Package 'farr'

November 17, 2025

Title Data and Code for Empirical Research in Accounting

Version 1.0.0

Description Handy functions and data to support the course book 'Empirical Research in Accounting: Tools and Methods' (1st ed.). Chapman and Hall/CRC. <doi:10.1201/9781003456230> and <https://iangow.github.io/far_book/>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.3.3

BugReports https://github.com/iangow/farr/issues

Imports dbplyr (>= 2.2.0), dplyr, magrittr, rlang, tidyr, tibble, readr, stringr, DBI, lubridate, rpart

Depends R (>= 3.5.0)

Suggests RPostgres, duckdb, knitr, rmarkdown, testthat (>= 3.0.0), spelling

Config/testthat/edition 3

Language en-US

URL https://github.com/iangow/farr

NeedsCompilation no

Author Ian Gow [aut, cre] (ORCID: <https://orcid.org/0000-0002-6243-8409>)

Maintainer Ian Gow <iandgow@gmail.com>

Repository CRAN

Date/Publication 2025-11-17 21:51:14 UTC

Contents

`aaer_dates` *AAER dates from SEC*

Description

A data set containing dates and descriptions for AAERs.

Usage

```
aaer_dates
```

Format

A tibble with 2,920 rows and 4 variables:

aaer_num AAER number

aaer_date Date

aaer_desc Description

year Year of AAER

`aaer_firm_year` *AAERs from Bao et al. (2020)*

Description

A data set containing AAER firms-years used in Bao et al. (2020).

Usage

```
aaer_firm_year
```

Format

A tibble with 415 rows and 4 variables:

p_aaer AAER identifier

gkvey GVKEY (firm identifier)

min_year First affected year

max_year Last affected year

Source

doi:10.1111/1475679X.12292

`apple_events` *Dates for Apple Events*

Description

A data set containing the dates of Apple media events since 2005.

Usage

```
apple_events
```

Format

A tibble with 47 rows and 3 variables:

event Description of event

event_date First date of event

end_event_date Last date of event

Source

https://en.wikipedia.org/wiki/List_of_Apple_Inc._media_events

---

`auc` *Area under curve*

Description

A function returning AUC.

Usage

```
auc(scores, response)
```

Arguments

`scores`	Probability that response is true or 1.
`response`	Responses coded as logical or 0-or-1.

Value

Vector including AUC

Source

https://blog.mbq.me/augh-roc/

https://stackoverflow.com/questions/4903092/calculate-auc-in-r

---

aus_banks *Australian banks*

Description

A data set containing identifying information for 10 Australian banks.

Usage

```
aus_banks
```

Format

A tibble with 10 rows and 3 variables:

gvkey GVKEY (firm identifier)

ticker Stock exchange ticker

co_name Bank name

---

aus_bank_funds *Australian bank fundamental data*

Description

A data set containing fundamental financial information for Australian banks.

Usage

```
aus_bank_funds
```

Format

A tibble with 283 rows and 7 variables:

gvkey GVKEY (firm identifier)

datadate Fiscal year-end

at Total assets

ib Income before extraordinary items

xi Extraordinary items

do Income from discontinued operations

aus_bank_rets *Australian bank stock market data*

Description

A data set containing fundamental financial information for Australian banks.

Usage

```
aus_bank_rets
```

Format

A tibble with 3,047 rows and 4 variables:

gvkey GVKEY (firm identifier)

datadate Last trading date of month

ret Stock return for month

mkt_cap Market capitalization on datadate

bloomfield_2021 *Firm-years in RDD analysis of Bloomfield (2021)*

Description

Firm-years in RDD analysis of Bloomfield (2021).

Usage

```
bloomfield_2021
```

Format

A tibble with 1,855 rows and 2 variables:

fyear Fiscal year

permco CRSP firm identifier (PERMCO)

Source

doi:10.1111/1475679X.12346

by_tag_year *Tags on StackOverflow*

Description

A data set containing data on tagged questions on StackOverflow.

Usage

```
by_tag_year
```

Format

A tibble with 40,518 rows and 4 variables:

year Year

tag Tag

number Number of questions with tag during year

year_total Total number of questions with tag during year

camp_attendance *Camp attendance*

Description

A simulated data set related to camp attendance.

Usage

```
camp_attendance
```

Format

A tibble with 1,000 rows and 2 variables:

id Student identifier

camp Indicator for student attendance at camp

`cmsw_2018` *Data for CMSW*

Description

Data on whistleblowers and enforcement actions from Call et al. (2018).

Usage

```
cmsw_2018
```

Format

A tibble with 1,133 rows and 31 variables:

recid CMSW record identifier

firmpenalty The total firm civil and criminal monetary penalties assessed against the firm, its parent and subsidiaries consisting of disgorgement, prejudgment interest, civil fines, criminal restitution, and criminal fines in millions of dollars

otherpenalty The total firm civil and criminal monetary penalties assessed against the agent firms and/or respondents (e.g., the audit firm, bankers, suppliers) in connection with the financial misrepresentation of the target firm, in millions of dollars

emppenalty The total civil and criminal penalties assessed against all employees consisting of disgorgement, prejudgment interest, civil fines, criminal restitution, and criminal fines in millions of dollars

empprisonmos Total incarceration consisting of jail, prison, home detention, and halfway house in months imposed upon employee respondents named in the enforcement action

selfdealflag An indicator variable equal to one if the violation includes self-dealing such as embezzlement and theft by respondents and equal to zero otherwise

blckownpct The percentage of blockholder ownership, defined as owners with at least five percent of common shares outstanding from the last 10-K or DEF 14A prior to the first public announcement the firm may be (is) subject to a regulatory enforcement action

initabret The value-weighted market-adjusted return measured at the close of trading on the initial public announcement date that the firm may be (is) subject to a regulatory enforcement action

wbflag An indicator variable equal to one if a whistleblower is associated with the enforcement action and equal to zero otherwise

tousesox Post-SOX action flag

lnvioperiod The natural logarithm of the total time the violation occurred in months as indicated in the regulatory enforcement proceedings

bribeflag An indicator variable equal to one if the enforcement actions includes charges under the Foreign Corrupt Practices Act for bribery of a foreign official and zero otherwise

mobflag An indicator variable equal to one if violation or any of the respondents were associated with a known organized crime family and zero otherwise

deter An indicator variable equal to one if the violation includes an offense for either option backdating, insider trading, or an offense related to an offering, IPO, merger, or reverse merger and equal to zero otherwise

lnempcleveln The natural logarithm of the total number of C-level respondents (e.g. CEO, COO, CFO, CAO, CMO, and CIO) named in the enforcement action

lnuscodecnt The natural logarithm of the total number of unique code sections and rules violated (charges) associated with the enforcement action

viofraudflag n indicator variable equal to one if fraud under 15 USC §§ 77q, 78j(b), or rules promulgated thereunder are included among the charges in the enforcement action

misledflag An indicator variable equal to one if the violation included violations of 17 CFR 240.13b2-2 that prohibits materially false or misleading statement to an accountant in connection with the preparation of financial statements and zero otherwise

audit8flag An indicator variable equal to one if the misreporting firm used a Big N auditor, and equal to zero otherwise

exectermflag An indicator variable equal to one if the firm terminated an executive respondent as a result of the violations and equal to zero otherwise

coopflag An indicator variable equal to one if the firm received credit in the assessment of penalties for cooperation as stated in regulatory enforcement documents during the course of the investigation and equal to zero otherwise

impedeflag An indicator variable equal to one if regulators acknowledged they were deliberately misled and/or charges were included for lying to investigators and equal to zero otherwise

pctinddir The percentage of the firm's directors that are independent from the last 10-K or DEF 14A prior to the first public announcement the firm may be (is) subject to a regulatory enforcement action

recidivist An indicator variable equal to one if the firm was previously the subject of a securities regulatory enforcement action and equal to zero otherwise

lnmktcap The natural logarithm of the market value of equity measured in millions of dollars prior to the first public announcement that the firm may be (is) subject to a regulatory enforcement action

mkt2bk The sum of market value of equity plus total assets minus total debt divided by total assets with market value determined below and total assets and total debt measured at the last fiscal year end prior to the first public announcement the firm may be (is) subject to a regulatory enforcement action

lev Total debt divided by total assets measured at the last fiscal year end prior to the first public announcement the firm may be (is) subject to a regulatory enforcement action

lndistance The natural logarithm of the distance in miles from the location of the firm's headquarters to the offices of the regulator assigned to the geographic area of the firm's headquarter location (closer of the SEC Regional Office or DOJ U.S. District Attorney).

ff12 Fama-French industry code (12-industry)

wbsource Whistleblower data source

wbtype Whistleblower type: tipster or nontipster

Source

doi:10.1111/1475679X.12177

comp *Data on accruals and auditor choice*

Description

A data set containing data about accruals for 2,000 firms.

Usage

```
comp
```

Format

A tibble with 16,237 rows and 14 variables:

gvkey GVKEY (firm identifier)

datadate Fiscal year-end

fyear Fiscal year

big_n Indicator for Big Four auditor

ta Total accruals (scaled by assets)

roa Return on assets

cfo Cash flow from operating activities (scaled by assets)

size Size

lev Leverage

mtb Market-to-book ratio

inv_at 1/Total assets

d_sale Change in revenue

d_ar Change in accounts receivable

ppe Property, plant & equipment (scaled by assets)

confusion_stats *Confusion statistics.*

Description

A function returning sensitivity and precision.

Usage

```
confusion_stats(scores, response, predicted = NULL, k = NULL)
```

Arguments

`scores`	Probability that response is true or 1.
`response`	Responses coded as logical or 0-or-1.
`predicted`	Predicted value coded as 0-or-1.
`k`	Percentage to classify as TRUE or 1.

Value

vector including sensitivity and precision

`fhk_firm_years` *Firm-years for replication of Fang, Huang and Karpoff (2016)*

Description

A data set containing the GVKEYs and datadates for firm-years used in Fang, Huang and Karpoff (2016).

Usage

```
fhk_firm_years
```

Format

A tibble with 60,272 rows × 2 variables.

gvkey GVKEY (firm identifier)

datadate Fiscal year-end

Source

doi:10.1111/jofi.12369

fhk_pilot *Treatment indicators for SHO pilot firms*

Description

A data set containing the tickers, GVKEYs, and treatment indicator for the SHO pilot program.

Usage

```
fhk_pilot
```

Format

A tibble with 3,030 rows × 4 variables.

ticker Ticker

gvkey GVKEY (firm identifier)

permno PERMNO (CRSP security identifier)

pilot SHO pilot program treatment indicator

Source

doi:10.1111/jofi.12369

form_deciles *Form deciles*

Description

Calculate deciles for a variable.

Usage

```
form_deciles(x)
```

Arguments

x	A vector for which deciles are to be calculated.

Value

vector

Examples

```
library(farr)
library(dplyr, warn.conflicts = FALSE)

df <-
    tibble(x = rnorm(100)) %>%
    mutate(dec_x = form_deciles(x))
df
```

`get_annc_dates` *Produce a table mapping announcements to trading dates*

Description

Produce a table mapping announcements to trading dates.

Usage

```
get_annc_dates(conn)
```

Arguments

conn	connection to a PostgreSQL or DuckDB database

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
library(RPostgres)
pg <- dbConnect(Postgres())
get_annc_dates(pg)

## End(Not run)
## End(Not run)
```

`get_event_cum_rets` *Produce a table of cumulative event returns*

Description

Produce a table of event returns from CRSP.

Usage

```
get_event_cum_rets(
  data,
  conn,
  permno = "permno",
  event_date = "event_date",
  win_start = 0,
  win_end = 0,
  end_event_date = NULL,
  suffix = ""
)
```

Arguments

`data`	data frame containing data on events
`conn`	connection to a PostgreSQL or DuckDB database
`permno`	string representing column containing PERMNOs for events
`event_date`	string representing column containing dates for events
`win_start`	integer representing start of trading window (e.g., -1)
`win_end`	integer representing start of trading window (e.g., 1)
`end_event_date`	string representing column containing ending dates for events
`suffix`	Text to be appended after "ret" in variable names

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
library(RPostgres)
pg <- dbConnect(Postgres())
events <- tibble(permno = c(14593L, 10107L),
                 event_date = as.Date(c("2019-01-31", "2019-01-31")))
get_event_cum_rets(events, pg)
```

```
## End(Not run)
## End(Not run)
```

`get_event_cum_rets_mth` *Produce a table of cumulative event returns using monthly data*

Description

Produce a table of event returns from CRSP

Usage

```
get_event_cum_rets_mth(
  data,
  conn,
  permno = "permno",
  event_date = "event_date",
  win_start = 0,
  win_end = 0,
  end_event_date = NULL,
  suffix = ""
)
```

Arguments

`data`	data frame containing data on events
`conn`	connection to a PostgreSQL or DuckDB database
`permno`	string representing column containing PERMNOs for events
`event_date`	string representing column containing dates for events
`win_start`	integer representing start of trading window (e.g., -1) in months
`win_end`	integer representing start of trading window (e.g., 1) in months
`end_event_date`	string representing column containing ending dates for events
`suffix`	Text to be appended after "ret" in variable names.

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
library(RPostgres)
pg <- dbConnect(Postgres())
events <- tibble(permno = c(14593L, 10107L),
                 event_date = as.Date(c("2019-01-31", "2019-01-31")))
get_event_cum_rets_mth(events, pg)

## End(Not run)
## End(Not run)
```

`get_event_dates` *Produce a table mapping announcements to trading dates*

Description

Produce a table of event dates for linking with CRSP.

Usage

```
get_event_dates(
  data,
  conn,
  permno = "permno",
  event_date = "event_date",
  win_start = 0,
  win_end = 0,
  end_event_date = NULL
)
```

Arguments

`data`	data frame containing data on events
`conn`	connection to a PostgreSQL or DuckDB database
`permno`	string representing column containing PERMNOs for events
`event_date`	string representing column containing dates for events
`win_start`	integer representing start of trading window (e.g., -1)
`win_end`	integer representing start of trading window (e.g., 1)
`end_event_date`	string representing column containing ending dates for events

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
pg <- dbConnect(RPostgres::Postgres())
events <- tibble(permno = c(14593L, 10107L),
                 event_date = as.Date(c("2019-01-31", "2019-01-31")))
get_event_dates(events, pg, win_start = -3, win_end = + 3)

## End(Not run)
## End(Not run)
```

`get_event_rets` *Produce a table of event returns*

Description

Produce a table of event returns from CRSP.

Usage

```
get_event_rets(
  data,
  conn,
  permno = "permno",
  event_date = "event_date",
  win_start = 0,
  win_end = 0,
  end_event_date = NULL
)
```

Arguments

`data`	data frame containing data on events
`conn`	connection to a PostgreSQL database
`permno`	string representing column containing PERMNOs for events
`event_date`	string representing column containing dates for events
`win_start`	integer representing start of trading window (e.g., -1)
`win_end`	integer representing start of trading window (e.g., 1)
`end_event_date`	string representing column containing ending dates for events

Value

tbl_df

Examples

```
## Not run:
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
pg <- dbConnect(RPostgres::Postgres())
events <- tibble(permno = c(14593L, 10107L),
                 event_date = as.Date(c("2019-01-31", "2019-01-31")))
get_event_rets(events, pg, win_start = -3, win_end = +3) %>%
  select(permno, event_date, date, ret)

## End(Not run)
## End(Not run)
```

`get_ff_ind` *Fetch Fama-French industry grouping.*

Description

Fetch Fama-French industry grouping from Ken French's website.

Usage

```
get_ff_ind(ind)
```

Arguments

`ind`	Fama-French industry grouping (e.g., 11, 48)

Value

tbl_df

Examples

```
## Not run:
get_ff_ind(5)
## End(Not run)
```

`get_got_data` *Generate simulated data as described in Gow, Ormazabal and Taylor (2010).*

Description

Function to generate simulated panel data as described in Gow, Ormazabal and Taylor (2010).

Usage

```
get_got_data(N = 400, T = 20, Xvol, Evol, rho_X, rho_E)
```

Arguments

`N`	Number of firms
`T`	Number of years
`Xvol`	Cross-sectional correlation of X
`Evol`	Cross-sectional correlation of errors
`rho_X`	Autocorrelation coefficient for firm-effect portion of X
`rho_E`	Autocorrelation coefficient for firm-effect portion of epsilon

Value

tibble

Source

https://www.jstor.org/stable/20744139

Examples

```
set.seed(2021)
test <- get_got_data(N = 500, T = 10, Xvol = 0.75,
                     Evol = 0.75, rho_X = 0.5, rho_E = 0.5)
```

get_idd_periods *Period for Inevitable Disclosure Doctrine (IDD)*

Description

Periods defined by precedent-setting legal cases adopting or rejecting the Inevitable Disclosure Doctrine (IDD) by state.

Usage

```
get_idd_periods(min_date, max_date)
```

Arguments

min_date	First date of sample period
max_date	Last date of sample period

Details

Three kinds of period by state:

- Pre-adoption
- Post-adoption
- Post-rejection

Value

tibble with four columns: state, period_type, start_date, end_date

Examples

```
idd_periods <- get_idd_periods(min_date = "1994-01-01",
                               max_date = "2010-12-31")
idd_periods
```

get_me_breakpoints *Create a table of with cut-offs for size portfolios*

Description

Create a table of with cut-offs for size portfolios

Usage

```
get_me_breakpoints(keep_max = FALSE)
```

Arguments

keep_max	Set to TRUE to keep upper-bound of highest decile. Default is FALSE, which will replace upper bound with Inf.

Examples

```
library(dplyr, warn.conflicts = FALSE)
get_me_breakpoints() %>% filter(month == '2022-04-01')
```

---

get_size_rets_monthly *Create a table of monthly returns for size portfolios*

Description

Create a table of monthly returns for size portfolios

Usage

```
get_size_rets_monthly()
```

Value

tbl_df

---

get_test_scores *A function returning data on test_scores*

Description

A function returning simulated data on test_scores.

Usage

```
get_test_scores(
  effect_size = 15,
  n_students = 1000L,
  n_grades = 4L,
  include_unobservables = FALSE,
  random_assignment = FALSE
)
```

Arguments

effect_size	Effect of attending camp on subsequent test scores
n_students	Number of students in simulated data set
n_grades	Number of grades in simulated data set
include_unobservables	Include talent in returned data (TRUE or FALSE)
random_assignment	Is assignment to treatment completely random? (TRUE or FALSE)

Value

tbl_df

Examples

```
set.seed(2021)
library(dplyr, warn.conflicts = FALSE)
get_test_scores() %>% head()
```

---

get_trading_dates *Produce a table mapping dates on CRSP to "trading days"*

Description

Produce a table mapping dates on CRSP to "trading days". Returned table has two columns: date, a trading date on CRSP; td, a sequence of integers ordered by date.

Usage

```
get_trading_dates(conn)
```

Arguments

conn	connection to a PostgreSQL or DuckDB database

Value

tbl_df

Examples

```
## Not run:
library(DBI)
library(dplyr, warn.conflicts = FALSE)
pg <- dbConnect(RPostgres::Postgres())
get_trading_dates(pg) %>%
  filter(between(date, as.Date("2022-03-18"), as.Date("2022-03-31")))

## End(Not run)
```

gvkey_ciks *GVKEY-CIK links*

Description

Link table from GVKEYs to CIKs

Usage

```
gvkey_ciks
```

Format

A tibble with 48,517 rows and 5 variables:

gvkey GVKEY (Compustat firm identifier)

iid Issue ID

cik CIK (SEC firm identifier)

first_date First link date

last_date Last link date

idd_dates *Dates for Inevitable Disclosure Doctrine (IDD)*

Description

Dates of precedent-setting legal cases adopting or reject the Inevitable Disclosure Doctrine (IDD) by state.

Usage

```
idd_dates
```

Format

A tibble with 24 rows and 3 variables:

state Two-letter state abbreviation

idd_date Date of precedent-setting legal case

idd_type Either "Adopt" or "Reject"

Source

doi:10.1016/j.jfineco.2018.02.008

iliev_2010 *Data on public float*

Description

Data on public float of listed companies from Iliev (2010).

Usage

```
iliev_2010
```

Format

A tibble with 7,214 and 9 variables:

gvkey Compustat firm identifier (GVKEY)

fyear Fiscal year

fdate Date of end of fiscal year

pfdate Date for public float value

pfyear Year for public float value

publicfloat Public float in $ million

mr Indicator for filing of a management report

af Indicator for accelerator filer

cik SEC firm identifier (CIK)

Source

doi:10.1111/j.15406261.2010.01564.x

llz_2018 *GVKEYs used in Li, Lin and Zhang (2018).*

Description

GVKEYs used in Li, Lin and Zhang (2018).

Usage

```
llz_2018
```

Format

A tibble with 5,830 rows and 1 variable:

gvkey GVKEY

Source

doi:10.1111/1475679X.12187

`load_parquet` *Function to load parquet file into database.*

Description

Function to read data from a parquet file data_dir/schema/table_name.parquet into a table in the DuckDB database at conn.

Usage

```
load_parquet(conn, table, schema = "", data_dir = Sys.getenv("DATA_DIR"))
```

Arguments

`conn`	DuckDB connection
`table`	Name of table to be loaded
`schema`	Database schema for table
`data_dir`	Directory for data repository

Value

Remote data frame in conn

`michels_2017` *Data on firms suffering natural disasters*

Description

Data on firms suffering natural disasters based on the sample in Michels (2017).

Usage

```
michels_2017
```

Format

A tibble with 423 rows and 12 variables:

cusip CUSIP supplied by Michels (2017)

eventdate Date of relevant natural disaster supplied by Michels (2017)

cik Matched CIK (SEC firm identifier)

permno Matched PERMNO (CRSP security identifier)

gvkey Matched GVKEY (Compustat firm identifier)

date_filed Date of next filing of type 10-Q, 10-K, 10QSB, 10-K405 after event

form_types List of relevant form types filed on date_filed

next_period_end Next fiscal period-end after event date

next_fqtr Fiscal quarter of next period-end after event date

prev_period_end Last fiscal period-end before event date

prev_fqtr Fiscal quarter of last period-end before event date

recognize Indicator for event being recognized (next_period_end before date_filed)

Source

doi:10.1111/1475679X.12128

`ndcg` *Calculate metric: NDCG at k*

Description

A function returning NDCG-at-k metric.

Usage

```
ndcg(scores, response, k = 0.01)
```

Arguments

`scores`	Probability that response is true or 1.
`response`	Responses coded as logical or 0-1.
`k`	Percentage to classify as TRUE or 1.

Value

vector including sensitivity and precision

pg_to_parquet *Save WRDS table as parquet file.*

Description

Function to get data from a table on the WRDS PostgreSQL server and save to local parquet file using DuckDB.

Usage

```
pg_to_parquet(table_name, schema, data_dir = Sys.getenv("DATA_DIR"))
```

Arguments

table_name	Name of table on WRDS
schema	Database schema for table
data_dir	Directory for data repository

Value

Number of rows created

roc *A function returning data for a ROC plot.*

Description

A function returning data for a ROC plot.

Usage

```
roc(scores, response)
```

Arguments

scores	Probability that response is true or 1.
response	Responses coded as logical or 0-or-1.

Value

tbl_df

rus *Random under-sampling function*

Description

Function to create temporary training dataset using distribution implied by w.

Usage

```
rus(y_train, ir = 1)
```

Arguments

y_train	df on the target variable.
ir	Imbalance ratio. Specifies how many times the under-sampled majority instances are over minority instances.

Details

Following MATLAB, function samples observations of the minority class with replacement and observations of the majority class without replacement.

Value

vector

rusboost *RUSBoost for two-class problems*

Description

RUSBoost for two-class problems.

Usage

```
rusboost(formula, df, size, ir = 1, learn_rate = 1, rus = TRUE, control)
```

Arguments

formula	A formula specify predictors and target variable. Target variable should be a factor of 0 and 1. Predictors can be either numerical and categorical.
df	A df frame used for training the model, i.e. training set.
size	Ensemble size, i.e. number of weak learners in the ensemble model
ir	Imbalance ratio. Specifies how many times the under-sampled majority instances are over minority instances.

learn_rate	Default of 1.
rus	TRUE for random undersampling; FALSE for AdaBoost with full sample
control	Control object passed onto rpart function.

Value

rusboost object

`sho_r3000` *Russell 3000 stocks at time of SEC Reg SHO sample formation*

Description

A data set containing the tickers and company names for Russell 3000 at time SEC created the pilot sample. Data are created from sample supplied by FHK.

Usage

```
sho_r3000
```

Format

A tibble with 3000 rows × 2 variables.

russell_ticker Ticker

russell_name Company name

Source

doi:10.1111/jofi.12369

`sho_r3000_gvkeys` *Russell 3000 sample used by SEC with GVKEYs*

Description

A data set containing the tickers, PERMNOs, GVKEYs, and treatment assignments for Russell 3000 sample used by SEC.

Usage

```
sho_r3000_gvkeys
```

Format

A tibble with 2,951 rows × 3 variables.

ticker Ticker

permno PERMNO (CRSP security identifier)

gvkey GVKEY (Compustat firm identifier)

pilot Indicator for stock being part of Reg SHO pilot program

Source

https://iangow.github.io/far_book/natural-revisited.html#the-sho-pilot-sample

---

sho_r3000_sample *Russell 3000 sample used by SEC*

Description

A data set containing the tickers, PERMNOs, and treatment assignments for Russell 3000 sample used by SEC.

Usage

```
sho_r3000_sample
```

Format

A tibble with 2,954 rows × 3 variables.

ticker Ticker

permno PERMNO (CRSP security identifier)

pilot Indicator for stock being part of Reg SHO pilot program

Source

https://iangow.github.io/far_book/natural-revisited.html#the-sho-pilot-sample

sho_tickers *Tickers of pilot firms for Reg SHO*

Description

A data set containing the tickers and company names for pilot firms from Reg SHO pilot. Data are scraped from the SEC's own website.

Usage

```
sho_tickers
```

Format

A tibble with 986 rows × 2 variables.

ticker Ticker

co_name Company name

Source

https://www.sec.gov/rule-release/34-50104

state_hq *Data on firm headquarters based on SEC EDGAR filings*

Description

Data on firm headquarters based on SEC EDGAR filings. Dates related to SEC filing dates. Rather than provide dates for all filings, data are aggregated into groups of filings by state and CIK and dates are collapsed into windows over which all filings for a given CIK were associated with a given state. For example, CIK 0000037755 has filings with a CA headquarters from 1994-06-02 until 1996-03-25, then filings with an OH headquarters from 1996-05-30 until 1999-04-05, then filings with a CA headquarters from 1999-06-11 onwards. To ensure continuous coverage over the sample period, it is assumed that any change in state occurs the day after the last observed filing for the previous state.

Usage

```
state_hq
```

Format

A tibble with 53,133 rows and 4 variables:

cik SEC's Central Index Key (CIK)
ba_state Two-letter abbreviation of state
min_date Date of first filing with CIK-state combination in a contiguous series of filings
max_date Date of last filing with CIK-state combination in a contiguous series of filings

Source

https://sraf.nd.edu/sec-edgar-data/10-x-header-data/

---

system_time *Version of* system.time() *that works with assignment*

Description

Print CPU (and other) times that expr used, return value of expr.

Usage

```
system_time(expr)
```

Arguments

expr	Valid R expression to be timed, evaluated and returned

Value

Result of evaluating expr

---

test_scores *Test scores*

Description

A simulated data set of test scores.

Usage

```
test_scores
```

Format

A tibble with 4,000 rows and 3 variables:

id Student identifier
grade School grade at time of test
score Test score

truncate *Truncate a vector.*

Description

Truncate a vector at `prob` and `1 - prob`. Extreme values are turned into `NA` values.

Usage

```
truncate(x, prob = 0.01, p_low = prob, p_high = 1 - prob)
```

Arguments

`x`	A vector to be winsorized
`prob`	Level (two-sided) for winsorization (e.g., 0.01 gives 1% and 99%)
`p_low`	Optional lower level for winsorization (e.g., 0.01 gives 1%)
`p_high`	Optional upper level for winsorization (e.g., 0.99 gives 99%)

Value

vector

Examples

```
trunced <- truncate(1:100, prob = 0.05)
min(trunced, na.rm = TRUE)
max(trunced, na.rm = TRUE)
```

undisclosed_names *Customer names that represent non-disclosures*

Description

Data to be combined with data in `compsegd.seg_customer` to create an indicator for non-disclosure of customer names.

Usage

```
undisclosed_names
```

Format

A tibble with 460 rows and 2 variables:

cnms Matches field in `compsegd.seg_customer` (WRDS)

disclosure Indicator that name is not disclosed

winsorize *Winsorize a vector*

Description

Winsorize a vector at `prob` and `1 - prob`.

Usage

```
winsorize(x, prob = 0.01, p_low = prob, p_high = 1 - prob)
```

Arguments

`x`	A vector to be winsorized
`prob`	Level (two-sided) for winsorization (e.g., 0.01 gives 1% and 99%)
`p_low`	Optional lower level for winsorization (e.g., 0.01 gives 1%)
`p_high`	Optional upper level for winsorization (e.g., 0.99 gives 99%)

Value

vector

Examples

```
winsorized <- winsorize(1:100, prob = 0.05)
min(winsorized, na.rm = TRUE)
max(winsorized, na.rm = TRUE)
```

zhang_2007_events *Event dates from Zhang (2007)*

Description

A data set containing the event dates used in Zhang (2007). Data obtained from Panel of Table of Zhang (2007). If an event spans multiple dates, then a row is included for each date.

Usage

```
zhang_2007_events
```

Format

A tibble with 30 rows × 3 variables.

event Identifier for the event

date Date of event

event_desc Description of the event

Source

doi:10.1016/j.jacceco.2007.02.002

`zhang_2007_windows` *Event windows from Zhang (2007)*

Description

A data set containing the event windows used in Zhang (2007). Data obtained from Panel of Table of Zhang (2007).

Usage

```
zhang_2007_windows
```

Format

A tibble with 17 rows × 3 variables.

event Identifier for the event

beg_date First date of event window

end_date Last date of event window

Source

doi:10.1016/j.jacceco.2007.02.002

Index