Package ‘medicalrisk’

August 29, 2016

Type      Package
Title     Medical Risk and Comorbidity Tools for ICD-9-CM Data
Version   1.2
Date      2016-01-23

Description Generates risk estimates and comorbidity flags from ICD-9-CM codes available in administrative medical datasets. The package supports the Charlson Comorbidity Index, the Elixhauser Comorbidity classification, the Revised Cardiac Risk Index, and the Risk Stratification Index. Methods are table-based, fast, and use the 'plyr' package, so parallelization is possible for large jobs. Also includes a sample of real ICD-9 data for 100 patients from a publicly available dataset.

Depends   R (>= 3.0.0)
Imports   plyr (>= 1.5), reshape2, hash
Suggests  testthat, knitr, ggplot2, gridExtra
License   GPL-3 | file LICENSE
LazyData  true
VignetteBuilder knitr
RoxygenNote 5.0.1

BugReports https://github.com/patrickmdnet/medicalrisk/issues

URL https://github.com/patrickmdnet/medicalrisk

NeedsCompilation no

Author   Patrick McCormick [aut, cre],
          Thomas Joseph [aut]

Maintainer Patrick McCormick <patrick.mccormick@alum.mit.edu>
Repository CRAN
Date/Publication 2016-01-24 15:45:31
**R topics documented:**

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>charlson_list</td>
<td>2</td>
</tr>
<tr>
<td>charlson_weights</td>
<td>3</td>
</tr>
<tr>
<td>charlson_weights_orig</td>
<td>4</td>
</tr>
<tr>
<td>elixhauser_list</td>
<td>5</td>
</tr>
<tr>
<td>generate_charlson_index_df</td>
<td>5</td>
</tr>
<tr>
<td>generate_comorbidity_df</td>
<td>6</td>
</tr>
<tr>
<td>icd9cm_charlson_ddeo</td>
<td>7</td>
</tr>
<tr>
<td>icd9cm_charlson_quan</td>
<td>8</td>
</tr>
<tr>
<td>icd9cm_charlson_romano</td>
<td>9</td>
</tr>
<tr>
<td>icd9cm_elixhauser_abrq37</td>
<td>11</td>
</tr>
<tr>
<td>icd9cm_elixhauser_quan</td>
<td>12</td>
</tr>
<tr>
<td>icd9cm_list</td>
<td>13</td>
</tr>
<tr>
<td>icd9cm_rcri</td>
<td>14</td>
</tr>
<tr>
<td>icd9cm_sessler_rsi</td>
<td>15</td>
</tr>
<tr>
<td>melt_icd9list</td>
<td>16</td>
</tr>
<tr>
<td>merge_icd9_dx_and_procs</td>
<td>17</td>
</tr>
<tr>
<td>rsi_beta_1yrpod</td>
<td>17</td>
</tr>
<tr>
<td>rsi_beta_30dlos</td>
<td>18</td>
</tr>
<tr>
<td>rsi_beta_30dpod</td>
<td>18</td>
</tr>
<tr>
<td>rsi_beta_in hosp</td>
<td>19</td>
</tr>
<tr>
<td>rsi_sample_data</td>
<td>19</td>
</tr>
<tr>
<td>rsi_sample_results</td>
<td>20</td>
</tr>
<tr>
<td>sessler_get_single_beta</td>
<td>21</td>
</tr>
<tr>
<td>verify_sessler_rsi</td>
<td>22</td>
</tr>
<tr>
<td>vt_inp_sample</td>
<td>22</td>
</tr>
</tbody>
</table>

---

**Index**

| charlson_list | List of Charlson comorbidities |

**Description**

List of Charlson comorbidities

**Usage**

`charlson_list`

**Format**

A list, with one column for each comorbidity; value is a textual description
charlson_weights

Examples

# List the strings used to refer to Charlson comorbidities
names(charlson_list)

# List descriptions of comorbidities
charlson_list

charlson_weights  Map of Charlson comorbidity categories to revised weights

Description

List that links the Charlson comorbidity categories to revised weights as calculated by Schneeweiss in Table 4 of his paper.

Usage

charlson_weights

Format

A list, with Charlson comorbidities as names and weight as value

Details

Revised Schneeweiss weights:
0 = Connective tissue dz, Ulcer
1 = MI, PVD, CVD, Diabetes, Hemiplegia
2 = CHF, Chronic pulm dz, Mild liver dz, Diabetes with end organ damage, Any tumor, Leukemia, Lymphoma
3 = Dementia, Mod or severe renal dz
4 = Moderate or severe liver dz, AIDS
6 = Metastatic solid tumor

References


See Also

charlson_weights_orig, icd9cm_charlson_deyo, icd9cm_charlson_romano, icd9cm_charlson_quan, melt_icd9list

Examples

charlson_weights["dementia"]
charlson_weights_orig  Map of Charlson comorbidity categories to weights

Description
List that links the Charlson comorbidity categories to the original weights (specified in the original Charlson paper, Table 3)

Usage
charlson_weights_orig

Format
A list, with Charlson comorbidities as names and weight as value

Details
Original Weights:
1 = MI, CHF, PVD, CVD, Dementia, Chronic pulm dz, Connective tissue dz, Ulcer, Mild liver dz, Diabetes
2 = Hemiplegia, Mod or severe renal dz, Diabetes with end organ damage, Any tumor, Leukemia, Lymphoma
3 = Moderate or severe liver dz
6 = Metastatic solid tumor, AIDS

References
comorbidity in longitudinal studies: development and validation. Journal of chronic diseases 1987;

See Also
charlson_weights, icd9cm_charlson_deyo, icd9cm_charlson_romano, icd9cm_charlson_quan,
melt_icd9list

Examples
charlson_weights_orig["aids"]
elixhauser_list

**Description**

List of Elixhauser comorbidities

**Usage**

elixhauser_list

**Format**

A list, with one column for each comorbidity; value is a textual description

**Examples**

```r
# List the strings used to refer to Elixhauser comorbidities
names(elixhauser_list)

# List descriptions of comorbidities
elixhauser_list
```

---

**generate_charlson_index_df**

*Calculate the Charlson Comorbidity Index*

**Description**

`generate_charlson_index_df` merges a data frame of Charlson comorbidities with `charlson_weights` and sums the results per patient.

**Usage**

```r
generate_charlson_index_df(df, idvar = "id",
                           weights = medicalrisk::charlson_weights)
```

**Arguments**

- `df`: a data frame with ID column `idvar` and logical columns for each comorbidity, such as that generated by `generate_comorbidity_df`
- `idvar`: string with name of ID variable within `df`
- `weights`: defaults to `charlson_weights`
generate_comorbidity_df

Value

a dataframe with two columns, idvar and "index"

See Also

generate_comorbidity_df, charlson_weights, charlson_weights_orig

Examples

# calculate Charlson Comorbidity Index for all patients in the \code{\link{vt_inp_sample}}
data(vt_inp_sample)
generate_charlson_index_df(generate_comorbidity_df(vt_inp_sample))

Description

Merges a given DF of IDs and ICD-9-CM codes to one of the ICD9CM maps, removes redundant comorbidities, and returns a dataframe.

Usage

generate_comorbidity_df(df, idvar = "id", icd9var = "icd9cm",
icd9mapfn = icd9cm_charlson_quan, .progress = "none", .parallel = FALSE,
icd9mapfn = NULL)

Arguments

df a data frame with at least two columns, specified as idvar and icd9var.
idvar string with name of ID variable within df (defaults to "id")
icd9var string with name of ICD code variable within df (defaults to icd9cm)
icd9mapfn Function to generate comorbidity data frame from ICD-9 codes (defaults to icd9cm_charlson_quan)
.progress passed to dply
.parallel passed to dply
.paropts passed to dply
**Details**

Redundancy rules: * If "tumor" and "mets", only "mets" will be returned. * If "htn" and "htncx", only "htncx" will be returned. * If "dm" and "dmcx", only "dmcx" will be returned. * If "liver" and "modliver", only "modliver" will be returned.

Van Walraven has a modification adopted here where the following "dmcx" codes are downgraded to "dm" if the specific DM complication is separately coded: * D2(49|50)4x is DM w renal * D2(49|50)6x is DM w neuro * D2(49|50)7x is DM w PVD

Cases without any comorbidities will not appear in the returned data frame.

**Value**

a dataframe with column idvar and a logical column for each comorbidity

**Examples**

cases <- data.frame(id=c(1,1,1,2,2,2,2),
                    icd9cm=c("D20206","D24220","D4439","D5064","DE8788","D40403","D1960","D1958"))
generate_comorbidity_df(cases)
# generate categories for patients in the \code{\link{vt_inp_sample}}
generate_comorbidity_df(vt_inp_sample)
# in this example, D25071 is reduced to "dm" from "dmcx" because D4439 already codes perivasc
# also, D20206 "tumor" and D1970 "mets" lead to just "mets"
# D25001 and D25040 are just "dmcx"
# D45621 and D570 are just "modliver"
cases <- data.frame(id=c(1,1,1,2,2,2),
                    icd9cm=c("D1970","D20206","D25071","D4439","D25001","D25040","D45621","D570"))
generate_comorbidity_df(cases)

---

**icd9cm_charlson_deyo**  
Create Deyo map of ICD-9-CM to Charlson comorbidities

**Description**

Function that generates a data frame linking ICD-9-CM codes to the Charlson comorbidity categories using the Deyo mapping.

**Usage**

icd9cm_charlson_deyo(icd9)

**Arguments**

icd9  
a unique character vector of ICD-9-CM codes
icd9cm_charlson_quan

Details

NOTE: The input vector of ICD-9-CM codes must be unique, because the output dataframe uses the ICD-9-CM code as row.name.

Uses regular expressions created from the paper by Deyo in 1992.

ICD-9-CM codes must have periods removed. Diagnostic codes are prefixed with 'D' while procedure codes are prefixed with 'P'. So, diagnostic code 404.03 should be "D40403".

Value

A data frame, with ICD9 codes as row names and one logical column for each comorbidity in charlson_list

References


See Also

icd9cm_charlson_quan, icd9cm_charlson_romano, icd9cm_elixhauser_quan, icd9cm_elixhauser_ahrq37, charlson_weights,

Examples

# Identify Charlson categories in ICD-9-CM listing
cases <- data.frame(id=c(1,1,2,2,2),
icd9cm=c("D20206","D24220","D4439","D5064","DE8788","D40403"))
cases_with_cm <- merge(cases, icd9cm_charlson_deyo(levels(cases$id9cm)),
by.x="id9cm", by.y="row.names", all.x=TRUE)

# generate crude comorbidity summary for each patient
library(plyr)
ddply(cases_with_cm, .(id),
function(x) { data.frame(lapply(x[3:ncol(x)], any) )})

icd9cm_charlson_quan Create Quan map of ICD-9-CM to Charlson comorbidities

Description

Function that creates a dataframe that links ICD-9-CM codes to the Charlson comorbidity categories using Quan’s method.

Usage

icd9cm_charlson_quan(icd9)
Arguments

icd9 a unique character vector of ICD-9-CM codes

Details

NOTE: The input vector of ICD-9-CM codes must be unique, because the output dataframe uses the ICD-9-CM code as row.name.

Uses regular expressions created from the paper by Quan in 2005.

ICD-9-CM codes must have periods removed. Diagnostic codes are prefixed with 'D' while procedure codes are prefixed with 'P'. So, diagnostic code 404.03 should be "D40403".

Value

A data frame, with ICD9 codes as row names and one logical column for each comorbidity in charlson_list

References


See Also

icd9cm_charlson_deyo, icd9cm_charlson_romano, icd9cm_charlson_quan, icd9cm_elixhauser_quan

Examples

# Identify Charlson categories in ICD-9-CM listing
cases <- data.frame(id=c(1,1,1,2,2,2),
icd9cm=c("D20206","D24220","D4439","D5064","DE8788","D40403"))
cases_with_cm <- merge(cases, icd9cm_charlson_quan(levels(cases$id9cm)),
  by.x="id9cm", by.y="row.names", all.x=TRUE)

# generate crude comorbidity summary for each patient
library(plyr)
ddply(cases_with_cm, .(id),
  function(x) { data.frame(lapply(x[,3:ncol(x)], any)) })

icd9cm_charlson_romano

Create Romano map of ICD-9-CM to Charlson comorbidities

Description

Function that creates a dataframe which links ICD-9-CM codes to the Charlson comorbidity categories using the Romano mapping.
Usage

icd9cm_charlson_romano(icd9)

Arguments

icd9  a unique character vector of ICD-9-CM codes

Details

NOTE: The input vector of ICD-9-CM codes must be unique, because the output dataframe uses the ICD-9-CM code as row.name.

Uses regular expressions created from the paper by Romano in 1993.

ICD-9-CM codes must have periods removed. Diagnostic codes are prefixed with 'D' while procedure codes are prefixed with 'P'. So, diagnostic code 404.03 should be "D40403".

Value

A data frame, with ICD9 codes as row names and one logical column for each comorbidity in charlson_list

References


See Also

icd9cm_charlson_quan, icd9cm_charlson_deyo, icd9cm_elixhauser_quan, icd9cm_elixhauser_ahrq37, charlson_weights.

Examples

# Identify Charlson categories in ICD-9-CM listing
cases <- data.frame(id=c(1,1,1,2,2,2),
    icd9cm=c("D20206","D24220","D4439","D5064","DE8788","D40403"))
cases_with_cm <- merge(cases, icd9cm_charlson_romano(levels(cases$icd9cm)),
    by.x="icd9cm", by.y="row.names", all.x=TRUE)

# generate crude comorbidity summary for each patient
library(plyr)
ddpply(cases_with_cm,.(id),
    function(x) { data.frame(lapply(x[,3:ncol(x)], any)) })
icd9cm_elixhauser_ahrq37

Create AHRQ v3.7 map of ICD-9-CM to Elixhauser comorbidities

Description

Function makes a dataframe that links ICD-9-CM codes to the Elixhauser comorbidity categories using the AHRQ v3.7 mapping.

Usage

icd9cm_elixhauser_ahrq37(icd9)

Arguments

icd9 a unique character vector of ICD-9-CM codes

Details

Uses regular expressions based on the file "comformat2012-2013.txt" from AHRQ.

The Agency for Healthcare Research and Quality (AHRQ) has developed Comorbidity Software as part of the Healthcare Cost and Utilization Project (HCUP). The software was developed to report on the comorbidity measures reported by Elixhauser (1998).

The AHRQ software has two parts, one that classifies ICD-9-CM codes by comorbidity, and another that performs heuristics to eliminate duplicate comorbidities and ignore comorbidities which are the primary reason for the hospital visit, as per the DRG.

This table is a translation of the first part of the software, the classifier, as implemented in the SAS file Comformat2012-2013.txt.

ICD-9-CM codes must have periods removed. Diagnostic codes are prefixed with 'D' while procedure codes are prefixed with 'P'. So, diagnostic code 404.03 should be "D40403".

Value

A data frame, with ICD9 codes as row names and one logical column for each comorbidity in elixhauser_list

References


See Also

icd9cm_charlson_deyo, icd9cm_charlson_romano, icd9cm_charlson_quan, icd9cm_elixhauser_quan
Examples

# Identify Elixhauser categories
cases <- data.frame(id=c(1,1,2,2,2),
                  icd9cm=c("D20206","D24220","D4439","D5064","DE8788","D40403"))
cases_with_cm <- merge(cases, icd9cm_elixhauser_ahrq37(levels(cases$icd9cm)),
                        by.x="icd9cm", by.y="row.names", all.x=TRUE)

# generate crude comorbidity summary for each patient
library(plyr)
ddply(cases_with_cm, .(id),
     function(x) { data.frame(lapply(x[3:ncol(x)], any) )})

icd9cm_elixhauser_quan

Create Quan map of ICD-9-CM to Elixhauser comorbidities

Description

Function to make a dataframe that links ICD-9-CM codes to the Elixhauser comorbidity categories using the Quan mapping.

Usage

icd9cm_elixhauser_quan(icd9)

Arguments

icd9 a unique character vector of ICD-9-CM codes

Details

Uses regular expressions created from the Quan paper from 2005.
ICD-9-CM codes must have periods removed. Diagnostic codes are prefixed with 'D' while procedure codes are prefixed with 'P'. So, diagnostic code 404.03 should be "D40403".
Some ICD-9-CM codes will correspond to more than one category. For example, 404.03 (Hypertensive heart and chronic kidney disease ... stage V) is in both chf and renfail categories.

Value

A data frame, with ICD9 codes as row names and one logical column for each comorbidity in elixhauser_list

References

### icd9cm_list

**List of ICD-9-CM diagnostic and procedural codes**

**Description**

ICD-9-CM codes have the periods removed. Diagnostic codes are prefixed with 'D' while procedure codes are prefixed with 'P'. So, diagnostic code 404.03 appears as "D40403".

**Usage**

icd9cm_list

**Format**

A string vector

**Details**

Obsolete codes not active in 2012 are not present, and may cause this dataset to miss certain classifications when applied to older datasets. For example, codes 043 and 044 (both obsolete AIDS codes) are not included.

**References**

1. [https://www.cms.gov/Medicare/Coding/ICD9ProviderDiagnosticCodes/codes.html](https://www.cms.gov/Medicare/Coding/ICD9ProviderDiagnosticCodes/codes.html)

**Examples**

```r
# Count procedural codes
length(icd9cm_list[grep('^P',icd9cm_list)])
```
Create Map of ICD-9-CM to Revised Cardiac Risk Index classes

Description

Function to generate data frame that links ICD-9-CM codes to the RCRI comorbidity categories.

Usage

icd9cm_rcri(icd9)

Arguments

icd9 a unique character vector of ICD-9-CM codes

Details

Lee et al in 1999 published a "Revised Cardiac Risk Index" based on the work on Goldman in 1997. The RCRI is used to determine the major cardiac complication risk for a patient about to undergo major noncardiac surgery. The six predictors that make up the RCRI are: 1. high-risk surgery, 2. history of ischemic heart disease, 3. history of congestive heart failure, 4. history of cerebrovascular disease, 5. preoperative treatment with insulin, and 6. preoperative serum creatinine with Cr > 2 mg/dL.

In 2005 Boersma et al demonstrated that the Lee indexed can be adapted to use administrative data to predict cardiovascular mortality. They used the following for each point above: 1. retroperitoneal, intrathoracic, or suprainguinal vascular procedure; 2. Ischemia: ICD-9 codes 410.*, 411.*, 412.*, 413.*, 414.*; 3. CVA: ICD-9 428.*; 4. CHF: ICD-9 943.0; 5. DM: ICD-9 425.0; 6. Renal: ICD-9 958.0.

This function merges the ICD-9 guidelines used by Boersma with some of the other ICD-9 classifiers in this package. This data set uses the following for each aspect of the RCRI: 1. procedure is left to you, 2. 'ischemia' as defined in Boersma, 3. 'cvd' as defined by Quan in icd9cm_charlson_quan, 4. 'chf' as defined by AHRQ in icd9cm_elixhauser_ahrqS7, 5. 'dm' as defined by AHRQ (both 'dm' and 'dmcx') 6. renlfail' as defined by AHRQ.

Value

A data frame, with ICD9 codes as row names and logical columns for chf, cvd, dm, ischemia, and renlfail.

References

See Also

icd9cm_charlson_quan, icd9cm_elixhauser_quan, icd9cm_elixhauser_ahrq37

---

`icd9cm_sessler_rsi`  
Returns composite Sessler risk stratification index, given a list of ICD-9-CM codes.

Description

ICD-9-CM codes must have periods removed. Diagnostic codes are prefixed with 'D' while procedure codes are prefixed with 'P'. So, diagnostic code 404.03 should be "D40403".

Usage

`icd9cm_sessler_rsi(icd9)`

Arguments

- `icd9` a unique character vector of ICD-9-CM codes

Details

Note: A subsequent publication (Sigakis, 2013) found the following: "Calibration "in-the-large" for RSI in-hospital mortality illustrated a discrepancy between actual (1.5) and a regression constant (-2.198) in the published RSI "all-covariates.xls" file that was not used in the published SPSS implementation macro."

Value

The risk stratification index score

Author(s)

Tom Joseph <thomas.joseph@mountsinai.org>, Patrick McCormick <patrick.mccormick@mountsinai.org>

References

Examples

```r
# Calculate RSI for each patient ("id") in dataframe
cases <- data.frame(id=c(1,1,2,2,2),
icd9cm=c("D20206","D24220","D4439","D5064","DE8788","D40403"))
library(plyr)
ddply(cases, .(id), function(x) { icd9cm_sessler_rsi(x$id$icd9cm) })
```

melt_icd9list  
Convert ICD-9-CM code list to dataframe

Description

melt_icd9list uses `ddply` to melt a column of comma-separated ICD-9-CM codes into a series of rows, one for each code.

Usage

```r
melt_icd9list(df, idvar = "id", icd9var = "icd9cm", .progress = "none",
.parallel = FALSE, .paropts = NULL)
```

Arguments

- `df`: a data frame with at least two columns, specified as `idvar` and `icd9var`.
- `idvar`: string with name of ID variable within `df` (defaults to "id")
- `icd9var`: string with name of ICD code variable within `df` (defaults to "icd9cm")
- `.progress`: passed to `ddply`
- `.parallel`: passed to `ddply`
- `.paropts`: passed to `ddply`

Value

a dataframe with two columns, `idvar` and "icd9cm"

Examples

```r
cases <- data.frame(id=c(1,2), icd9list=c('162.4,070.30,155.0,401.9','996.52,E878.8,V45.86'))
melt_icd9list(cases, "id", "icd9list")
```
### merge_icd9_dx_and_procs

**Merge ICD-9-CM diagnostic and procedure codes**

**Description**

Merges a dataframe containing ICD-9-CM diagnostic codes with a dataframe containing ICD-9 procedure codes. Diagnostic codes are prefixed with 'D', while procedure codes are prefixed with 'P'.

**Usage**

```r
merge_icd9_dx_and_procs(dx_df, proc_df, icd9dxvar = "icd9cm", icd9pvar = "icd9cm")
```

**Arguments**

- **dx_df**: a data frame with at least two columns, specified as `idvar` and `icd9dxvar`, where the values are ICD-9 diagnostic codes
- **proc_df**: a data frame with at least two columns, specified as `idvar` and `icd9pvar`, where the values are ICD-9 procedure codes
- **icd9dxvar**: name of icd9 diagnostic code column, default "icd9cm"
- **icd9pvar**: name of icd9 procedure code column, default "icd9cm"

**Value**

a merged dataframe with common columns and "icd9cm"

**Examples**

```r
cases <- data.frame(id=c(1,2),
                    icd9dxlist=c("162.4", "070.30", "155.0", "401.9", "996.52", "E878.8", "V45.86"),
                    icd9pplist=c("38.16", "38.42"))
dx_df <- melt_icd9list(cases, "id", "icd9dxlist")
proc_df <- melt_icd9list(cases, "id", "icd9pplist")
merge_icd9_dx_and_procs(dx_df, proc_df)
```

### rsi_beta_1yrpod

**Values for calculating RSI for 1 year mortality**

**Description**

Values for calculating RSI for 1 year mortality

**Usage**

```r
rsi_beta_1yrpod
```
Format

A hash (see package "hash"), where key is icd9cm code, and value is beta. Special key "popbeta" has the population beta for the entire table.

References


---

rsi_beta_30dlos

Values for calculating RSI for 30-day length of stay

Description

Values for calculating RSI for 30-day length of stay

Usage

rsi_beta_30dlos

Format

A hash (see package "hash"), where key is icd9cm code, and value is beta. Special key "popbeta" has the population beta for the entire table.

References


---

rsi_beta_30dpod

Values for calculating RSI for 30-day mortality

Description

Values for calculating RSI for 30-day mortality

Usage

rsi_beta_30dpod
Format

A hash (see package "hash"), where key is icd9cm code, and value is beta. Special key "popbeta" has the population beta for the entire table.

References


rsi_beta_inhosp

Values for calculating RSI for in-hospital mortality.

Description

Values for calculating RSI for in-hospital mortality.

Usage

rsi_beta_inhosp

Format

A hash (see package "hash"), where key is icd9cm code, and value is beta. Special key "popbeta" has the population beta for the entire table.

References


rsi_sample_data

Sample data for validating RSI

Description

Sample data for validating RSI

Usage

rsi_sample_data
rsi_sample_results

**Format**

A data table with a patient ID and several columns with ICD-9-CM codes.

**References**


**See Also**

rsi_sample_results, verify_sessler_rsi

---

rsi_sample_results  Sample results for validating RSI

**Description**

Sample results for validating RSI

**Usage**

rsi_sample_results

**Format**

A data table with a patient ID, principal diagnosis, principal procedure, and RSI result columns.

**References**


**See Also**

rsi_sample_data, verify_sessler_rsi
**sessler_get_single_beta**

Returns the covariate coefficient for a particular diagnosis or procedure code, along with the actual code that was found in the internal database of coefficients. If a child code is supplied but its parent is in the database, the coefficient for the parent will be returned, along with that parent code. For example, if D1231 is supplied but only D123 is available, D123 will be used. This is apparently how the SPSS sample code works.

**Description**

Returns the covariate coefficient for a particular diagnosis or procedure code, along with the actual code that was found in the internal database of coefficients. If a child code is supplied but its parent is in the database, the coefficient for the parent will be returned, along with that parent code. For example, if D1231 is supplied but only D123 is available, D123 will be used. This is apparently how the SPSS sample code works.

**Usage**

```r
sessler_get_single_beta(code, betalist)
```

**Arguments**

- `code` A single ICD-9-CM code
- `betalist` One of the rsi_beta_* datasets (supplied with this package)

**Value**

Covariate coefficient. You must sum all of these for a given patient and then subtract the appropriate population beta (e.g. rsi_beta_1yrpop$popbeta)

**Examples**

```r
# get coefficient for hypercholesterolemia
sessler_get_single_beta('D2720', rsi_beta_inhosp)
# Also works with extra 0 on the end
sessler_get_single_beta('D27200', rsi_beta_inhosp)
```
**verify_sessler_rsi**  Validates this Sessler RSI implementation against reference data

**Description**
Requires that "sample data rev2.csv" and "sample results rev2.csv" be available in datasrc directory

**Usage**
```r
verify_sessler_rsi()
```

**Value**
Table of patients with scores >0.001 difference.

**Author(s)**
Patrick McCormick <patrick.mccormick@mountsinai.org>

**References**

**See Also**
rsi_sample_data, rsi_sample_results

**Examples**
```r
## Not run:
verify_sessler_rsi()

## End(Not run)
```

**vt_inp_sample**  First 100 patients and their ICD-9-CM codes from the Vermont Uniform Hospital Discharge Data Set for 2011, Inpatient.

**Description**
Diagnostic ICD-9 codes are prefixed with 'D', while procedural ICD-9 codes are prefixed with 'P'.

**Usage**
```r
vt_inp_sample
```
vt_inp_sample

Format

A data frame, with column "id" (numeric), some descriptive columns, "dx" (factor), and "icd9cm" (factor)

References

http://healthvermont.gov/research/hospital-utilization/RECENT PU_FILES.aspx

See Also

icd9cm_charlson_deyo, icd9cm_charlson_quan, icd9cm_charlson_romano, icd9cm_elixhauser_quan, icd9cm_elixhauser_ahrq37

Examples

max(vt_inp_sample$scu_days)
Index

*Topic **datasets**
- charlson_list, 2
- charlson_weights, 3
- charlson_weights_orig, 4
- elixhauser_list, 5
- icd9cm_list, 13
- vt_inp_sample, 22

charlson_list, 2, 8–10
charlson_weights, 3, 4–6, 8, 10
charlson_weights_orig, 3, 4, 6
dply, 6, 16
elixhauser_list, 5, 11, 12
generate_charlson_index_df, 5
generate_comorbidity_df, 5, 6, 6

icd9cm_charlson_deyo, 3, 4, 7, 9–11, 13, 23
icd9cm_charlson_quan, 3, 4, 6, 8, 8, 9–11, 13–15, 23
icd9cm_charlson_romano, 3, 4, 8, 9, 9, 11, 13, 23
icd9cm_elixhauser_ahrq37, 8, 10, 11, 13–15, 23
icd9cm_elixhauser_quan, 8–11, 12, 15, 23
icd9cm_list, 13
icd9cm_rcri, 14
icd9cm_sessler_rsi, 15

melt_icd9list, 3, 4, 16
merge_icd9_dx_and_procs, 17

rsi_beta_lyrpod, 17
rsi_beta_30dlos, 18
rsi_beta_30dpod, 18
rsi_beta_inhosp, 19
rsi_sample_data, 19, 20
rsi_sample_results, 20, 20

sessler_get_single_beta, 21
verify_sessler_rsi, 20, 22
vt_inp_sample, 22