF-8

Config/roxygen2/version 8.0.0

Config/testthat/edition 3

NeedsCompilation no

Author Gaurav Sood [aut, cre]

Maintainer Gaurav Sood <gsood07@gmail.com>

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

Date/Publication 2026-10-29 08:50:11 UTC

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

RemoteRef HEAD

RemoteSha edd931e44fe306ea321882798db2f608790628e7

Contents

rdomains-package . . . 2
adult_name_probability . . . 3
cache_clear . . . 4
classification_results . . . 4
claude_cat . . . 5
collect_content . . . 6
dmoz_cat . . . 8
fetch_error_codes . . . 9
fetch_report . . . 10
get_dmoz_data . . . 10
get_shalla_data . . . 11
get_stevenblack_data . . . 12
get_ut1_data . . . 12
glm_shalla . . . 13
html_text_content . . . 14
news_url_signals . . . 15
openai_cat . . . 16
page_signals . . . 17
pie_cat . . . 18
rdomains_cache_dir . . . 19
shalla_cat . . . 19
source_vintage . . . 20
stevenblack_cat . . . 21
uni_cat . . . 22
ut1_cat . . . 23
virustotal_cat . . . 24
web_categories . . . 25

Index **26**

`rdomains-package` *rdomains: Classify Domains by their Content*

Description

Want to know what kind of content is carried on a domain? Get the results quickly using rdomains. The package provides access to virustotal API, UT-Capitole, archived lists, OpenAI models, Anthropic Claude models, and a historical model trained on Shallalist data. Use `pie_cat()` to classify fetched content through the Python piedomains text model.

Details

To learn how to use rdomains, see this vignette: ../doc/rdomains.html.

Author(s)

Gaurav Sood

See Also

Useful links:

- https://themains.github.io/rdomains/
- https://github.com/themains/rdomains
- Report bugs at https://github.com/themains/rdomains/issues

---

adult_name_probability

Probability that Domain Hosts Adult Content Based on features of Domain Name and Suffix alone.

Description

Uses a historical ML model that uses keywords in the domain name and suffix to predict probability that the domain hosts adult content. For more information see https://github.com/themains/keyword_porn

Usage

```
adult_name_probability(domains)
```

Arguments

domains	required; string; vector of domain names

Details

The bundled model artifact has not changed since November 2017. Its probabilities have not been validated on current domains. It uses only the name and suffix, not page content; use pie_cat() for content-based classification. Feature encoding remains the training-era encoding rather than silently changing a fitted model.

Value

A tibble with input identity, p_adult, model identity and artifact date.

Examples

```
## Not run:
adult_name_probability("http://www.google.com")

## End(Not run)
```

`cache_clear` *Empty the page cache*

Description

Empty the page cache

Usage

```
cache_clear(dir = NULL)
```

Arguments

`dir`	Cache directory. Defaults to the session cache used when `collect_content()` is called without `cache_dir`.

Value

Invisibly, the number of cached pages removed.

Examples

```
## Not run:
cache_clear()
cache_clear(rdomains_cache_dir())

## End(Not run)
```

`classification_results`
Classification result contract

Description

Classifiers preserve input order and return one row per input. `input_id` is the one-based position in that call; retain it when saving and resuming a batch. `status` is `ok`, `no_match` or `failed`. An absent lookup membership is not a failed request. `category` is populated only when there is one selected label; `labels` preserves every label in a tibble with `category`, `score` and `vendor`. Unknown scores are `NA`, not probabilities inferred from list membership.

Details

`provider` identifies the classifier or list; `evidence_type` distinguishes page content, archived content, name inference and list membership. `details` retains source-specific records, including the collected page for piedomains results. Model and source identifiers are reported where known; missing dates stay missing.

Value

A tibble with input identity, provider, evidence, category, confidence, labels, status, stage, error_code, retryable, http_status, model_id, model_revision, source_last_published, source_fetched_at and details.

`claude_cat` *Get Category from Anthropic Claude*

Description

Infers a category from the domain name using Anthropic's Claude API. No page content is fetched. The answer reflects the provider's knowledge and the domain name, not a verified observation of the current website.

Usage

```
claude_cat(
  domains,
  api_key = NULL,
  categories = NULL,
  model = "claude-haiku-5-5",
  delay = 0.5
)
```

Arguments

`domains`	vector of domain names
`api_key`	Anthropic API key. If not provided, looks for the ANTHROPIC_API_KEY or CLAUDE_API_KEY environment variable
`categories`	vector of categories to classify into. If NULL, uses default web categories
`model`	Claude model to use (default: "claude-haiku-5-5" for cost efficiency)
`delay`	delay in seconds between API calls (default: 0.5)

Value

A tibble following classification_results, including provider failures and model identity.

Examples

```
## Not run:
claude_cat("google.com")
claude_cat(c("google.com", "facebook.com"))
claude_cat("google.com", categories = c("search", "social", "ecommerce", "news", "other"))

## End(Not run)
```

`collect_content` *Fetch homepage HTML and text for domains*

Description

Every requested domain comes back, in order, whether or not it was reachable. Failures carry a code from `fetch_error_codes()` rather than a bare `NA`.

Usage

```
collect_content(
  domains = NULL,
  delay = 1,
  timeout = 10,
  max_bytes = 10 * 1024^2,
  obey_robots = TRUE,
  max_crawl_delay = 30,
  max_redirects = 5,
  cache_dir = NULL,
  cache_ttl = 7 * 86400,
  cache_max_size = 1024^3,
  allow_hosts = character(),
  archive_fallback = FALSE,
  archive_date = NULL,
  user_agent = rdomains_user_agent()
)
```

Arguments

`domains`	Character vector of domains or URLs.
`delay`	Minimum seconds between requests to the same host. `Crawl-delay` overrides this upward.
`timeout`	Per-request timeout, seconds.
`max_bytes`	Cap on the response body actually read. The default matches piedomains' 10 MB. A smaller cap looks prudent and is not: cnn.com's homepage alone is roughly 6 MB, so a 2 MB limit rejects major news sites as `content_too_large` while they return HTTP 200.

`obey_robots` Whether to fetch and honour robots.txt. Turning this off is discouraged and is your responsibility, not the package's.

`max_crawl_delay` Skip a host that asks for a longer delay than this rather than sleeping on it.

`max_redirects` Maximum redirect hops to follow. Every hop is re-validated, so a redirect cannot be used to reach an address the first check refused.

`cache_dir` Where to cache fetched pages. Defaults to a directory under `tempdir()`, so nothing persists past the session; pass `rdomains_cache_dir()` to opt into a cache that does.

`cache_ttl` Seconds a cached page stays fresh.

`cache_max_size` Prune the cache above this many bytes, oldest first.

`allow_hosts` Hosts exempt from the private-address checks, by exact name. Only the hosts named are exempt, so a redirect elsewhere is still refused. Intended for testing against a local server.

`archive_fallback` Opt in to archived content (default `FALSE`). When a host serves an anti-bot interstitial, try the Internet Archive's most recent capture. Rows recovered this way carry `source = "archive"` and a `snapshot_timestamp`, so the vintage is never hidden. Dead domains and explicitly exempted hosts are not recovered this way.

`archive_date` Fetch each domain as it was on this date (`"YYYYMMDD"`) from the Internet Archive instead of fetching it live. This is how you ask what a domain *used to be* – and, set against a live run, how you measure whether a label has gone stale. Rows carry `source = "archive"` and the realised `snapshot_timestamp`, which is the latest capture on or before the requested day.

`user_agent` Override the identifying user-agent.

Details

The crawler identifies itself as `rdomains/<version>`, obeys `robots.txt` including `Crawl-delay`, spaces requests to the same host, caps the response body, and refuses to fetch hosts that resolve to private or link-local addresses.

Value

A tibble with one row per input: `input_id`, `input`, `domain_name`, `archive_error_code`, `status`, `stage`, `error_code`, `retryable`, `http_status`, `final_url`, `fetched_at`, `content_bytes`, `title`, `description`, `lang`, `text`, `html`, `n_tokens`, `page_state`, `block_vendor`, `robots_allowed`, `from_cache`, `source`, `snapshot_timestamp`, `source_last_published`. Live pages have no inferred publication date; `fetched_at` records retrieval.

See Also

`fetch_report()` to summarise the run, `page_signals()` for what the page states are, `source_vintage()` for how a live fetch compares with the static lists.

Examples

```
## Not run:
res <- collect_content(c("example.com", "wikipedia.org"))
fetch_report(res)

# retry only what is worth retrying
again <- collect_content(res$input[res$retryable])

## End(Not run)
```

`dmoz_cat` *Get Category from DMOZ*

Description

Fetches category (or categories) of content hosted by a domain according to DMOZ. The function checks if path to the DMOZ file is provided by the user. If not, it looks for `dmoz_domain_category.csv` in the working directory. It also returns results for prominent subdomains.

Usage

```
dmoz_cat(domains, use_file = NULL)
```

Arguments

`domains`	vector of domain names
`use_file`	path to the dmoz file, which can be downloaded using `get_dmoz_data`

Details

DMOZ closed in March 2017 and the snapshot shipped here is from 2015, so these labels describe the domain as it was a decade ago. The returned `source_last_published` column carries that vintage when the file has verified download metadata. A warning is issued once per session. See `source_vintage`.

Publication dates come from the `.metadata.json` sidecar written by the downloader, checked against the file contents. `source_fetched_at` is the download time. Files without matching metadata return `NA` dates; filesystem timestamps are not used.

Value

A tibble following classification_results. Source-specific data are retained in `details`; all source labels are retained in `labels`.

See Also

`source_vintage` for the provenance of every category source

Examples

```
## Not run:
dmoz_cat(domains = "http://www.google.com")
dmoz_cat(domains = c("http://www.google.com", "http://plus.google.com"))

## End(Not run)
```

`fetch_error_codes` *Reasons a domain can fail to get a category*

Description

The closed set of `error_code` values that `collect_content()` and `pie_cat()` can return, with the pipeline stage each belongs to and whether it is worth retrying.

Usage

```
fetch_error_codes()
```

Details

Shared strings retain their meaning in `piedomains`. R-specific acquisition failures and retry-policy exceptions are recorded in the versioned contract.

Value

A tibble with columns `code`, `stage`, `retryable` and `description`.

See Also

`fetch_report()` to summarise an actual run, `source_vintage()` for the provenance of the static label sources.

Examples

```
fetch_error_codes()

# which failures are worth trying again?
subset(fetch_error_codes(), retryable)
```

fetch_report *Summarise the outcome of a fetch or classification run*

Description

A pure function over the returned tibble, deliberately **not** a stored summary. A snapshot goes stale the moment you filter the rows; recomputing means `fetch_report(subset(x, retryable))` is always correct.

Usage

```
fetch_report(x)
```

Arguments

x	A tibble from `collect_content()` or `pie_cat()`.

Value

A one-row tibble: `n`, `ok`, `failed`, `no_match`, `retryable`, and list-columns `by_stage`, `by_reason` and `missing` (the domains that produced nothing).

Examples

```
rows <- data.frame(
  domain_name = c("a.com", "b.com", "c.com"),
  status = c("ok", "failed", "failed"),
  stage = c("infer", "fetch", "process"),
  error_code = c(NA, "timeout", "thin_content"),
  retryable = c(FALSE, TRUE, FALSE)
)
fetch_report(rows)
```

get_dmoz_data *Get DMOZ Data*

Description

Downloads archived DMOZ (Open Directory Project) data. DMOZ was discontinued in March 2017. This function downloads our preserved 2015 DMOZ snapshot. For more details, check: https://github.com/themains/rdomains/tree/master/data-raw/dmoz/

Usage

```
get_dmoz_data(outdir = ".", overwrite = FALSE)
```

Arguments

outdir	Optional; folder to which you want to save the file; Default is same folder
overwrite	Optional; default is FALSE. If TRUE, the file is overwritten.

Value

Invisibly, the absolute path to the downloaded file.

References

https://archive.org/details/dmoz-rdf-20150327

Examples

```
## Not run:
get_dmoz_data()

## End(Not run)
```

---

get_shalla_data *Get Shalla Data*

Description

Shallalist service was discontinued in January 2022. This function downloads the last archived copy (from 1/14/22) that we have preserved on GitHub. The original service at shallalist.de is no longer available. Downloads, unzips and saves the final version of shallalist data. By default, saves shalla data as `shalla_domain_category.csv`.

Usage

```
get_shalla_data(outdir = ".", overwrite = FALSE)
```

Arguments

outdir	Optional; folder to which you want to save the file; Default is same folder
overwrite	Optional; default is FALSE. If TRUE, the file is overwritten.

References

https://web.archive.org/web/20210502020725/http://www.shallalist.de/

Examples

```
## Not run:
get_shalla_data()

## End(Not run)
```

get_stevenblack_data *Get Steven Black's Host List Data*

Description

Downloads the latest version of Steven Black's unified hosts file. The hosts file contains domains known for serving ads, malware, and tracking.

Usage

```
get_stevenblack_data(outdir = ".", variant = "base", overwrite = FALSE)
```

Arguments

outdir	Optional; folder to save the file in. Default is the current directory
variant	Optional; which variant to download. Options: "base", "porn", "social", "gambling", "all"
overwrite	Optional; default is FALSE. If TRUE, the file is overwritten.

Value

Invisibly, the absolute path to the downloaded file.

References

https://github.com/StevenBlack/hosts

Examples

```
## Not run:
get_stevenblack_data()
get_stevenblack_data(variant = "all")

## End(Not run)
```

get_ut1_data *Download the UT-Capitole blacklists*

Description

Downloads and flattens the University of Toulouse (UT1) blacklists into a single `domain,category,usage` CSV. These are the **maintained successor** to Shallalist: where Shallalist last published in January 2022, UT1 is updated continuously.

Usage

```
get_ut1_data(outdir = "./", overwrite = FALSE)
```

Arguments

`outdir`	Optional; folder to save the file in. Default is the working directory.
`overwrite`	Optional; default `FALSE`. If `TRUE`, an existing file is replaced.

Details

The archive carries ~69 categories under its own French-influenced vocabulary (`drogue`, `publicite`, `tricheur`), which does not line up with Shallalist's names. `ut1_cat()` therefore reports UT1's category verbatim rather than mapping it onto another taxonomy and losing the distinction.

A domain may appear in more than one category. Every membership is retained; duplicate identical records are removed and rows are sorted by domain and category.

Value

Invisibly, the path to the written CSV.

References

https://dsi.ut-capitole.fr/blacklists/download/blacklists.tar.gz

See Also

`ut1_cat()`, and `source_vintage()` for how this compares with the discontinued sources.

Examples

```
## Not run:
get_ut1_data()
ut1_cat("example.com")

## End(Not run)
```

---

glm_shalla *ML Model*

Description

ML Model

Usage

```
glm_shalla
```

Format

A list

Author(s)

Gaurav Sood

Source

ML model based on shallalist using keywords and domain suffixes,

---

`html_text_content` *Extract text, title, description and language from HTML*

Description

Extract text, title, description and language from HTML

Usage

```
html_text_content(html)
```

Arguments

`html`	Raw HTML as a single string.

Value

A list with `text` (cleaned, lowercased), `title`, `description` and `lang`. Missing elements are `NA_character_`; `text` is `""` when nothing could be extracted.

See Also

`page_signals()`, which uses this to decide whether a page is classifiable.

Examples

```
html <- "<html lang='en'><head><title>Example</title>
  <meta name='description' content='A demo page'>
  <script>ignored()</script></head>
  <body><p>Hello   World</p><style>p{}</style></body></html>"
html_text_content(html)
```

`news_url_signals` *Detect soft- and hard-news keywords in URLs*

Description

Based on a slightly amended version of the regular expression used to classify news, and non-news in: "Exposure to ideologically diverse news and opinion on Facebook" by Bakshy, Messing, and Adamic. Science. 2015.

Usage

```
news_url_signals(urls)
```

Arguments

`urls`	vector of URLs

Details

Amendment: sport rather than sports

URL containing any of the following words is classified as soft news: "sport|entertainment|arts|fashion|style|lifestyle|leisure|ce
nolint: line_length_linter.

URL containing any of following words is classified as hard news: "politi|usnews|world|national|state|elect|vote|govern|campa
nolint: line_length_linter.

Note that it is based on patterns existing in a small set of domains. See paper for details.

Value

A tibble with `input_id`, `input`, `soft_news` and `hard_news`. The two flags are independent; neither is a general news/non-news decision.

References

https://www.science.org/doi/10.1126/science.aaa1160

Examples

```
## Not run:
news_url_signals("http://www.bbc.com/sport")
news_url_signals(c("http://www.bbc.com/sport", "http://www.washingtontimes.com/news/politics/"))

## End(Not run)
```

openai_cat *Get Category from OpenAI*

Description

Infers a category from the domain name using OpenAI's chat completion API. No page content is fetched. The answer reflects the provider's knowledge and the domain name, not a verified observation of the current website.

Usage

```
openai_cat(
  domains,
  api_key = NULL,
  categories = NULL,
  model = "gpt-6-luna",
  delay = 0.5
)
```

Arguments

domains	vector of domain names
api_key	OpenAI API key. If not provided, looks for OPENAI_API_KEY environment variable
categories	vector of categories to classify into. If NULL, uses default web categories
model	OpenAI model to use (default: "gpt-6-luna" for cost efficiency)
delay	delay in seconds between API calls (default: 0.5)

Value

A tibble following classification_results, including provider failures and model identity.

Examples

```
## Not run:
openai_cat("google.com")
openai_cat(c("google.com", "facebook.com"))
openai_cat("google.com", categories = c("search", "social", "ecommerce", "news", "other"))

## End(Not run)
```

page_signals *What kind of page is this?*

Description

Inspects already-fetched HTML for the three things that are not a classifiable site: an anti-bot interstitial, a domain-parking placeholder, and a server's "nothing here" page. No network access.

Usage

```
page_signals(html, text = NULL, domain = "", status = NULL)
```

Arguments

html	Raw response body.
text	Extracted page text. If NULL, derived from html.
domain	The domain requested, used to spot a page whose only content is its own name.
status	HTTP status code, if known.

Details

parked and unavailable are **answers, not failures** – they are facts about the domain, plainly stated in the page, that a caller can act on. They are also free: no classification service needs to be consulted.

Value

A one-row tibble: page_state (one of "content", "blocked", "parked", "unavailable", "thin"), blocked, block_vendor, block_reason, parked, unavailable, thin, n_tokens.

See Also

fetch_error_codes() for how these map onto run outcomes.

Examples

```
# A Cloudflare challenge, not a website
page_signals("<html><title>Just a moment...</title><body>cf_chl_opt</body></html>")

# A parking page
page_signals("<html><body>This domain is for sale. Inquire now.</body></html>")

# A real page that merely embeds reCAPTCHA is not blocked
page_signals(paste0(
  "<html><title>Reddit</title><body>",
  paste(rep("real discussion content", 200), collapse = " "),
  "<script src='recaptcha/api.js'></script></body></html>"
))
```

pie_cat *Classify fetched HTML with piedomains*

Description

Fetches pages with `collect_content()` and passes their original HTML to piedomains for extraction, tokenization, calibration and classification. The Python package must be version 0.15.0 or newer. This uses its text classifier, not its image or fusion models.

Usage

```
pie_cat(domains = NULL, pages = NULL, threshold = 0.1, model = NULL, ...)
```

Arguments

`domains`	Character vector of domains or URLs. Supply either `domains` or `pages`.
`pages`	Data frame from `collect_content()`, including its retained `html` column, or previous piedomains results. Previously extracted text alone is insufficient to reproduce model preprocessing. Failed fetches are retained. Failed inference can be retried on the same pages.
`threshold`	Finite probability in `[0,1]`. Include every category at or above this value from the full model distribution, always retaining the predicted category.
`model`	Optional Hugging Face model ID or local checkpoint directory. Overrides `PIEDOMAINS_TEXT_MODEL` for this call. The pinned default model is reused within a session.
`...`	Passed to `collect_content()` when `pages` is absent.

Details

A current fetch does not guarantee a correct current label. The shipped checkpoint has known retail/drugs confusion; see its model card. Archive and cached content retain their collection dates. `details` preserves the page and its `fetch_*` collection outcomes separately from the terminal classification `status`, `stage`, `error_code` and `retryable` columns.

Value

A tibble following classification_results. Source-specific data are retained in `details`; all source labels are retained in `labels`.

Setup

Install `reticulate` and declare Python requirements before initializing Python: `reticulate::py_require("piedomains==`
`python_version = ">=3.12")`. Set `RETICULATE_PYTHON` to select an existing Python environment. Model weights download from Hugging Face on first use. The Python package pins its default checkpoint; `PIEDOMAINS_TEXT_MODEL` can select another checkpoint or local path. Results report the resolved model, revision when available, Python package version, extractor, cleaning mode and calibration temperature.

Examples

```
## Not run:
pages <- collect_content(c("cnn.com", "wikipedia.org"))
pie_cat(pages = pages)

## End(Not run)
```

---

rdomains_cache_dir *A persistent cache directory for fetched pages*

Description

Returns the conventional per-user cache location for the package. It deliberately does **not** create the directory: pass it to `collect_content()` to opt in, and until you do nothing is written outside the session's temporary directory.

Usage

```
rdomains_cache_dir()
```

Value

Path to the per-user cache directory, as a string.

Examples

```
rdomains_cache_dir()
## Not run:
# opt in to caching that survives the session
collect_content("example.com", cache_dir = rdomains_cache_dir())

## End(Not run)
```

---

shalla_cat *Get Category from Shallalist*

Description

Fetches category of content hosted by a domain according to Shalla. The function checks if path to the shalla file is provided by the user. If not, it looks for `shalla_domain_category.csv` in the working directory.

Usage

```
shalla_cat(domains, use_file = NULL)
```

Arguments

domains	vector of domain names
use_file	path to the latest shallalist file downloaded using get_shalla_data

Details

Shallalist stopped publishing in 2022, so these labels describe the domain as it was then or earlier, not as it is now. The returned source_last_published column carries that vintage when the file has verified download metadata. A warning is issued once per session. See source_vintage.

Publication dates come from the .metadata.json sidecar written by the downloader, checked against the file contents. source_fetched_at is the download time. Files without matching metadata return NA dates; filesystem timestamps are not used.

Value

A tibble following classification_results. Source-specific data are retained in details; all source labels are retained in labels.

See Also

source_vintage for the provenance of every category source

Examples

```
## Not run:
shalla_cat(domains = "http://www.google.com")

## End(Not run)
```

---

source_vintage *Report what each category source is and when it was last published*

Description

The sources listed here answer from static lists, and those lists have very different vintages. Two of them are no longer published at all. Use this to see publication dates and maintenance status; freshness alone does not establish accuracy.

Usage

```
source_vintage(source = NULL)
```

Arguments

source	Optional; one of "shalla", "dmoz", "stevenblack", "ut1", "uni". If NULL, every source is returned.

Value

A tibble with one row per source: `key`, `source`, `last_published`, `status`, `successor` and `note`.

Examples

```
source_vintage()
source_vintage("dmoz")
```

`stevenblack_cat` *Get Category from Steven Black's Host List*

Description

Classifies domains based on Steven Black's unified host list which blocks ads, malware, and tracking domains. The function checks if a domain appears in the blocklist and categorizes it accordingly.

Usage

```
stevenblack_cat(domains, use_file = NULL)
```

Arguments

`domains`	domain names as character vector
`use_file`	path to a local Steven Black hosts file. If NULL, downloads from GitHub once per session and reuses it

Details

Steven Black's host list is a consolidated list from multiple sources including adaway.org, mvps.org, malwaredomainlist.com, and someonewhocares.org.

Listed hosts receive `blocked`; absent hosts have `status = "no_match"`. Neither outcome determines safety or a content subclass.

Publication dates come from the `.metadata.json` sidecar written by the downloader, checked against the file contents. `source_fetched_at` is the download time. Files without matching metadata return `NA` dates; filesystem timestamps are not used.

Value

A tibble following classification_results. Source-specific data are retained in `details`; all source labels are retained in `labels`.

References

https://github.com/StevenBlack/hosts

See Also

`source_vintage` for the provenance of every category source

Examples

```
## Not run:
stevenblack_cat("doubleclick.net")
stevenblack_cat(c("google.com", "googleadservices.com", "malware-example.com"))

## End(Not run)
```

`uni_cat` *Get Category from University Domain List*

Description

Fetches university domain json from: https://raw.githubusercontent.com/Hipo/university-domains-list/master/world_universities_and_domains.json # nolint: line_length_linter.

Usage

```
uni_cat(domains, use_file = NULL)
```

Arguments

`domains`	Vector of domain names.
`use_file`	Optional saved Hipo JSON file. Otherwise downloaded once per session.

Value

A tibble following classification_results. Source-specific data are retained in `details`; all source labels are retained in `labels`.

Examples

```
## Not run:
uni_cat(domains = "http://www.google.com")

## End(Not run)
```

ut1_cat *Get category from the UT-Capitole blacklists*

Description

Looks a domain up in the University of Toulouse (UT1) blacklists, the maintained successor to Shallalist. Unlike `shalla_cat()` and `dmoz_cat()`, which answer from lists that stopped publishing in 2022 and 2017, this list is updated continuously — so publication and download dates are recorded separately.

Usage

```
ut1_cat(domains, use_file = NULL)
```

Arguments

`domains`	Vector of domain names.
`use_file`	Path to the file written by `get_ut1_data()`. If `NULL`, looks for `ut1_domain_category.csv` in the working directory.

Details

The categories are UT1's own (`drogue`, `publicite`, `tricheur`, `agressif`, ...) and are reported verbatim. They do not share names with Shallalist's, and mapping one onto the other would discard exactly the distinctions the two taxonomies disagree about.

Publication dates come from the `.metadata.json` sidecar written by the downloader, checked against the file contents. `source_fetched_at` is the download time. Files without matching metadata return `NA` dates; filesystem timestamps are not used.

Value

A tibble following classification_results. Source-specific data are retained in `details`; all source labels are retained in `labels`.

See Also

`get_ut1_data()` to download the data, `source_vintage()` for how this source compares with the discontinued ones.

Examples

```
## Not run:
get_ut1_data()
ut1_cat(domains = c("example.com", "wikipedia.org"))

# what a maintained list says versus one that stopped in 2022
merge(ut1_cat("example.com"), shalla_cat("example.com"), by = "domain_name")

## End(Not run)
```

`virustotal_cat` *Get Category from VirusTotal*

Description

Returns category of content from multiple security vendors using the VirusTotal API v3. The function retrieves domain analysis results including categories from various security services. Not all services will have categories for all domains.

Usage

```
virustotal_cat(domains, api_key = NULL)
```

Arguments

`domains`	domain names as character vector
`api_key`	virustotal API key

Details

Get the API Access Key from https://www.virustotal.com/. Either pass the API Key to the function # nolint: line_length_linter. or set the environmental variable: `VirustotalToken`. Environment variables persist within a R session.

Value

A tibble following classification_results. Source-specific data are retained in `details`; all source labels are retained in `labels`.

References

https://docs.virustotal.com/reference/domains

Examples

```
## Not run:
virustotal_cat("http://www.google.com")
virustotal_cat(c("google.com", "facebook.com"))

## End(Not run)
```

`web_categories` *Default categories for domain-name inference*

Description

Default categories for domain-name inference

Usage

```
web_categories()
```

Value

A character vector of category names shared by the LLM adapters.

Examples

```
web_categories()
```

Index

∗ **keywords**
    glm_shalla, 13
∗ **model**
    glm_shalla, 13

adult_name_probability, 3

cache_clear, 4
classification_results, 4, *5*, *8*, *16*, *18*, *20–24*
claude_cat, 5
collect_content, 6
collect_content(), *4*, *9, 10*, *18, 19*

dmoz_cat, 8
dmoz_cat(), *23*

fetch_error_codes, 9
fetch_error_codes(), *6*, *17*
fetch_report, 10
fetch_report(), *7*, *9*

get_dmoz_data, *8*, 10
get_shalla_data, 11, *20*
get_stevenblack_data, 12
get_ut1_data, 12
get_ut1_data(), *23*
glm_shalla, 13

html_text_content, 14

news_url_signals, 15

openai_cat, 16

page_signals, 17
page_signals(), *7*, *14*
pie_cat, 18
pie_cat(), *2, 3*, *9, 10*

rdomains *(*rdomains-package*)*, 2
rdomains-package, 2
rdomains_cache_dir, 19
rdomains_cache_dir(), *7*

shalla_cat, 19
shalla_cat(), *23*
source_vintage, *8*, 20, *20, 21*
source_vintage(), *7*, *9*, *13*, *23*
stevenblack_cat, 21

tempdir(), *7*

uni_cat, 22
ut1_cat, 23
ut1_cat(), *13*

virustotal_cat, 24

web_categories, 25