Package: avoncap (via r-universe)

November 6, 2024

Title AvonCap Study Analysis

Version 0.0.0.9028

Description A WIP set of functions allowing data load, wrangling of the AvonCap data set.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Imports dplyr, lubridate, magrittr, rlang, tibble, tidyr, utils, fs, readr, digest, forcats, methods, purrr, stringr, tidyselect (>= 1.2.0), yaml, ggplot2, stats, ggstance, glue, nplyr, writexl, patchwork, rappdirs, binom, scales, survival, data.tree, colorspace, dtrackr (>= 0.4.4)

Suggests knitr, rmarkdown, tidyverse, testthat

VignetteBuilder knitr

Depends R (>= 2.10)

LazyData true

Config/pak/sysreqs make libicu-dev libjpeg-dev libpng-dev librsvg2-dev libssl-dev libpoppler-cpp-dev libnode-dev libx11-dev zlib1g-dev

Repository https://bristol-vaccine-centre.r-universe.dev

RemoteUrl https://github.com/bristol-vaccine-centre/avoncap

RemoteRef HEAD

RemoteSha 8b2fc0b36b32b48d63d9842c0f5199d8ba0800cf

Contents

`.cached` *A simple pass-through cache for complex or long running operations*

Description

executes expr and saves the output as an RDS file indexed by has of code in expr and the hash of input variables (which should contain any variable inputs)

Usage

```
.cached(
  .expr,
  ...,
  .nocache = getOption("cache.disable", default = FALSE),
  .cache = getOption("cache.dir", rappdirs::user_cache_dir(utils::packageName())),
  .prefix = "cached",
  .stale = Inf
)
```

Arguments

`.expr` the code the output of which requires caching. Other than a return value this should not create side effects or change global variables.

`...` inputs that the code in expr depends on and changes in which require the code re-running, Could be Sys.Date()

`.nocache` an option to defeat the caching which can be set globally as options("cache.disable"=TRUE)

`.cache` the location of the cache as a directory. May get its value from options("cache.dir") or the default value of rappdirs::user_cache_dir("ggrrr")

`.prefix` a name of the operation so that you can namespace the cached files and do selective clean up operations on them

`.stale` the length of time in days to keep cached data before considering it as stale. can also be set by options("cache.stale")

Value

the output of .expr which will usually be a value

.cache_clear *Clear data from the passthrough cache for complex or long running operations*

Description

Clear data from the passthrough cache for complex or long running operations

Usage

```
.cache_clear(
  .cache = getOption("cache.dir", rappdirs::user_cache_dir(utils::packageName())),
  .prefix = ".*",
  interactive = TRUE
)
```

Arguments

`.cache`	the location of the cache as a directory. May get its value from options("ggrrr.cache.dir") or the default value of rappdirs::user_cache_dir("ggrrr")
`.prefix`	a regular expression matching the prefix of the cached item, so that do selective clean up operations. defaults to everything.
`interactive`	suppress `are you sure?` warning with a FALSE value (defaults to TRUE)

Value

nothing. called for side effects

.cache_delete_stale *Delete stale files in a cache*

Description

Staleness is determined by the number of days from 2am on the current day in the current timezone. A item cached for only one day becomes stale at 2am the day after it is cached. The time is configurable and option(cache.time_day_starts = 0) would be midnight. Automated analysis using caches and updated data should ensure that analysis does not cross this time point otherwise it may end up using old data.

Usage

```
.cache_delete_stale(
  .cache = getOption("cache.dir", rappdirs::user_cache_dir(utils::packageName())),
  .prefix = ".*",
  .stale = Inf
)
```

Arguments

`.cache` the location of the cache as a directory. May get its value from options("cache.dir") or the default value of rappdirs::user_cache_dir("ggrrr")

`.prefix` a name of the operation so that you can namespace the cached files and do selective clean up operations on them

`.stale` the length of time in days to keep cached data before considering it as stale.

Value

nothing. called for side effects.

`.cache_download` *Download a file into a local cache.*

Description

This function copies a remote file to a local cache once and makes sure it is reused.

Usage

```
.cache_download(
  url,
  ...,
  .nocache = getOption("cache.disable", default = FALSE),
  .cache = getOption("cache.download", rappdirs::user_cache_dir(utils::packageName())),
  .stale = Inf,
  .extn = NULL
)
```

Arguments

`url` the url to download

`...` ignored

`.nocache` if set to TRUE all caching is disabled

`.cache` the location of the downloaded files

`.stale` how long to leave this file before replacing it.

`.extn` the file name extension

Value

the path to the downloaded file

`all_files` *Scans the input directory and returns csv or xlsx files in that directory*

Description

Extracting metadata from the filename where present - particularly hospital, and year number

Usage

```
all_files()
```

Value

a dataframe containing filename, path, date, hospital, and study_year fields

`augment_data` *Sanitise AvonCap data columns*

Description

AvonCap data has lots of columns which are named in a difficult to remember fashion, composed of data items that have enumerated values with no semantics. This makes displaying them difficult and any filtering done on the raw data inscrutable. Depending on the source of the data some different columns may be present due to differences in the NHS and UoB data sets. The redcap database has some options that may be checklists and some that are radio buttons, both of these end up with mysterious names in the data.

Usage

```
augment_data(x, ...)
```

Arguments

`x`	- the raw data from `load_data()`
`...`	Arguments passed on to `augment_generic` `df` a data frame

Details

This function maps the data into a tidy dataframe with consistently named columns, and named factors where appropriate. If not present in the data the ethnicity

files Most of the sanitisation code is held in the `zzz-avoncap-mappings.R` file.

Value

a tracked dataframe with

`augment_generic` *Applies a set of functions to the whole dataframe*

Description

This sequences, catches errors and allows parameters to be passed by name

Usage

```
augment_generic(df, ...)
```

Arguments

`df`	a data frame
`...`	unnamed parameters are a list of functions, named parameters are passed to those functions (if they match formal arguments).

Value

the altered df

Examples

```
fn1 = function(df,v) {df %>% dplyr::filter(cut=="Fair") %>% dplyr::mutate(x_col = color)}
fn2 = function(df,v) {df %>% dplyr::filter(color==v$color$J)}
df = ggplot2::diamonds %>% augment_generic(fn1, fn2)
```

`binomial_proportion_points` *Dodged bar and whiskers proportions*

Description

This function plots a stacked bar of proportions for an input set of data

Usage

```
binomial_proportion_points(data, mapping, ..., width = 0.8, size = 0.5)
```

Arguments

`data`	the data
`mapping`	a aes mapping with at least `x` and `fill`. If facetting then `group` must contain the facet variable
`...`	passed to `geom_bar`
`width`	width of position dodge
`size`	the bar size

Value

a ggplot

`cut_integer` *Cut and label an integer valued quantity*

Description

Deals with some annoying issues classifying integer data sets, such as ages, into groups. where you want to specify just the change over points as integers and clearly label the resulting ordered factor.

Usage

```
cut_integer(
  x,
  cut_points,
  glue = "{label}",
  lower_limit = -Inf,
  upper_limit = Inf,
  ...
)
```

Arguments

`x` a vector of integer valued numbers, e.g. ages, counts

`cut_points` a vector of integer valued cut points which define the lower boundaries of conditions

`glue` a glue spec that may be used to generate a label. It can use `low`, `high`, `next_low`, or `label` as values.

`lower_limit` the minimum value we should include (this is inclusive for the bottom category) (default -Inf)

`upper_limit` the maximum value we should include (this is also inclusive for the top category) (default Inf)

`...` not used

Value

an ordered factor of the integer

Examples

```
cut_integer(stats::rbinom(20,20,0.5), c(5,10,15))
cut_integer(floor(stats::runif(100,-10,10)), cut_points = c(2,3,4,6), lower_limit=0, upper_limit=10)
```

`default_column_names` *default column naming mappings*

Description

default column naming mappings

Usage

```
default_column_names(...)
```

Arguments

`...`	additional named items to add

Value

a set of mappings

`denom_by_age_by_day` *The avoncap denominator dataset*

Description

The denominator is a time varying quantity

Usage

```
data(denom_by_age_by_day)
```

Format

A dataframe containing the following columns:

- method (character) - estimation method. The default is "Campling 2019"
- age (character) - the age category
- date (date) - the date for which this estimate is valid
- population (integer) - the esimtate of the population size for that age group on that day

No default value.

32592 rows and 4 columns

`derive_admission_episode` *Create a counter in the event of repeated admissions*

Description

This also will calculate a time interval between admissions. There is also a repeat admission instrument that this does not use.

Usage

```
derive_admission_episode(df, v)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

Value

a dataframe

`derive_aLRTD_categories` *The aLRTD incidence paper classifications*

Description

The 3 category classifications

Usage

```
derive_aLRTD_categories(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

- aetiological:
  - Confirmed SARS-CoV-2 - implies Infective
  - No evidence SARS-CoV-2 - implies Infective but not confirmed as SARS-CoV-2
  - Non-infective - presumed non infective
- clinical presentation:
  - Pneumonia - implies Infective
  - NP-LRTI - implies Infective
  - No evidence LRTI (include CRDE and HF)

Some cases do not get a clinical presentation in this. Typically they are people who have an infective cause, but LRTI and pneumonia have been excluded. These could be URTI and or incidental COVID cases.

Value

a dataframe

derive_antiviral_status

Create a flag for patients who have been given antivirals

Description

Create a flag for patients who have been given antivirals

Usage

```
derive_antiviral_status(df, v)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

Value

a dataframe

`derive_catchment_status` *Identify patients who are in the BNSSG ICB based on their GP practice name*

Description

Names are normalised by removing commonly mixed up components and

Usage

```
derive_catchment_status(df, v)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

Value

a dataframe

`derive_completed_vaccination_status` *Derive detailed vaccination status on admission*

Description

Vaccination is deemed to have had effect if given > 14 days before admission for 1st dose or >7 days before admission for subsequent doses. This does not account for previous infection which is not in the data set.

Usage

```
derive_completed_vaccination_status(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Value

a dataframe

derive_continuous_categories

Categorical scores for continuous variables

Description

Typically used in regression models with non-linear effects over splines

Usage

```
derive_continuous_categories(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

- Age category - UK demographic data ends at 85, and 65 key cut off in 5 year bands, so 10 year bands age categories end at 85 (N.b.) there is a more principled reason here. Boundaries fall approx 0.1, 0.2, 0.4, 0.6, 0.8 quantiles. Could merge first two groups but outcomes are usually different. Covid vaccination cohorts were in 5 year age groups, but vaccination prioirity was in these groups approximately.
- Age of eligibility for vaccines: 65+ Age of pneumovax eligibility
- CCI - 4 bands as defined in original Charleson paper: ** https://pubmed.ncbi.nlm.nih.gov/3558716/ ** in https://link.springer.com/article/10.1007/s10654-021-00802-z there is rationale given for not using the charleson score as a continuous value.
- Alternate CCI - 0,1,2,3+ is also used as a grouping in the original charleson paper
- Rockwood score - Completely independent versus dependent frailty levels.
- CURB65 categorisation - As per derivation study (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1746657/): 0-1 consider home treatment; 2 consider admit as inpatient; 3-5 admit, consider ICU.

Value

a dataframe

`derive_continuous_categories_pneumo` *Age and CURB score categories*

Description

This should be consistent with AvonCAP age / CURB cateories.

Usage

```
derive_continuous_categories_pneumo(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Value

a dataframe

`derive_covid_status` *Determine if an admission is proven SARS-CoV-2 PCR positive*

Description

SARS-CoV-2 PCR positive only lab confirmed diagnosis.

Usage

```
derive_covid_status(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

admission.covid_pcr_result:

- based on fields: c19_adm_swab and covid_19_diagnosis
- Patient reported, clinical diagnoses are assumed PCR negative (although possible in some cases they may not have been done).
- Lateral flows done in hospital are counted as PCR negative.
- negative admission swabs are counted as negative
- NA signifies test not done.

admission.is_covid:

- Binary confirmed or no-evidence.
- PCR results count as confirmed,
- Lateral flow results count as confirmed,
- anything else is no evidence (includes negatives and test not done)

Value

a dataframe

---

derive_diagnosis_categories

Create 4 non exclusive diagnostic categories

Description

Pneumonia if one of:

- Standard of care diagnosis of CAP (radiologically or clinically)
- Empyema or abscess
- Admission chest X-ray shows pneumonia

Usage

```
derive_diagnosis_categories(df, v)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

Details

NP-LRTI if:

- Not pneumonia and Standard of care LTRI diagnosis

Exacerbation of CRDE:

- Standard of care exacerbation COPD
- Standard of care exacerbation Non-COPD
- (N.B. may be pneumonia or NP-LRTI)

Heart failure:

- Standard of care congestive heart failure.

Value

a dataframe

`derive_effective_vaccination_status` *A simple vaccination status on admission as an ordered number of doses*

Description

This does not account for previous infection which is not in the data set.

Usage

```
derive_effective_vaccination_status(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Value

a dataframe

derive_genomic_variant *Give a inferred Alpha, Delta or Omicron status based on time alone.*

Description

This relies on date period during which we are very confidence that the only variants circulating are of a given type. These are quite conservative estimates based on the frequency of sequenced cases in the bristol area (according to the Sanger centre and to cases identified in the hospital testing)

Usage

```
derive_genomic_variant(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

Sanger centre data

- Pre-alpha before 05 Dec 2020
- Alpha between 13 Feb 2021 and 15 May 2021
- Delta between 01 Jun 2021 and 07 Nov 2021
- Omicron from 07 Feb 2022 to present

Value

a dataframe

derive_gp_linkage *Identify patients from the GP surgeries in linked primary care study*

Description

Identify patients from the GP surgeries in linked primary care study

Usage

```
derive_gp_linkage(df, v)
```

Arguments

`df` the dataframe.

`v` the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

Value

a dataframe

`derive_haematology_categories` *Binary outcomes for haematology data*

Description

- Elevated troponin : > 18: 18ng/L is simply the 99th percentile value Beckman assay we use as quoted by the IFCC. We elected to not use sex-specific 99th percentile values although they are also quoted here and you could incorporate into your analysis. I am sure you are aware of the 4th Universal definition of MI that requires a rise or fall above the 99th percentile etc.

Usage

```
derive_haematology_categories(df, v, ...)
```

Arguments

`df` the dataframe.

`v` the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

`...` ignored

Value

a dataframe

`derive_hospital_burden_outcomes` *Binary outcomes for hospital burden*

Description

These outcomes were tested in the Delta vs Omicron severity paper and sensitivity analysis. These are only defined for COVID cases.

Usage

```
derive_hospital_burden_outcomes(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

- O2 requirement within 7 days (various cut-offs)
- Any respiratory support in 7 days (various cut-offs)
- LOS > X days in first 7 days (various cut-offs)

Value

a dataframe

`derive_infective_classification` *Determine if an admission is due to an infective cause*

Description

Infective admissions are defined as any of:

- pneumonias
- NP-LRTI
- laboratory confirmed COVID diagnosis
- admission swab COVID positive

Usage

```
derive_infective_classification(df, v)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

Details

Infective admissions are excluded if:

- Standard of care states non-infectious process
- SOC non-LRTI (and none of the other categories above)

Any unknowns are defined as non-Infective

Value

a dataframe

`derive_invasive_status`

Pneumococcal invasive status and binary test category

Description

Pneumococcal invasive status and binary test category

Usage

```
derive_invasive_status(df, ...)
```

Arguments

`df`	the dataframe.
`...`	ignored

Value

a dataframe

derive_nosocomial_covid_status

Did the patient catch COVID in hospital

Description

Only relevant to SARS-CoV-2 PCR positive patient. Timing of positive test compared to admission: This relies on knowing dates and hence only works on the identifiable data sets,

Usage

```
derive_nosocomial_covid_status(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

Logic is:

- Community if PCR result predates admission
- Probably commuinity if PCR result within 7 days of admission
- Probably nosocomial if 7-28 days after admission
- Otherwise is it undefined.

Value

a dataframe

derive_nosocomial_status

Identify patients who were admitted already prior to study entry

Description

Hospital acquired COVID is recorded explicitly in 2 places for some patients. A large difference between admission date and enrollment date (<21 days) is suggestive in other cases. The data is probably only collected in COVID cases so shoudl be treated with caution.

Usage

```
derive_nosocomial_status(df, v)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

Value

a dataframe

`derive_pandemic_timings` *Date columns*

Description

Date columns

Usage

```
derive_pandemic_timings(date_col, prefix)
```

Arguments

`date_col`	the date column
`prefix`	a prefix for the columns to be added

Value

a `derive_...` style function to augment a data set containing `date_col` with a set of columns describing the timing.

derive_patient_identifier *Create a unique patient level id (if it does not already exist)*

Description

The patient identifier is derived from the record number or the first record number (ensuring it matches) an entry in the record number. This deals with multiple admissions in the data set. In the patient identifiable NHS data this is the NHS number.

Usage

```
derive_patient_identifier(df, v)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

Value

a dataframe

derive_pcv_groupings *Group pneumo serotypes according to e.g. vaccine coverage*

Description

A range of useful serotype groups is defined in the list `uad_groups`. The `default_pcv_map` gives a set of mappings to group headings that gives the overall serotype distribution by vaccine.

Usage

```
derive_pcv_groupings(
  df,
  ...,
  pcv_map = uad_pcv_map,
  not_matched = "Other",
  col_name = "pneumo.pcv_group"
)
```

Arguments

`df` the normalised urine antigen data

`...` ignored

`pcv_map` a 2 column data frame mapping `group` to `uad_analysis`

`not_matched` what to call the column of non-matched serotypes? Default is `Other`, but `Non vaccine type` might be preferred.

`col_name` the target column name for the pcv grouping (defaults to `pneumo.pcv_group`)

Details

The logic employed in combining elements is:

- any(result == "Unknown") ~ "Unknown"
- any(result == "Positive") ~ "Positive"
- all(result == "Negative") ~ "Negative"
- TRUE ~ "Other"

Value

an augmented data frame with an additional column defined by `col_name`

`derive_phe_pcv_group` *Get vaccine coverage group for known serotype*

Description

For the longitudinal oneumocococcal data, a range of useful serotype groups is defined in the list `avoncap::serotype_data`. The `avoncap::serotype_pcv_map` gives a set of mappings to (multiple) group headings that gives the overall serotype distribution by vaccine.

Usage

```
derive_phe_pcv_group(df, v, ...)
```

Arguments

`df` the dataframe.

`v` the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

`...` ignored

Value

a dataframe

derive_pneumococcal_categories

The pneumococcal incidence diagnostic classifications

Description

The 4 category disjoint classification.

Usage

```
derive_pneumococcal_categories(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

- pneumo.presentation_class:
  - CAP+/RAD+ - radiologically proved pneumonia
  - CAP+/RAD- - pneumonia without x-ray confirmation
  - NP-LRTI - non-pneumonic lower respiratory tract infection
  - No evidence LRTI - believed to be non-infective at admission, this last group is usually discarded from analysis, however it only really describes people without a clinical diagnosis of LRTI on admission. There could still be undiagnosed infection there, and some of these patients have COVID (possibly without lower respiratory symptoms?).

Value

a dataframe

derive_pneumococcal_high_risk

Determine if patient is in a high pneumococcal risk group

Description

High pneumococcal risk defined if any of the following:

- over 65 years old
- other pneumococcal risks
- comorbid copd
- interstitial lung disease
- cystic fibrosis
- hypertension
- CCF
- ischaemic heart disease
- chronic kidney disease
- chronic liver disease
- diabetes
- asthmatic with immunodeficiency
- on immunosupression

Usage

```
derive_pneumococcal_high_risk(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Value

a dataframe

`derive_pneumococcal_risk_category` *Determine pneumococcal risk group*

Description

Original algorithm from B1851202 SAP defines a 3 class risk group:

Usage

```
derive_pneumococcal_risk_category(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

High-risk (immunocompromised)

- Asplenia - not supported
- Cancer/Malignancy, Hematologic - OK
- Cancer/Malignancy, Solid Tumor - OK
- Chronic Kidney Disease - OK
- Human Immunodeficiency Virus (HIV) – AIDS - OK
- Human Immunodeficiency Virus (HIV) – No AIDS - OK
- Immunodeficiency - OK
- Immunosuppressant Drug Therapy - OK
- Organ Transplantation - OK
- Multiple Myeloma - not supported

At Risk (immunocompetent)

- Asthma - OK
- Alcoholism - OK
- Celiac Disease - not supported
- Chronic Liver Disease without Hepatic Failure - OK
- Chronic Liver Disease with Hepatic Failure - OK
- Chronic Obstructive Pulmonary Disease - OK
- Cochlear Implant - not supported
- Congestive Heart Failure - OK
- Coronary Artery Disease (CAD) - OK
- Chronic Neurologic Diseases - OK
- Coagulation factor replacement therapy - not supported
- CSF Leak - not supported
- Diabetes Treated with Medication - OK
- Down syndrome - OK
- Institutionalized in nursing home or LTC facility (Nursing home or long-term care facility for those with disability or dependency on subject characteristics/risk determinants eCRF page) - OK
- Occupational risk with exposure to metal fumes - OK

- Other Chronic Heart Disease - OK
- Other Chronic Lung Disease - OK
- Other pneumococcal disease risk factors - OK
- Previous Invasive Pneumococcal Disease - not supported
- Tobacco smoking (Tobacco/E-Cigarettes) - OK

Anything else is low risk

Value

a dataframe

`derive_pneumo_clinical_syndrome`
Add in clinical syndrome indicator

Description

A list of presentations based on site which

- LRTI
- Meningitis
- Effusion/Empyema
- Septic arthritis
- URTI
- Other

Usage

```
derive_pneumo_clinical_syndrome(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Value

a dataframe

`derive_pneumo_polyfill` *Make pneumo data compatible with AvonCAP*

Description

Needed for:

- `derive_simpler_comorbidities`
- `derive_pneumococcal_high_risk`
- `derive_pneumococcal_risk_category`

Usage

```
derive_pneumo_polyfill(df, ...)
```

Arguments

`df`	the dataframe.
`...`	ignored

Value

a dataframe

`derive_pneumo_uad_panel` *Calculate UAD panel for test*

Description

The panels are UAD1 for PCV13 serotypes, UAD2 for PPV23 serotypes.

Usage

```
derive_pneumo_uad_panel(df, ...)
```

Arguments

`df`	a pneumo serotype dataframe
`...`	ignored

Value

a dataframe with additional columns `pneumo.uad1_panel_result`, `pneumo.uad2_panel_result`, `pneumo.non_uad_panel_result`, `pneumo.serotype_summary_result`

derive_pneumo_uad_status

Calculate summary status from UAD (or other serotype) panel results

Description

logic is defined in derive_pcv_groupings().

Usage

```
derive_pneumo_uad_status(df, ...)
```

Arguments

df a pneumo serotype dataframe

... ignored

Value

a dataframe with additional columns pneumo.uad1_panel_result, pneumo.uad2_panel_result, pneumo.non_uad_panel_result, pneumo.serotype_summary_result

derive_polyfill_central

Polyfill data

Description

Some basic context to allow comparison to ED data.

Usage

```
derive_polyfill_central(df, v, ...)
```

Arguments

df the dataframe.

v the value set. usually precomputed by the augment framework the value set can be explicitly supplied with v = get_value_sets(df)

... ignored

Details

- All of the patients admitted

Value

a dataframe

`derive_polyfill_ed` *Polyfill ED data*

Description

The ED data has some different fields from the main avoncap data.

Usage

```
derive_polyfill_ed(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

- It is missing an admission cxr summary field needed to calculate pneumonia
- It has a fixed admission route of "A&E" (i.e. ED to non UK people)
- None of the patients admitted
- Hospital admission length of stay is zero

Value

a dataframe

`derive_presumed_diagnosis_categories` *Create presumed diagnostic categories*

Description

Pneumonia if one of:

- Initial diagnosis of CAP (supported by initial radiology or clinically)
- Empyema or abscess

Usage

```
derive_presumed_diagnosis_categories(df, v)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`

Details

Presumed clinical presentation:

- Pneumonia - implies Infective
- NP-LRTI - implies Infective
- No evidence LRTI (include CRDE and HF)

Value

a dataframe

`derive_qcovid` *Calculate a QCOVID2 score from AvonCap data source*

Description

uses inbuilt imd_to_townsend map. This implements a cut down version of the QCovid2 score depending on what data is available.

Usage

```
derive_qcovid(df, v = avoncap_df %>% get_value_sets())
```

Arguments

`df`	a normalised avoncap data source
`v`	a value set

Value

the same dataframe with additional columns,

- qcovid2.log_hazard, covid2.hazard_ratio: a log hazard rate for the QCOVID2 score where missing data is substituted with the reference value for the QCOVID2 population.
- qcovid2.log_comorbid_hazard, qcovid2.comorbid_hazard_ratio: a log hazard rate for the comorbid conditions and not including age and BMI.

derive_quintile_category

Split a continuous variable into quintiles

Description

Split a continuous variable into quintiles

Usage

```
derive_quintile_category(col, labels = c("1-short", "2", "3", "4", "5-long"))
```

Arguments

`col`	the continuous data column that is to be categorised by quintile.
`labels`	the category labels

Value

a `derive_...` style function that augments a data set with col `xxx` with col `xxx_quintile` containing the quintiles

derive_severe_disease_outcomes

Binary outcomes for severe disease

Description

- Confirmed death within 30 days (subject to potential censoring)
- Confirmed death within 1 year (subject to potential censoring). The date of censoring depends on when the mortality data was updated. Currently this is 04 Oct 2024
- Confirmed death (any length follow up)
- Any ICU admission

Usage

```
derive_severe_disease_outcomes(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

described in aLRTD paper. These outcomes are

Value

a dataframe

---

derive_simpler_comorbidities

Rationalise some of the more detailed comorbidities

Description

and generate some summary values

Usage

```
derive_simpler_comorbidities(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

- simple DM without insulin dependence
- Solid / Haematological / Any cancer present binary indicators
- any chronic resp dx: i.e. any of asthma, bronchiectasis, chronic pleural disease, COPD, interstitial lung dx, cyctic fibrosis, other chronic resp dx
- any chronic heart disease: pulmonary htn, CCF, IHD, previous MI, congential heart dx, hypertension, AF, other arrythmia, other heart dx, other other heart dx
- Stroke or TIA binary
- Any immune compromise binary (immunodeficient or on immune suppressants)

Value

a dataframe

derive_survival_censoring *Survival outcomes*

Description

Expects as days since admission:

- `survival.length_of_stay` - length of stay until discharge or death (NA if still in hosptial),
- `survival.uncensored_time_to_death` - time until death (NA if alive at last obs),
- `survival.last_observed_event` - last time patient observed alive.

Usage

```
derive_survival_censoring(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

Calculates

- a 30 day survival duration and censoring status for survfit
- a 1 year survival duration and censoring status for survfit
- Hospital length of stay and censoring status for survfit
- Categorical length of stay and 30 day survival 0-3, 4-6, 7-13, 14-29, gte 30

Survival data will be of the form:

`survival.30_day_death_xxx`, `survival.1_yr_death_xxx`, `survival.30_day_discharge`

`xxx_time`: for this is the follow up time to event in days (max 30 or 365).

`xxx_event`: The event type indicator

- 0 = alive at event (censored),
- 1 = dead.

or for length of stay:

- 0 = still inpatient / died (censored),
- 1 = discharged from hospital

A survival model will be of the form:

`survival::Surv(time = xxx_time, event=xxx_event) ~ ...`

Value

a dataframe

`derive_survival_times_avoncap`

Survival analysis times

Description

Fixes a data issue with length of stay and survival duration being filled in across 2 columns. and missing last observation dates so that we can calculate survival censoring consistently in other data sets.

Usage

```
derive_survival_times_avoncap(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

Calculates:

- A consistent length of stay - shortest of length of stay and 30 day and 1 yr survival duration
- A consistent uncensored time to death - shortest of 30 day and 1 yr survival duration
- A consistent time to last observation

Value

a dataframe

`derive_survival_times_pneumo` *Survival analysis times*

Description

Fixes a data issue with length of stay and survival duration being filled in across 2 columns. and missing last observation dates so that we can calculate survival censoring consistently in other data sets.

Usage

```
derive_survival_times_pneumo(df, v, ...)
```

Arguments

`df` the dataframe.

`v` the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v` = `get_value_sets(df)`

`...` ignored

Details

Calculates:

- A consistent length of stay - shortest of length of stay and 30 day and 1 yr survival duration
- A consistent uncensored time to death - shortest of 30 day and 1 yr survival duration
- A consistent time to last observation

Value

a dataframe

`derive_template` *Derived data function template*

Description

Derived data function template

Usage

```
derive_template(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Value

a dataframe

`derive_vaccination_timings` *Derive times from vaccination to symptom onset*

Description

If symptom duration is not given it is assumed to be zero.

Usage

```
derive_vaccination_timings(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Value

a dataframe

derive_vaccine_combinations

Deprecated - Vaccine combinations are less relevant now

Description

There are too many potential combinations with 4th, 5th and sixth dose to make this useful.

Usage

```
derive_vaccine_combinations(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Value

a dataframe

derive_WHO_outcome_score

determine WHO outcome score

Description

Scores 0-3 are for community cases.

Usage

```
derive_WHO_outcome_score(df, v, ...)
```

Arguments

`df`	the dataframe.
`v`	the value set. usually precomputed by the augment framework the value set can be explicitly supplied with `v = get_value_sets(df)`
`...`	ignored

Details

We generally can't tell the difference between 7 and 8.

- 4: Hospitalised; no oxygen therapy
- 5: Hospitalised; oxygen by mask or nasal prongs
- 6: Hospitalised; oxygen by NIV or high flow
- 7: Intubation and mechanical ventilation, pO2/FiO2 >= 150 or SpO2/FiO2 >= 200
- 8: Mechanical ventilation pO2/FIO2 <150 (SpO2/FiO2 <200) or vasopressors
- 9: Mechanical ventilation pO2/FiO2 <150 and vasopressors, dialysis, or ECMO
- 10: Dead

Value

a dataframe

`extract_dependencies` *Get provenance of data column*

Description

When a data set is normalised or augmented the original column names are stored as metadata. This helps us determine how a particular item was created. In future this will be useful for documentation.

Usage

```
extract_dependencies(data, col, original = TRUE)
```

Arguments

`data`	the dataframe
`col`	the column as a symbol
`original`	map the names to the original column names from the data. If this is false the function returns a list of current normalised column names.

Value

a named list of dependencies and original column names for a given column

`find_new_field_names` *Get the transformed columns from original field names*

Description

Get the transformed columns from original field names

Usage

```
find_new_field_names(normalised, fields)
```

Arguments

`normalised`	the transformed data set.
`fields`	a vector of field names

Value

a named list mapping original to new columns

`frameworks` *Frameworks*

Description

The list of validation, normalisation and augmentation frameworks. There should be one validation per data set. The may be mulitple normalisations and augmentations depending on the aspect of the data we are extracting (e.g. re-nesting flattened data.)

`get_value_sets` *Get a value set list of a dataframe*

Description

This function examines a dataframe and returns a list of the columns with sub-lists as all the options for factors. This provides programmatic access (and automcomplete) to the values available in a dataframe, and throws and early error if we try and access data by a variable that does not exist.

Usage

```
get_value_sets(df)
```

Arguments

`df`	a dataframe to examine

Value

a list of lists with the column name and the factor levels as list, as a `checked list`.

`icb_surgeries` *GP surgeries in the Bristol ICB area*

Description

The denominator relates only to patients coming from these GP surgeries

Usage

```
data(icb_surgeries)
```

Format

A dataframe containing the following columns:

- code - an official ODS code for the GP surgery
- name - the official surgery name.

82 rows and 2 columns

`imd_to_townsend` *High level IMD to Townsend score map*

Description

A high level mapping from IMD to Townsend score This is inaccurate as townsend score

Details

A data frame with 10 rows and 2 columns:

imd_decile The IMD

mean_townsend the average townsend score for this IMD ...

Source

https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/845345/File_7_-_All_IoD2019_Scores__Ranks__Deciles_and_Population_Denominators_3.csv

https://s3-eu-west-1.amazonaws.com/statistics.digitalresources.jisc.ac.uk/dkan/files/Townsend_Deprivation_Scores/Scores/Scores-%202011%20UK%20LSOA.csv

input — *Locate the input directory*

Description

Locate the input directory

Usage

```
input(...)
```

Arguments

`...`	the sub paths within the input directory

Value

a path to the input directory and sub paths if provided

Examples

```
# devtools::load_all()
try({
  avoncap::set_input("~/Data/avoncap")
  avoncap::input("nhs-extract")

  avoncap::all_files()


  # exact match on filename column of all_data()
  avoncap::most_recent_files("AvonCAPLRTDCentralDa")


  # or matches by lower case startWith on directory
  avoncap::most_recent_files("nhs-extract","deltave")


  avoncap::most_recent_files("metadata")

  avoncap::valid_inputs()
})
```

key_dates *Key dates:*

Description

A list of key dates:

- mortality_updated - the last time the NHS mortality data was extracted and added to AvonCAP
- min_alpha - earliest observation of the alpha variant
- max_wuhan - last observation of the wuhan variant
- min_delta - earliest observation of the delta variant
- max_alpha - last observation of the alpha variant
- min_omicron - earliest observation of the omicron variant
- max_delta - last observation of the delta variant

The default catchment population for AvonCAP is limited to the Bristol, North Somerset and South Gloucestershire Integrated Care Board (BNSSG ICB). This list is the list of GP surgeries considered part of the denominator.

Details

- code - the NHS ODS organisational code of the practice.
- name - the official name of the practice

km_plot *Faceted Kaplan-Meier plot*

Description

Faceted Kaplan-Meier plot

Usage

```
km_plot(
  df,
  coxmodel,
  facet = NULL,
  ...,
  maxtime = NULL,
  ylab = if (!invert) "surviving (%)" else "affected (%)",
  xlab = "time (days)",
  facetlab = NULL,
  ylim = (if (invert) c(0, NA) else c(NA, 100)),
  n_breaks = 5,
  heights = c(10, 1),
  invert = FALSE,
  show_label = FALSE,
  show_legend = TRUE
)
```

Arguments

`df`	the data
`coxmodel`	the cox model output of survival::coxph from the data
`facet`	the division to highlight in the KM strata. Defaults to first term on the lhs of the cox model formula
`...`	Arguments passed on to `survival::survfit` `formula` either a formula or a previously fitted model
`maxtime`	the longest x value to plot (optional)
`ylab`	the y axis label
`xlab`	the x axis label
`facetlab`	a label to add as a facet title
`ylim`	the range to show on the KM plot
`n_breaks`	number of x axis breaks to display this also determines the timing and number of "at risk" counts to display.
`heights`	the relative height between the KM plot and the "at risk" table
`invert`	reverse survival statistics to count number of affected
`show_label`	show the label on the at risk table (which is somewhat redundant as items are coloured)
`show_legend`	show the legend for the strata. (This is sometimes redundant if the at risk table is labelled)

Value

a ggplot patchwork.

Examples

```
cox = survival::coxph(survival::Surv(time, status) ~ trt + celltype + karno +
  diagtime + age + prior , data = survival::veteran)

km_plot(survival::veteran, cox)
km_plot(survival::veteran, cox, facet = 1)

km_plot(survival::veteran, cox, "celltype", show_label=TRUE) &
   ggplot2::theme(legend.position="bottom")

km_plot(survival::veteran, cox, "trt", show_label=TRUE) &
   ggplot2::theme(legend.position="bottom")
```

`load_data` *Load data and check structure*

Description

Loads the AvonCap data from a set of csv files, which may optionally be qualified by site (`'BRI' or 'NBT'`) and database year (`'y1', 'y2', 'y3'`) as part of the file name. This selects the most recent files earlier than the `reproduce_at` date and detects whether they are in a set of files.

Usage

```
load_data(
  type,
  subtype = NULL,
  reproduce_at = as.Date(getOption("reproduce.at", default = Sys.Date())),
  merge = TRUE,
  ...
)
```

Arguments

`type` the file category see `valid_inputs()` for current list in input directory

`subtype` the subtype from `valid_inputs()`

`reproduce_at`
- the date at which to cut off newer data files

`merge`
- setting to `TRUE` forces multiple files be merged into a single data frame by losing mismatching columns.

`...`
- passed to cached may specifically want to use ‘.nocache=TRUE“

Details

The files are loaded as csv as checked that files have (A) the same columns, (B) the same type (or are empty) (C) have any major parse issues. It then merges the files into a single dataframe, if possible, otherwise it will return the individually loaded files as a list of dataframes.

Value

either a list of dataframes or a single merged dataframe

Examples

```
try(load_data("nhs-extract","deltave"))
```

map_avoncap_central *Core avoncap normalisation*

Description

- record_number -> admin.record_number (name)
- what_was_the_first_surveil -> admin.first_record_number (name)
- ac_study_number -> admin.consented_record_number (study_id)
- nhs_number -> admin.patient_identifier (ppi)
- duplicate -> admin.duplicate (yesno)
- enrollment_date -> admin.enrollment_date (date)
- admission_type -> admission.admission_route (list)
- study_year -> admin.study_year (name)
- file -> admin.data_file (name)
- week_number -> admin.week_number (name)
- c19_diagnosis -> diagnosis.standard_of_care_COVID_diagnosis (list)
- clinical_radio_diagnosis -> diagnosis.clinical_or_radiological_LRTI_or_pneumonia (yesno)
- c19_adm_swab -> diagnosis.admission_swab (list)
- c19_test_type -> diagnosis.test_type (list)
- qualifying_symptoms_signs -> diagnosis.qualifying_symptoms_signs (name)
- cc_critieria -> diagnosis.meets_case_control_criteria (yesno)
- cc_pos_date -> diagnosis.first_COVID_positive_swab_date (date)
- gender -> demog.gender (list)
- age_at_admission -> demog.age (double)
- age_march -> demog.age_in_march_2021 (double)
- imd -> demog.imd_decile (name)
- gp_practice -> admin.gp_practice_old (name)
- gp_practice_drop_down -> admin.gp_practice (list)
- smoking -> demog.smoker (list)
- ethnicity2 -> demog.ethnicity (list)
- care_home -> demog.care_home_resident (yesno)
- hapcovid_screening -> admission.non_lrtd_hospital_acquired_covid (yesno)
- hospital_covid -> admission.hospital_acquired_covid (yesno)
- drugs -> demog.no_drug_abuse, demog.alcohol_abuse, demog.ivdu_abuse, demog.marijuana_abuse, demog.other_inhaled_drug_abuse (checkboxes)
- vaping -> demog.vaping (list)
- alc_units -> demog.units_of_alcohol (name)

- np_swab -> admin.np_swab_taken_1 (list)
- adm_np_type -> admin.np_swab_site_1 (list)
- np_date -> admin.np_swab_date_1 (date)
- days_adm_npswab -> admin.np_swab_day_since_admission (double)
- np_swab_2 -> admin.np_swab_taken_2 (list)
- adm_np_type_2 -> admin.np_swab_site_2 (list)
- np_date_2 -> admin.np_swab_date_2 (date)
- np_swab_3 -> admin.np_swab_taken_3 (list)
- adm_np_type_3 -> admin.np_swab_site_3 (list)
- np_date_3 -> admin.np_swab_date_3 (date)
- saliva -> admin.saliva_sample_taken (list)
- saliva_date -> admin.saliva_sample_date (date)
- days_adm_saliva -> admin.saliva_sample_day_since_admission (double)
- sputum -> admin.sputum_sample_taken (list)
- sputum_date -> admin.sputum_sample_date (date)
- days_adm_sputum -> admin.sputum_sample_day_since_admission (double)
- pt_ad_ur -> admin.urine_sample_needed (yesno)
- adm_ur_taken -> admin.urine_sample_taken (list)
- nourine_reason -> admin.urine_sample_failure_reason (list)
- adm_np_type_2 -> admin.urine_sample_site (list)
- adm_ur_date -> admin.urine_sample_date (date)
- days_adm_urine -> admin.urine_sample_day_since_admission (double)
- adm_serum_tak -> admin.serum_sample_taken (list)
- adm_seru_date -> admin.serum_sample_date (date)
- days_adm_serum -> admin.serum_sample_day_since_admission (double)
- contraindication -> vaccination.covid_vaccine_contraindicated (yesno)
- covid19_vax -> vaccination.covid_vaccination (list)
- covidvax_date -> vaccination.first_dose_date (date)
- covidvax_dose_2 -> vaccination.second_dose_date (date)
- covidvax_dose_3 -> vaccination.third_dose_date (date)
- covidvax_dose_4 -> vaccination.fourth_dose_date (date)
- covidvax_dose_5 -> vaccination.fifth_dose_date (date)
- covidvax_dose_6 -> vaccination.sixth_dose_date (date)
- brand_of_covid19_vaccinati -> vaccination.first_dose_brand (list)
- covid19vax_brand_2 -> vaccination.second_dose_brand (list)
- covid19vax_brand_3 -> vaccination.third_dose_brand (list)
- covid19vax_brand_4 -> vaccination.fourth_dose_brand (list)

- covid19vax_brand_5 -> vaccination.fifth_dose_brand (list)
- covid19vax_brand_6 -> vaccination.sixth_dose_brand (list)
- c19vaxd1_adm -> admission.time_since_first_vaccine_dose (name)
- c19vaxd2_adm -> admission.time_since_second_vaccine_dose (name)
- c19vaxd3_adm -> admission.time_since_third_vaccine_dose (name)
- c19vaxd4_adm -> admission.time_since_fourth_vaccine_dose (name)
- c19vax5_adm -> admission.time_since_fifth_vaccine_dose (name)
- c19vax6_adm -> admission.time_since_sixth_vaccine_dose (name)
- flu_date -> vaccination.last_flu_dose_date (date)
- fluvax_adm_d1 -> admission.time_since_last_flu_vaccine_dose (name)
- ppv23_date -> vaccination.last_pneumococcal_dose_date (date)
- ppv23vax_adm_d -> admission.time_since_last_pneumococcal_vaccine_dose (name)
- c19_variant -> genomic.variant (variant)
- year -> admission.year (double)
- study_week -> admission.study_week (double)
- admission_date -> admission.date (date)
- hospital -> admin.hospital, toupper (text_to_factor)
- adm_diagnosis -> admission.presumed_CAP_radiologically_confirmed, admission.presumed_CAP_clinically_confirm admission.presumed_CAP_no_radiology, admission.presumed_LRTI, admission.presumed_Empyema_or_abscess, admission.presumed_exacerbation_COPD, admission.presumed_exacerbation_non_COPD, admission.presumed_congestive_heart_failure, admission.presumed_non_infectious_process, admission.presumed_non_LRTI (checkboxes)
- ics -> admission.on_inhaled_corticosteroids (yesno)
- immsup -> admission.on_immunosuppression (yesno)
- psi_class -> admission.pneumonia_severity_index_class (list)
- crb_test_mai -> admission.curb_65_severity_score (list)
- news_2_total -> admission.news2_score (name)
- pulse_ox -> admission.oximetry (name)
- rr -> admission.respiratory_rate (name)
- fio2 -> admission.max_oxygen (name)
- systolic_bp -> admission.systolic_bp (name)
- diastolic_bp -> admission.diastolic_bp (name)
- hr -> admission.heart_rate (name)
- temperature -> admission.temperature (list)
- symptom_days_preadmit -> admission.duration_symptoms (double)
- previous_infection -> admission.previous_covid_infection (yesno_unknown)
- previousinfection_date -> admission.previous_covid_infection_date (date)
- c19d_preadm -> admission.time_since_covid_diagnosis (name)

- rockwood -> admission.rockwood_score (name)
- cci_total_score -> admission.charlson_comorbidity_index (name)
- height -> admission.height (name)
- weight -> admission.weight (name)
- bmi -> admission.BMI (double)
- first_radio -> admission.cxr_normal, admission.cxr_pneumonia, admission.cxr_heart_failure, admission.cxr_pleural_effusion, admission.cxr_covid_changes, admission.cxr_other (checkboxes)
- c19_peep -> day_7.max_peep (name)
- c19_hospadm -> day_7.length_of_stay (list)
- c17_high -> day_7.max_care_level (list)
- c19icuon -> day_7.still_on_icu (yesno)
- c19_icudays -> day_7.icu_length_of_stay (list)
- c19_vent -> day_7.max_ventilation_level (list)
- c19_ox -> day_7.max_o2_level (list)
- c19_ionotropes -> day_7.ionotropes_needed (yesno)
- c19_complication -> day_7.PE, day_7.DVT, day_7.ARF, day_7.NSTEMI, day_7.STEMI, day_7.cardiac_failure, day_7.new_AF, day_7.new_other_arrythmia, day_7.inpatient_fall, day_7.other_complication, day_7.no_complication (checkboxes)
- c19_death7d -> day_7.death (yesno)
- c19_meds -> treatment.dexamethasone, treatment.remdesevir, treatment.tocilizumab, treatment.sarilumab, treatment.in_drug_trial, treatment.no_drug_treatment, treatment.sotrovimab (checkboxes)
- hospital_length_of_stay -> outcome.length_of_stay, floor (integer)
- survival_days -> outcome.survival_duration, round (integer)
- ip_death -> outcome.inpatient_death (yesno)
- days_in_icu -> outcome.icu_duration (double)
- did_the_patient_have_respi -> outcome.respiratory_support_needed (yesno)
- number_of_days_of_ventilat -> outcome.ventilator_duration (double)
- ett_days -> outcome.endotracheal_tube_duration (double)
- renal_replacement_therapy -> outcome.renal_support_duration (double)
- complications -> outcome.acute_renal_failure, outcome.liver_dysfunction, outcome.hospital_acquired_infection, outcome.acute_respiratory_distress_syndrome, outcome.NSTEMI, outcome.STEMI, outcome.new_AF, outcome.new_other_arrhthmia, outcome.stroke, outcome.DVT, outcome.PE, outcome.heart_failure, outcome.fall_in_hospital, outcome.reduced_mobility, outcome.increasing_care_requirement, outcome.no_complications (checkboxes)
- ventilatory_support -> outcome.highest_level_ventilatory_support (list)
- did_the_patient_receive_ec -> outcome.received_ecmo (yesno)
- inotropic_support_required -> outcome.received_ionotropes (yesno_unknown)
- lrtd_30d_outcome -> outcome.functional_status (list)

- survive_1yr -> outcome.one_year_survival (yesno)
- survival_1yr_days -> outcome.one_year_survival_duration (integer)
- yr_survival_complete -> outcome.one_year_survival_complete (list)
- fever2 -> symptom.abnormal_temperature (yesno)
- pleurtic_cp -> symptom.pleuritic_chest_pain (yesno)
- cough2 -> symptom.cough (yesno)
- sput_prod -> symptom.productive_sputum (yesno)
- dyspnoea -> symptom.dyspnoea (yesno)
- tachypnoea2 -> symptom.tachypnoea (yesno)
- confusion -> symptom.confusion (yesno)
- anosmia -> symptom.anosmia (yesno_unknown)
- ageusia -> symptom.ageusia (yesno_unknown)
- dysgeusia -> symptom.dysguesia (yesno_unknown)
- fever -> symptom.fever (yesno_unknown)
- hypothermia -> symptom.hypothermia (yesno_unknown)
- chills -> symptom.chills (yesno_unknown)
- headache -> symptom.headache (yesno_unknown)
- malaise -> symptom.malaise (yesno_unknown)
- wheeze -> symptom.wheeze (yesno_unknown)
- myalgia -> symptom.myalgia (yesno_unknown)
- worse_confusion -> symptom.worsening_confusion (yesno_unknown)
- general_det -> symptom.general_deterioration (yesno_unknown)
- ox_on_admission -> symptom.oxygen_required_on_admission (yesno_unknown)
- resp_disease -> comorbid.no_resp_dx, comorbid.copd, comorbid.asthma, comorbid.resp_other (checkboxes)
- other_respiratory_disease -> comorbid.bronchiectasis, comorbid.interstitial_lung_dx, comorbid.cystic_fibrosis, comorbid.pulmonary_hypertension, comorbid.chronic_pleural_dx, comorbid.other_chronic_resp_dx (checkboxes)
- chd -> comorbid.no_heart_dx, comorbid.ccf, comorbid.ihd, comorbid.hypertension, comorbid.other_heart_dx (checkboxes)
- mi -> comorbid.previous_mi (yesno)
- other_chd -> comorbid.congenital_heart_dx, comorbid.af, comorbid.other_arrythmia, comorbid.pacemaker, comorbid.valvular_heart_dx, comorbid.other_other_heart_dx (checkboxes)
- diabetes -> comorbid.diabetes (list)
- dm_meds -> comorbid.diabetes_medications (list)
- neurological_disease -> comorbid.neuro_other, comorbid.cva, comorbid.tia, comorbid.hemiplegia, comorbid.paraplegia, comorbid.no_neuro_dx (checkboxes)
- dementia -> comorbid.no_dementia, comorbid.dementia, comorbid.cognitive_impairment (checkboxes)

- cancer -> comorbid.solid_cancer (list)
- haem_malig -> comorbid.no_haemotological_cancer, comorbid.leukaemia, comorbid.lymphoma (checkboxes)
- ckd -> comorbid.ckd (list)
- liver_disease -> comorbid.liver_disease (list)
- gastric_ulcers -> comorbid.gastric_ulcers (yesno)
- pvd -> comorbid.periph_vasc_dx (yesno)
- ctd -> comorbid.connective_tissue_dx (yesno)
- immunodeficiency -> comorbid.immunodeficiency (yesno)
- other_pn_disease -> comorbid.other_pneumococcal_risks (yesno)
- transplant -> comorbid.transplant_recipient (yesno)
- pregnancy -> comorbid.pregnancy (list)
- hiv -> comorbid.no_HIV, comorbid.HIV, comorbid.AIDS (checkboxes)
- final_soc_lrtd_diagnosis -> diagnosis.SOC_CAP_radiologically_confirmed, diagnosis.SOC_CAP_clinically_confirmed diagnosis.SOC_CAP_no_radiology, diagnosis.SOC_LRTI, diagnosis.SOC_Empyema_or_abscess, diagnosis.SOC_exacerbation_COPD, diagnosis.SOC_exacerbation_non_COPD, diagnosis.SOC_congestive_heart_fail diagnosis.SOC_non_infectious_process, diagnosis.SOC_non_LRTI (checkboxes)
- covid_19_diagnosis -> diagnosis.covid_19_diagnosis (list)
- ppv23 -> vaccination.pneumovax (list)
- flu_vaccine -> vaccination.influenza_vaccination (list)
- abx_14d_prior -> admission.pre_admission_antibiotics_given (yesno_unknown)
- antibiotic_used -> admission.pre_admission_antibiotic (checkboxes_to_nested_list)
- antiplatelets -> admission.antiplatelet_therapy (list)
- anticoagulants -> admission.anticoagulant_therapy (list)
- statins -> admission.cholesterol_lowering_therapy (list)
- hypertensives -> admission.antihypertensive_therapy (list)
- antiviral_14d_prior -> admission.pre_admission_antiviral (checkboxes_to_nested_list)

Usage

```
map_avoncap_central()
```

Value

a list

`map_avoncap_consent` *Core avoncap consent*

Description

- consented -> admin.consented (list)
- ppc -> admin.pp_consented (list)
- withdrawal -> admin.withdrawal (yesno)
- consent_urine -> admin.consent_for_urine (yesno)
- consent_blood -> admin.consent_for_blood (yesno)
- consent_resp_samples1 -> admin.consent_for_respiratory_samples (yesno)

Usage

```
map_avoncap_consent()
```

Value

a list

`map_avoncap_ed` *Avoncap ED normalisation*

Description

All the ED data is also mapped using the `map_avoncap_central()` list as it si quite similar

Usage

```
map_avoncap_ed()
```

Details

- ed_hours -> outcome.emergency_dept_length_of_stay (name)
- ed_reattendance -> admin.ed_episodes_in_last_30_days (name)
- hosp_adm_30d -> outcome.admitted_within_30_days (yesno)
- hosp_adm_7d -> outcome.admitted_within_7_days (yesno)
- home_d_1 -> outcome.days_since_last_ed_episode (name)
- radiology_result_1___2 -> radio.consistent_with_pneumonia_1 (yesno)
- radiology_result_2___2 -> radio.consistent_with_pneumonia_2 (yesno)

Value

a list

map_avoncap_ed_consent *ED consent*

Description

- consented -> admin.consented (list)
- ppc -> admin.pp_consented (list)
- withdrawal -> admin.withdrawal (yesno)
- consent_urine -> admin.consent_for_urine (yesno)
- consent_blood -> admin.consent_for_blood (yesno)
- consent_resp_samples1 -> admin.consent_for_respiratory_samples (yesno)

Usage

```
map_avoncap_ed_consent()
```

Value

a list

map_avoncap_haem *Normalise the avoncap data haematology data*

Description

- record_number -> admin.record_number (name)
- ac_study_number -> admin.consented_record_number (study_id)
- ph_7_35 -> haem.blood_gas_ph (double)
- glucose -> haem.glucose (double)
- albumin -> haem.albumin (double)
- wcc -> haem.white_cell_count (double)
- eos -> haem.eosinophils (double)
- hb -> haem.haemoglobin (double)
- haematocrit -> haem.haemotocrit (double)
- pmn -> haem.neutrophils (double)
- lymphocytes -> haem.lymphocytes (double)
- crp -> haem.crp (double)
- na_result -> haem.sodium (double)
- ur_result -> haem.urea (double)

- egfr -> haem.egfr (double)
- sars_cov2_antigen -> haem.sars_cov2_antigen (trunc_double)
- ferritin -> haem.ferritin (double)
- troponin -> haem.troponin (double)
- nt_probnp -> haem.pro_bnp (double)
- d_dimer -> haem.d_dimer (double)
- patient_blood_group -> haem.blood_group (list)

Usage

```
map_avoncap_haem()
```

Value

a list

map_avoncap_micro *Normalise the avoncap data microbiology data*

Description

- microtest_done -> micro.test_performed (yesno)
- microtest_date -> micro.test_date (date)
- microday -> micro.test_days_from_admission (pos_integer)
- micro_test -> micro.test_type (list)
- micro_isolates -> micro.pathogen_detected (yesno_unknown)
- isolate_identified -> micro.pathogen, .micro_isolate_list (checkboxes_to_nested_list)
- pn_result -> micro.pneumo_serotype_status (list)
- pn_st -> micro.pneumo_serotype (pneumo_serotype)
- micro_lab -> micro.sent_to_central_lab (yesno_unknown)
- pen_susceptibility -> micro.penicillin_susceptibility (checkboxes_to_list)
- septrin_susceptibility -> micro.septrin_susceptibility (checkboxes_to_list)
- doxy_susceptibility -> micro.doxycycline_susceptibility (checkboxes_to_list)
- levoflox_suscept -> micro.levofloxacin_susceptibility (checkboxes_to_list)
- cef_susceptibility -> micro.ceftriaxone_susceptibility (checkboxes_to_list)
- pn_uat_result -> micro.pneumo_binax_now (list)
- lg_uat_result -> micro.pneumo_legionella_uat (list)
- micro_final_report -> micro.is_final_report (yesno)

Usage

```
map_avoncap_micro(instrument)
```

Arguments

`instrument`	the numeric instrument number

Value

a list

`map_avoncap_pneumococcal`

Normalise the avoncap pneumococcal data

Description

- participant_number -> admin.record_number (name)
- hospital -> admin.hospital (list)
- nhs_number -> admin.patient_identifier (ppi)
- age_at_admission -> demog.age (double)
- sex -> demog.gender (list)
- test_date -> pneumo.test_date (date)
- test -> pneumo.test_type (list)
- serotype -> pneumo.phe_serotype (pneumo_serotype)
- smoker -> demog.smoker (list)
- resp_disease -> comorbid.no_resp_dx, comorbid.copd, comorbid.asthma, comorbid.bronchiectasis, comorbid.pulmonary_fibrosis, comorbid.resp_other (checkboxes)
- chd -> comorbid.no_heart_dx, comorbid.ccf, comorbid.ihd, comorbid.hypertension, comorbid.af, comorbid.other_heart_dx (checkboxes)
- mi -> comorbid.previous_mi (yesno)
- ckd -> comorbid.ckd (list)
- liver_disease -> comorbid.liver_disease (list)
- diabetes -> comorbid.diabetes (list)
- dm_meds -> comorbid.diabetes_medications (list)
- dementia -> comorbid.no_dementia, comorbid.dementia, comorbid.cognitive_impairment (checkboxes)
- neurological_disease -> comorbid.neuro_other, comorbid.cva, comorbid.tia, comorbid.hemiplegia, comorbid.paraplegia, comorbid.no_neuro_dx (checkboxes)
- gastric_ulcers -> comorbid.gastric_ulcers (yesno)
- dysphagia -> comorbid.dysphagia (yesno)

- pvd -> comorbid.periph_vasc_dx (yesno)
- ctd -> comorbid.connective_tissue_dx (yesno)
- immunodeficiency -> comorbid.immunodeficiency (yesno)
- other_pn_disease -> comorbid.other_pneumococcal_risks (yesno)
- hiv -> comorbid.no_HIV, comorbid.HIV, comorbid.AIDS (checkboxes)
- cancer -> comorbid.solid_cancer (list)
- haem_malig -> comorbid.no_haemotological_cancer, comorbid.leukaemia, comorbid.lymphoma (checkboxes)
- recent_chemo -> comorbid.recent_chemotherapy (yesno)
- recent_radiotherapy -> comorbid.recent_radiotherapy (yesno)
- transplant -> comorbid.transplant_recipient (yesno)
- pregnancy -> comorbid.pregnancy (list)
- drugs -> demog.no_drug_abuse, demog.alcohol_abuse, demog.ivdu_abuse, demog.marijuana_abuse, demog.other_inhaled_drug_abuse (checkboxes)
- immsup -> admission.on_immunosuppression (yesno)
- weight_problem -> comorbid.bmi_status (list)
- concomittant_flu -> comorbid.influenza_infection (yesno)
- hcv -> comorbid.hepatitis_c (yesno)
- ppv23 -> vaccination.ppv23_vaccination (list)
- flu_vaccine -> vaccination.flu (list)
- cci_total_score -> admission.charlson_comorbidity_index (name)
- los_days -> outcome.length_of_stay (double)
- amts -> admission.triage_score (list)
- resp_rate -> admission.respiratory_rate (double)
- sats_ra -> admission.saturations_on_room_air (double)
- systolic_bp -> admission.systolic_bp (double)
- diastolic_bp -> admission.diastolic_bp (double)
- crb65_score -> admission.crb_65_severity_score (list)
- curb65_score -> admission.curb_65_severity_score (list)
- antibiotic_route -> outcome.antibiotic_route (list)
- antibiotic_days -> outcome.antibiotic_duration (double)
- infection_site -> admission.infection_site (list)
- deranged_lfts -> outcome.abnormal_lft (yesno)
- aki -> outcome.acute_kidney_injury (yesno)
- pleural_effusion -> outcome.pleural_effusion (yesno)
- empyema -> outcome.empyema (yesno)
- discharge_destination -> outcome.discharge_to (list)
- icu -> outcome.admitted_icu (yesno)

- niv -> outcome.non_invasive_ventilation (yesno)
- intubation -> outcome.intubation (yesno)
- recurrent_pneumonia -> outcome.recurrent_pneumonia (yesno)
- ecmo -> outcome.received_ecmo (yesno)
- inotropes -> outcome.received_ionotropes (yesno)
- trachy -> outcome.tracheostomy (yesno)
- inpatient_death -> outcome.inpatient_death (yesno)
- death_30days -> outcome.death_within_30_days (yesno)
- death_1year -> outcome.death_within_1_year (yesno)
- survival_days -> outcome.survival_duration (name)
- albumin -> haem.albumin (double)
- wcc -> haem.white_cell_count (double)
- hb -> haem.haemoglobin (double)
- pmn -> haem.neutrophils (double)
- lymphocytes -> haem.lymphocytes (double)
- crp -> haem.crp (double)
- na_result -> haem.sodium (double)
- ur_result -> haem.urea (double)
- egfr -> haem.egfr (double)
- creatinine -> haem.creatinine (double)
- cxr_sides -> radio.cxr_infection (list)
- cxr_lobes -> radio.cxr_lobar_changes (list)
- death_5year -> outcome.death_within_5_years (yesno)
- survival_days_2 -> outcome.5_yr_survival_duration (name)
- imd_decile -> demog.imd_decile (name)

Usage

```
map_avoncap_pneumococcal()
```

Value

a list

`map_avoncap_radio` *Normalise the avoncap data radiology data*

Description

- radio_exam -> radio.test_performed (yesno)
- radiology_date -> radio.test_date (date)
- radiodays -> radio.test_days_from_admission (pos_integer)
- radio_test -> radio.test_type (list)
- radiology_result -> radio.alrtd_finding (checkboxes_to_nested_list)
- radiology_other_result -> radio.non_alrtd_finding (checkboxes_to_nested_list)

Usage

```
map_avoncap_radio(instrument)
```

Arguments

`instrument`	the numeric instrument number

Value

a list

`map_avoncap_virol` *Normalise the avoncap data virology data*

Description

- viral_testing_performed -> virol.test_performed (yesno)
- virology_date_of_asst -> virol.test_date (date)
- viroldays -> virol.test_days_from_admission (pos_integer)
- specimen_type -> virol.test_type (list)
- virus_isolated -> virol.pathogen_detected (yesno)
- test_type -> virol.test_type (list)
- virus_pathogen -> virol.pathogen, .virol_isolate_list (checkboxes_to_nested_list)
- virol_patient_lab -> virol.test_provenance (list)

Usage

```
map_avoncap_virol(instrument)
```

Arguments

`instrument`	the numeric instrument number

Value

a list

`map_urine_antigens` *Normalise the urinary antigen data*

Description

- RESULT -> pneumo.urine_antigen_result, .x (text)
- EVENT_DATE -> pneumo.test_date (date)
- ANALYSIS -> pneumo.urine_antigen_test (name)
- SUBJECT -> admin.consented_record_number (study_id)
- BARCODE -> pneumo.urine_antigen_sample_id (name)

Usage

```
map_urine_antigens()
```

Value

a list

`map_urine_binax` *Normalise the urinary antigen data (binax results)*

Description

- RESULT -> pneumo.binax_result, .x (text)
- EVENT_DATE -> pneumo.test_date (date)
- SUBJECT -> admin.consented_record_number (study_id)
- BARCODE -> pneumo.urine_antigen_sample_id (name)

- RESULT -> pneumo.binax_result, .x (text)
- EVENT_DATE -> pneumo.test_date (date)
- SUBJECT -> admin.consented_record_number (study_id)
- BARCODE -> pneumo.urine_antigen_sample_id (name)

Usage

```
map_urine_binax()

map_urine_binax()
```

Value

a list

a list

---

most_recent_files   *find most recent files of a specific type*

Description

find most recent files of a specific type

Usage

```
most_recent_files(
  type = "",
  subtype = NULL,
  reproduce_at = as.Date(getOption("reproduce.at", default = Sys.Date()))
)
```

Arguments

`type`	see valid_inputs() for current list of supported types in input directory
`subtype`	see valid_inputs() for list of supported filenames
`reproduce_at`	after this date new files are ignored. This enforces a specific version of the data.

Value

a list of the file paths to the most up to date files of the given type relevant to each site and study year

Examples

```
# devtools::load_all()
try({
  avoncap::set_input("~/Data/avoncap")
  avoncap::input("nhs-extract")

  avoncap::all_files()


  # exact match on filename column of all_data()
  avoncap::most_recent_files("AvonCAPLRTDCentralDa")


  # or matches by lower case startWith on directory
  avoncap::most_recent_files("nhs-extract","deltave")


  avoncap::most_recent_files("metadata")

  avoncap::valid_inputs()
})
```

`normalise_data` *Sanitise AvonCap data columns*

Description

AvonCap data has lots of columns which are named in a difficult to remember fashion, composed of data items that have enumerated values with no semantics. This makes displaying them difficult and any filtering done on the raw data inscrutable. Depending on the source of the data some different columns may be present due to differences in the NHS and UoB data sets. The redcap database has some options that may be checklists and some that are radio buttons, both of these end up with mysterious names in the data.

Usage

```
normalise_data(rawData, instrument = NULL, ...)
```

Arguments

`rawData`
- the raw data from `load_data()`

`instrument` the numeric instrument number if applicable

`...` Arguments passed on to `normalise_generic`

`remove_mapped` gets rid of original columns for which we have a mapping (leaving the new versions)

`remove_unmapped` gets rid of columns for which we do not have a mapping

`mappings` a set of mappings (see `zzz-avoncap-mappings.R`)

`messages` a set of `dtrackr` glue specs that populate the first box fo the flow chart. (can use `{files}`, `{reproduce_at}`, `{date}`, `{.total}`)

`data_source_info`
- if not null a filename, and the function will write out a file with the details of the input files used.

Details

This function maps the data into a tidy dataframe with consistently named columns, and named factors where appropriate. The mapping is defined in data.

files Most of the sanitisation code is held in the `normalise-xxx.R` file. but these in turn may depend on the `mapping-xxx.R` files

Value

a tracked dataframe with n

original_field_names *Get the mapping of transformed columns back to original*

Description

Get the mapping of transformed columns back to original

Usage

```
original_field_names(data, inverse = TRUE)
```

Arguments

`data`	the transformed data set.
`inverse`	give the data as a old -> new mapping for finding normalised names of original columns. if false gives it as new->old for finding original names of normalised columns

Value

a named list mapping original to new columns

phe_serotypes *Pneumococcal UAD serotypes*

Description

A somewhat complete list of pneumococcal serotypes as seen in Bristol

readable_label *Get a label for a column*

Description

Get a label for a column

Usage

```
readable_label(columnVar, colNames = default_column_names())
```

Arguments

columnVar	the column name as a string
colNames	bespoke column names mapping (see default_column_names(...))

Value

a mapped column name

readable_label_mapping *Get a readable label for the AvonCap data as a named list (for ggplot)*

Description

Get a readable label for the AvonCap data as a named list (for ggplot)

Usage

```
readable_label_mapping(x, ...)

## S3 method for class 'data.frame'
readable_label_mapping(x, colNames = default_column_names(...), ...)

## S3 method for class 'list'
readable_label_mapping(x, colNames = default_column_names(...), ...)

## S3 method for class 'character'
readable_label_mapping(x, colNames = default_column_names(...), ...)

## Default S3 method:
readable_label_mapping(x, colNames = default_column_names(...), ...)
```

Arguments

`x` either the column names as strings, or a dataframe

`...` ignored

`colNames` a mapping to convert a column name (as a string) to a readable label

Value

a named list of the labels for the columns

Methods (by class)

- `readable_label_mapping(data.frame)`: for data frames
- `readable_label_mapping(list)`: for lists
- `readable_label_mapping(character)`: for character vectors
- `readable_label_mapping(default)`: defaults

`relevel_serotypes` *Relevel serotype data into an factor based on PCV group status and serotype name.*

Description

Relevel serotype data into an factor based on PCV group status and serotype name.

Usage

```
relevel_serotypes(serotypes, ..., exprs)
```

Arguments

`serotypes` a vector of serotypes as a factor or character.

`...` an unwrapped version of the `exprs` parameter

`exprs` a list of formulae with a predicate on the LHS and a PCV group name on the RHS. which are interpreted as the parameters for a `dplyr::case_when` call. This must be protected against interpretation by wrapping it in `rlang::exprs()`. The predicates are tested against `avoncap::serotype_data$map` and could use any of the following columns 'c("4", "6B", "9V", "14", "18C", "19F", "23F", "1", "3", "5", "6A", "7F", "19A", "22F", "33F", "8", "10A", "11A", "12F", "15B", "2", "9N", "17F", "20")','c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE)','c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE)','c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE,

FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE)’,’c(TRUE, FALSE, FALSE, FALSE, FALSE)’,’c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE)’,’c(FALSE, FALSE, FALS a default option of the form `TRUE ~ "Non PCV serotype"` must exist to capture unmatched items.

Examples

```
x = rlang::exprs(
  PCV7 ~ "PCV7",
  PCV15 ~ "PCV15-7",
  TRUE ~ "Non-PCV15 serotype"
)
relevel_serotypes(avoncap::phe_serotypes, exprs=x)
relevel_serotypes(avoncap::phe_serotypes)

relevel_serotypes(avoncap::phe_serotypes,
  PCV24Affinivax ~ "Affinivax",
  TRUE ~ "Non-affinivax"
)
```

`save_data_source_info` *Write file source information out to a text files*

Description

Write file source information out to a text files

Usage

```
save_data_source_info(..., .file)
```

Arguments

`...`	A list of data frames loaded with the `load_data(...)` call
`.file`	the output file location

Value

the file name written (invisibly)

scale_fill_serotype *A ggplot scale for pneumococcal serotypes that keeps PCV groups together*

Description

The scale groups colours by PCV group, but it is important to have the source data using the same levels as this scale otherwise the colour legend will be ordered in a different sequence. This can be achieved using relevel_serotypes,

Usage

```
scale_fill_serotype(
  ...,
  palette_fn = scales::brewer_pal(palette = "Dark2"),
  undefined = "#606060",
  exprs = rlang::exprs()
)
```

Arguments

`...` Arguments passed on to ggplot2::scale_fill_manual

`values` a set of aesthetic values to map data values to. The values will be matched in order (usually alphabetical) with the limits of the scale, or with breaks if provided. If this is a named vector, then the values will be matched based on the names instead. Data values that don't match will be given na.value.

`aesthetics` Character string or vector of character strings listing the name(s) of the aesthetic(s) that this scale works with. This can be useful, for example, to apply colour settings to the colour and fill aesthetics at the same time, via aesthetics = c("colour", "fill").

`breaks` One of:

- NULL for no breaks
- waiver() for the default breaks (the scale limits)
- A character vector of breaks
- A function that takes the limits as input and returns breaks as output

`na.value` The aesthetic value to use for missing (NA) values

`palette_fn` a function that returns a set of colours for a number of levels. Such functions can be obtained from things like scales::brewer_pal(...)

`undefined` the colour for the last group which is assumed to be the Unknown types

`exprs` a list of formulae with a predicate on the LHS and a PCV group name on the RHS. which are interpreted as the parameters for a dplyr::case_when call. This must be protected against interpretation by wrapping it in rlang::exprs(). The predicates are tested against avoncap::serotype_data$map and could use any of the following columns 'c("4", "6B", "9V", "14", "18C", "19F", "23F", "1", "3", "5", "6A", "7F", "19A", "22F", "33F", "8", "10A", "11A", "12F", "15B", "2", "9N", "17F", "20")’,’c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE)’,’c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE)’,’c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE)’,’c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE)’,’c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE)’,’c(FALSE, FALSE, FALS a default option of the form `TRUE ~ "Non PCV serotype"` must exist to capture unmatched items.

Value

A ggplot2 scale

`serotype_data` *Pneumococcal UAD serotype groups and crossmaps*

Description

A list of pneumococcal serotype / UAD cross mappings

`serotype_pcv_map` *Pneumococcal serotype PCV groups*

Description

Pneumococcal serotype PCV groups

`serotype_uad_map` *Serotype UAD mappings*

Description

Serotype UAD mappings

`set_input` *Sets the location of data for an analysis*

Description

Also performs some structure checks and makes sure that the README files are in place.

Usage

```
set_input(path)
```

Arguments

`path`	the path to the input directory

Value

the full path to the directory

`spline_term_plot` *Spline term marginal effects plot*

Description

Spline term marginal effects plot

Usage

```
spline_term_plot(
  coxmodel,
  var_name,
  xlab = var_name,
  max_y = NULL,
  n_breaks = 7
)
```

Arguments

`coxmodel`	an output of a coxph model
`var_name`	a variable that is involved in a spline term
`xlab`	x axis label
`max_y`	maximium hazard ratio to display on y axis. Inferred from the central estimates if missing, which will most likely cut off confidence intervals
`n_breaks`	The number of divisions on the y axis

Value

a ggplot

`stacked_barplot` *Stacked bar plot*

Description

This function plots a stacked bar of proportions for an input set of data

Usage

```
stacked_barplot(data, mapping, ...)
```

Arguments

`data`	the data
`mapping`	a aes mapping with at least `x` and `fill`. If facetting then `group` must contain the facet variable
`...`	passed to `geom_bar`

Value

a ggplot

Examples

```
stacked_barplot(
    ggplot2::diamonds,
    ggplot2::aes(x=cut, fill=clarity, group=color)
  )+
  ggplot2::facet_wrap(dplyr::vars(color))
```

`start_date_of_week` *Convert a study week back into a date*

Description

This is poorly named as only give the start date is the input is an integer

Usage

```
start_date_of_week(study_week)
```

Arguments

`study_week`	does accept decimals and returns the nearest whole date to the value

Value

a vector of sudy_week numbers

`study_week` *Convert a date to a study week*

Description

Convert a date to a study week

Usage

```
study_week(dates)
```

Arguments

`dates`	a list of date objects

Value

an integer number of weeks since 2019-12-30

`uad_groups` *UAD serotype groups*

Description

UAD serotype groups

`uad_pcv_map` *UAD PCV map*

Description

UAD PCV map

upset_plot *Upset plot with counts stratified by a categorical column*

Description

Upset plot with counts stratified by a categorical column

Usage

```
upset_plot(df, boolean_cols, categorical_col, lbl_size = 5)
```

Arguments

`df`	the data
`boolean_cols`	a tidyselect specification selecting the columns to be used as binary one-hot encoded classes
`categorical_col`	a column containing a disjoint category as a factor
`lbl_size`	font sise of the label

Value

a ggplot

validate_data *Validate AvonCap raw data*

Description

Runs a set of QA checks. This function dispatches the call in a data set specific function using the `type` and `subtype` of the data set. The checks are in source files named `validate-xxx.R` depending on the data source.

Usage

```
validate_data(rawData, ...)
```

Arguments

`rawData`	• the raw data from `load_data()`
`...`	not used / passed to the validation function specific to the type of data.

Value

the same input with a new `data_quality_failures` attribute containing issues.

`valid_inputs` *A valid set of types of file that can be loaded by* `load_data(...)`

Description

A valid set of types of file that can be loaded by `load_data(...)`

Usage

```
valid_inputs()
```

Value

a dataframe of type, subtype

Examples

```
# devtools::load_all()
try({
  avoncap::set_input("~/Data/avoncap")
  avoncap::input("nhs-extract")

  avoncap::all_files()


  # exact match on filename column of all_data()
  avoncap::most_recent_files("AvonCAPLRTDCentralDa")


  # or matches by lower case startWith on directory
  avoncap::most_recent_files("nhs-extract","deltave")


  avoncap::most_recent_files("metadata")

  avoncap::valid_inputs()
})
```

`write_issues` *Write out data quality issues*

Description

Write out data quality issues

Usage

```
write_issues(df, file)
```

Arguments

`df`	the raw data frame
`file`	the output data quality file

Value

the list of failures as a dataframe

`xglimpse` *Wrapper around* `table`

Description

Wrapper around `table`

Usage

```
xglimpse(data, ...)
```

Arguments

`data`	a dataframe
`...`	columns or named expressions to cross-tabulate

Value

the cross-tabulation

Index