Package: preventr (via r-universe)

September 17, 2024

Title An Implementation of the AHA PREVENT Equations

Version 0.10.0

Description Implements the American Heart Association (AHA) Predicting Risk of cardiovascular disease EVENTs (PREVENT) equations from Khan SS, Matsushita K, Sang Y, and colleagues (2024) <doi:10.1161/CIRCULATIONAHA.123.067626>.

License MIT + file LICENSE

URL `https://martingmayer.com/preventr`, `https://github.com/martingmayer/preventr`, `https://martingmayer.shinyapps.io/prevent-equations/`

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Depends R (>= 3.5.0)

LazyData true

Imports dplyr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

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

RemoteUrl https://github.com/martingmayer/preventr

RemoteRef HEAD

RemoteSha d2f9167f776ddac19d9be478bbc5a77d6b2a1432

Contents

app — *Navigate to browser-based Shiny implementation of PREVENT equations*

Description

This function opens a browser window (using the user's default browser) and navigates to the Shiny app located at:

https://martingmayer.shinyapps.io/prevent-equations

Easier-to-remember URLs:

- https://tiny.cc/prevent-equations
- https://tiny.cc/preventequations

The app includes risk visualization and several options for customizing the output.

Usage

```
app(...)
```

Arguments

`...`	Not used. Reserved for future use.

Value

Returns `NULL` invisibly after opening app in your default browser.

Examples

```
app()
```

estimate_risk — *Estimate risk of cardiovascular events using the American Heart Association (AHA) Predicting Risk of cardiovascular disease EVENTs (PREVENT) equations.*

Description

`estimate_risk()` and `est_risk()` are the same function, with the latter being a literal copy of the former just for those who favor syntactical brevity (i.e., a function synonym).

Estimation includes both 10- and 30-year risk of 5 events:

- Total cardiovascular disease (CVD)
  - This outcome includes atherosclerotic CVD (ASCVD) and heart failure as defined below
- ASCVD
  - This outcome includes coronary heart disease (CHD) and stroke as defined below
- Heart failure (often abbreviated HF, but not herein)
- CHD
  - This outcome includes nonfatal myocardial infarction (MI) and fatal CHD
- Stroke

See also the README for this package, which goes into additional detail about the PREVENT equations (site, GitHub).

Usage

```
estimate_risk(
  age,
  sex,
  sbp,
  bp_tx,
  total_c,
  hdl_c,
  statin,
  dm,
  smoking,
  egfr,
  bmi,
  hba1c = NULL,
  uacr = NULL,
  zip = NULL,
  model = NULL,
  time = "both",
  chol_unit = "mg/dL",
  optional_strict = FALSE,
  quiet = FALSE
)

est_risk(
  age,
  sex,
  sbp,
  bp_tx,
  total_c,
  hdl_c,
  statin,
  dm,
  smoking,
  egfr,
  bmi,
  hba1c = NULL,
  uacr = NULL,
  zip = NULL,
  model = NULL,
  time = "both",
  chol_unit = "mg/dL",
  optional_strict = FALSE,
  quiet = FALSE
)
```

Arguments

`age` Numeric (required predictor variable): Age in years, from 30-79

`sex` Character (required predictor variable): Either `"female"` or `"male"` (`"f"` and `"m"` are accepted abbreviations)

`sbp` Numeric (required predictor variable): Systolic blood pressure (SBP) in mmHg, from 90-180; see the "Details" section for more information about the upper bound of the range

`bp_tx` Logical or numeric equivalent (required predictor variable): Whether the person is on blood pressure treatment, either `TRUE` or `FALSE` (1 or 0 are accepted as alternative input)

`total_c` Numeric (required predictor variable): Total cholesterol in mg/dL or mmol/L (see `chol_unit` argument), from 130-320 (for `chol_unit = "mg/dL"`) or 3.36-8.28 (for `chol_unit = "mmol/L"`)

`hdl_c` Numeric (required predictor variable): High-density lipoprotein cholesterol (HDL-C) in mg/dL or mmol/L (see `chol_unit` argument), from 20-100 (for `chol_unit = "mg/dL"`) or 0.52-2.59 (for `chol_unit = "mmol/L"`)

`statin` Logical or numeric equivalent (required predictor variable): Whether the person is taking a statin, either `TRUE` or `FALSE` (1 or 0 are accepted as alternative input)

`dm` Logical or numeric equivalent (required predictor variable): Whether the person has diabetes mellitus (DM), either `TRUE` or `FALSE` (1 or 0 are accepted as alternative input)

`smoking` Logical or numeric equivalent (required predictor variable): Whether the person is currently smoking (which PREVENT defines as cigarette use within the last 30 days), either `TRUE` or `FALSE` (1 or 0 are accepted as alternative input)

`egfr` Numeric or call (required predictor variable): Estimated glomerular filtration rate (eGFR) in mL/min/1.73m2, entered either as a numeric from 15-140 or as a call to `calc_egfr()` or synonyms, as described in the "Details" section `bmi` Numeric or call (required predictor variable): Body mass index (BMI) in kg/m2, entered either as a numeric from 18.5-39.9 or as a call to `calc_bmi()` or synonyms, as described in the "Details" section

`hba1c` Numeric (optional predictor variable): Glycated hemoglobin (HbA1c) in %, from 4.5-15; see the "Details" section for more information about the lower bound of the range

`uacr` Numeric (optional predictor variable): Urine albumin-to-creatinine ratio (UACR) in mg/g, from 0.1-25000

`zip` Character (optional predictor variable): ZIP code of the person's residence, used to estimate the Social Deprivation Index (SDI); see the "Details" section for more information

`model` Character (required, but has default): The PREVENT model to use, one of `NULL` (the default), `"base"` (the base model), `"hba1c"` (the base model adding HbA1c), `"uacr"` (the base model adding UACR), `"sdi"` (the base model adding SDI), or `"full"` (the base model adding HbA1c, UACR, and SDI). If `NULL`, the model will be determined by algorithm specified in the "Details" section, and this is the intended argument for most users. The ability to specify mainly exists for specific use cases (e.g., research purposes).

`time` Character or numeric (required, but has default): Whether to estimate risk over 10 or 30 years, one of `"both"` (character; the default); `10` (numeric), `"10"` (character), or `"10yr"` (character); or `30` (numeric), `"30"` (character), or `"30yr"` (character); if estimating over 30 years and age > 59, a warning will accompany the results regarding the reliability of the estimation (see the "Value" section for more information)

`chol_unit` Character (required, but has default): The unit of measurement for `total_c` and `hdl_c`, either `"mg/dL"` (the default) or `"mmol/L"` (`"mg"` and `"mmol"` are accepted abbreviations)

`optional_strict`

Logical (required, but has default): Whether to enforce strictness on optional predictor variables, either `TRUE` or `FALSE` (the default). The argument itself is strict, so 1 or 0 are *not* accepted (in contrast with some of the other logical inputs considered by this function), and moreover, anything other than `TRUE` will be treated as `FALSE`. If `FALSE`, the function will discard invalid optional predictor variables but still allow the model to run. If `TRUE`, optional predictor variables entered (if any) must be valid for the function to return risk estimates. See the "Value" section for more information.

`quiet` Logical (required, but has default): Whether to suppress messages and warnings in the console, either `TRUE` or `FALSE` (the default); this argument is strict, so 1 or 0 are *not* accepted (in contrast with some of the other logical inputs considered by this function), and moreover, anything other than `TRUE` will be treated as `FALSE`

Details

Why is the upper limit of the SBP range 180 mmHg?:

Some may notice the upper limit is set to 180 mmHg here, whereas the PREVENT equations technically permit up to 200 mmHg. The Pooled Cohort Equations (PCEs) do this as well. I have restricted to 180 mmHg, as SBP beyond 180 mmHg constitutes hypertensive urgency (per AHA's own definitions), and irrespective of the debate surrounding labels like hypertensive urgency and emergency, it would seem clinically unreasonable to engage with the PREVENT equations when someone has more pressing matters to address (better blood pressure control *per se*).

Why is the lower limit of the HbA1c 4.5%?:

Some may notice the lower limit is set to 4.5% here, whereas the PREVENT equations technically permit down to 3%. I have restricted to 4.5%, as HbA1c of 3% is neither realistic nor safe for a person. For example, using the HbA1c to estimated average glucose (eAG) converter from the American Diabetes Association (https://professional.diabetes.org/glucose_calc), a HbA1c of 3% corresponds to an eAG of 39 mg/dL (2.2 mmol/L).

Entering eGFR and BMI as a call rather than a numeric value:

The `eGFR` and `bmi` arguments can be entered as numeric values or as calls to `calc_egfr()` and `calc_bmi()`, respectively. They both have synonyms as well:

- Synonyms for `calc_egfr()` are `calculate_egfr()`, `calc_ckd_epi()`, and `calculate_ckd_epi()`, with the latter two synonyms reflecting the calculation is from the CKD-EPI equations (the reparameterized version without race, which is also what the PREVENT equations use).
- The synonym for `calc_bmi()` is `calculate_bmi()`.

These convenience functions add value where a person might have the necessary components to calculate the respective parameter but do not have handy the parameter itself.

Although these convenience functions were, of course, tested for accuracy, they were written and tested within the context of supporting the PREVENT equations as implemented within this package. As such, they are not exported, and users will be warned to proceed with caution if they try to use these functions outside of `estimate_risk()` or its synonym `est_risk()`.

The syntax for these convenience functions is as follows:

- `calc_egfr(cr, units = "mg/dL", age, sex, quiet = FALSE)`
  - `cr` is the creatinine in whatever units are specified by `units`
  - `units` is the unit of measurement for `cr`, either `"mg/dL"` or `"umol/L"`, with `"mg"` and `"umol"` being accepted abbreviations
  - `age` is the age of the person, but there is no need to enter this, as the function will extract this from the `age` argument of `estimate_risk()`; in fact, any argument entered here will be ignored in favor of the `age` argument of `estimate_risk()`
  - `sex` is the sex of the person, but there is no need to enter this, as the function will extract this from the `sex` argument of `estimate_risk()`; in fact, any argument entered here will be ignored in favor of the `sex` argument of `estimate_risk()`
  - `quiet` is a logical indicating whether to suppress the warning about use outside of `estimate_risk()`
  - An example call would be `calc_egfr(1.2)` (because `units` defaults to `"mg/dL"`) or `calc_egfr(88, "umol")`
- `calc_bmi(weight, height, units = "nonmetric", quiet = FALSE)`
  - `weight` is the weight in pounds if `units = "nonmetric"` or kilograms if `units = "metric"`
  - `height` is the height in inches if `units = "nonmetric"` or centimeters if `units = "metric"`
  - `units` is the unit of measurement for `weight` and `height`, either `"nonmetric"` or `"metric"`
  - `quiet` is a logical indicating whether to suppress the warning about use outside of `estimate_risk()`

  - An example call would be `calc_bmi(150, 70)` (because `units` defaults to `"nonmetric"`) or `calc_bmi(68, 178, "metric")`

What is the Social Deprivation Index (SDI)?:

Read more from the Robert Graham Center's page on the SDI (https://www.graham-center.org/maps-data-tools/social-deprivation-index.html)

Model selection when `model = NULL`:

If `model = NULL`, the model will be determined by the following algorithm:

- If no optional predictor variables (HbA1c, UACR, zip code) are entered, or only invalid optional variables are entered and `optional_strict = FALSE`: The base model
- If one of the optional predictor variables is entered, or two or more optional predictor variables are entered but only one is valid and `optional_strict = FALSE`: The base model adding that variable (e.g., if HbA1c is entered and no other optional predictor variables are entered, the base model adding HbA1c; if HbA1c and UACR are entered, but HbA1c is invalid and `optional_strict = FALSE`, the base model adding UACR)
- If two or more of the optional predictor variables are entered, or all three optional variables are entered but one is invalid and `optional_strict = FALSE`: The full model (the PREVENT equations include a term for optional predictor variables being missing, so if one of the optional predictor variables is missing in this scenario, it is treated as such within the full model)

What if SDI is not available for a zip code?:

Some zip codes do not have SDI data available, and the PREVENT equations include a term for SDI being missing. As such, if a user enters a valid zip code but no SDI data are available, the user will be notified, and the tool will then implement the missing term as part of predicting risk whenever the full model is used, but SDI will otherwise be removed from prediction. Specifically, the following models will predict risk in the situation where the user enters a valid zip code, but no SDI data are available:

- If the user does not enter a valid HbA1c or UACR: The base model
- If the user enters valid HbA1c and UACR: The full model (treating SDI as missing)
- If the user enters a valid HbA1c: The base model adding HbA1c
- If the user enters a valid UACR: The base model adding UACR

Value

`estimate_risk()` will always return a data frame as a tibble, and all references herein to a data frame being returned are for a data frame as a tibble (see the tibble package for more detail). However, the manner in which the data frame is returned will come in one of two ways, depending on the `time` argument:

- When `time = "both"`: A list of length 2, with each item in the list being a data frame containing the 10-year and 30-year estimates, in that order
- Otherwise: A single data frame containing the risk estimate for the specified time horizon

The data frame will have the following columns:

- `total_cvd`: The estimated risk of a total CVD event (column type: double)
- `ascvd`: The estimated risk of an ASCVD event (column type: double)

- heart_failure: The estimated risk of a HF event (column type: double)
- chd: The estimated risk of a CHD event (column type: double)
- stroke: The estimated risk of a stroke event (column type: double)
- model: The PREVENT model used (column type: character)
- over_years: The time horizon for the risk estimate (column type: integer)
- input_problems: Semicolon-separated vector of length one delineating input problems, if any exist; otherwise, NA_character_ (column type: character)

When valid input parameters exist for all required predictor variables:

The risk estimate columns are all of type double, and they are presented as a proportion rounded to 3 decimal places. Halves are rounded up to align with what many people likely expect, but this is in contrast to base R's default rounding behavior (it is a perfectly reasonable default, but perhaps somewhat unexpected for people who are not familiar with different standards/conventions for rounding; see round() for further detail).

The model column will be of type character, taking one of the following values: "base", "hba1c", "uacr", "sdi", or "full".

The over_years column will be of type integer, either 10 or 30.

If optional_strict = TRUE, the above will only hold if the optional predictor variables that are entered (if any) are valid; if any optional variables are entered but are invalid, the function will behave in the same manner as when invalid input parameters exist for one or more required variables.

When invalid input parameters exist for one or more required predictor variable(s):

The function will issue a warning about the problematic variables, unless quiet = FALSE. A data frame will be returned with the following characteristics:

- All risk estimates will be set to NA_real_
- The model column will state "none"
- The over_years column will be set to NA_integer_
- The input_problems column will contain a character vector of length 1 delineating the problematic variable(s); if multiple problematic variables exist, they will be separated by semicolons

When invalid input parameters exist for one or more optional predictor variable(s):

When optional_strict = TRUE*:*
The function will behave similarly to when invalid input parameters exist for one or more required variables, with the input_problems column delineating the problematic variables

When optional_strict = FALSE*:*
The function will issue a warning about the problematic variables, unless quiet = FALSE. The problematic optional variables will then be functionally discarded and the PREVENT equations still run, in accordance with the specifications detailed in the "Details" section regarding model selection. A data frame will be returned with the following characteristics:

- All estimates will be returned as specified in the valid input parameters section, as will the model and over_years columns

  - The input_problems column will contain a character vector of length 1 delineating the problematic variables (because optional predictor variables are allowed to be empty, any input that is functionally empty or missing (such as NULL, numeric(0), NA, etc.) will not be considered problematic and thus not populate in the input_problems column)

When estimating 30-year risk and age > 59:

The function advises 30-year risk prediction for people > 59 years is questionable via two warnings:

- in the console (that can be suppressed by setting quiet = TRUE)
- in the column input_problems of the return tibble (quiet has no impact here)

The special case of the zip argument:

The above rule for optional predictor variables applies to the zip argument as well, but with the additional reminder that there are valid zip codes that do not have an SDI score. This is importantly different from an invalid input for zip. See the "Details" section for more information about how this is handled, but users should *not* expect anything to populate in the input_problems column if the zip is valid, regardless of whether that zip has an SDI score. As will be clear from the "Details" section, users will be able to determine when a zip code does not have an SDI score based on the model that was used.

Combining output into a single data frame:

The output when time = "both" is a list of data frames, one for each time horizon, but if desired, it is easy to combine these into a single data frame, e.g.:

```
res_base_r <- do.call(rbind, res)        # Combine in base R
res_dplyr <- dplyr::bind_rows(res)       # Combine in dplyr
res_dt <- data.table::rbindlist(res)     # Combine in data.table

# These all yield the same tabular output, but the attributes vary
# (e.g., base R adds row names)

all.equal(res_base_r, res_dplyr, check.attributes = FALSE)   # TRUE
all.equal(res_dplyr, res_dt, check.attributes = FALSE)       # TRUE
```

Examples

```
# Example with all required predictor variables (example from Table S25
# in the supplemental PDF appendix of the PREVENT equations article)
#
# Optional predictor variables are all omitted (and thus take their default)
# `model` is also omitted (and thus takes its default, with the function selecting
# the model based on the algorithm specified in the "Details" section)
# `time` is also omitted (and thus takes its default, with the function returning
# estimates for both 10- and 30-year risk as specified in the "Value" section)
#
# Expect the base model to run given absence of optional predictor variables.
res <- estimate_risk(
  age = 50,
  sex = "female",    # or "f"
  sbp = 160,
  bp_tx = TRUE,      # or 1
  total_c = 200,     # default unit is "mg/dL"
  hdl_c = 45,        # default unit is "mg/dL"
  statin = FALSE,    # or 0
  dm = TRUE,         # or 1
  smoking = FALSE,   # or 0
  egfr = 90,
  bmi = 35
)

# Based on Table S25, expect the 10-year risk for `total_cvd` to be 0.147.
# Based on the supplemental Excel file, also expect:
# 10-year risks: `ascvd`, 0.092; `heart_failure`, 0.081;
# `chd`, 0.044; `stroke`, 0.054
# 30-year risks: `total_cvd`, 0.53; `ascvd`, 0.354; `heart_failure`, 0.39;
# `chd`, 0.198; `stroke`, 0.221
res

# Example with HbA1c
# (also changing required predictor variables & limiting to 10-year results)
estimate_risk(
  age = 66,
  sex = "male",      # or "m"
  sbp = 148,
  bp_tx = FALSE,
  total_c = 188,
  hdl_c = 52,
  statin = TRUE,
  dm = TRUE,
  smoking = TRUE,
  egfr = 67,
  bmi = 30,
  hba1c = 7.5,
  time = "10yr"      # only 10-year results will show
)

# Example with UACR (limited to 30-year results)
estimate_risk(
  age = 66,
  sex = "female",
  sbp = 148,
  bp_tx = FALSE,
  total_c = 188,
  hdl_c = 52,
  statin = TRUE,
  dm = TRUE,
  smoking = TRUE,
  egfr = 67,
  bmi = 30,
  uacr = 750,
  time = "30yr"      # only 30-year results will show
)
```

```
# The remaining examples will all be limited to 10-year results

# Example with SDI with valid zip code with SDI data available
estimate_risk(
  age = 66,
  sex = "female",
  sbp = 148,
  bp_tx = FALSE,
  total_c = 188,
  hdl_c = 52,
  statin = TRUE,
  dm = TRUE,
  smoking = TRUE,
  egfr = 67,
  bmi = 30,
  zip = "59043",   # Lame Deer, MT (selected randomly)
  time = 10        # Note use of numeric 10 here (not "10yr")
)

# Example with SDI with valid zip code without SDI data available
# (base model will be used)
estimate_risk(
  age = 66,
  sex = "male",
  sbp = 148,
  bp_tx = FALSE,
  total_c = 188,
  hdl_c = 52,
  statin = TRUE,
  dm = TRUE,
  smoking = TRUE,
  egfr = 67,
  bmi = 30,
  zip = "00738",   # Fajardo, PR
  time = 10
)

# Example with full model (even though zip does not have available SDI, full
# model used given availability of HbA1c and UACR; because zip is valid,
# column `input_problems` will be `NA`)
estimate_risk(
  age = 66,
  sex = "female",
  sbp = 148,
  bp_tx = FALSE,
  total_c = 188,
  hdl_c = 52,
  statin = TRUE,
  dm = TRUE,
  smoking = TRUE,
  egfr = 67,
  bmi = 30,
  hba1c = 9,
  uacr = 75,
  zip = "00738",
  time = "10yr"
)

# Example with full model (zip has SDI data available, UACR is valid, but
# HbA1c is not; column `input_problems` will specify problem with `hba1c`,
# but full model will still run given availability of the other two optional
# predictor variables)
estimate_risk(
  age = 66,
  sex = "male",
  sbp = 148,
  bp_tx = FALSE,
  total_c = 188,
  hdl_c = 52,
  statin = TRUE,
  dm = TRUE,
  smoking = TRUE,
  egfr = 67,
  bmi = 30,
  hba1c = 20,
  uacr = 75,
  zip = "59043",
  time = "10yr"
)

# Expect table of `NA`s due to invalid input for `age` and `sbp`, and column
# `input_problems` to contain explanations about problems with `age` and `sbp`
res <- estimate_risk(
  age = 8675309,
  sex = "female",
  sbp = 112358,
  bp_tx = TRUE,
  total_c = 200,
  hdl_c = 45,
  statin = FALSE,
  dm = TRUE,
  smoking = FALSE,
  egfr = 90,
  bmi = 35,
  time = "10yr"
)

res

# Quiet version of the above example
res <- estimate_risk(
  age = 8675309,
  sex = "female",
  sbp = 112358,
  bp_tx = TRUE,
  total_c = 200,
  hdl_c = 45,
  statin = FALSE,
  dm = TRUE,
  smoking = FALSE,
  egfr = 90,
  bmi = 35,
  time = "10yr",
  quiet = TRUE       # Suppresses messages, but not column `input_problems`
)

res

# Note `input_problems` column is semicolon-separated, but it is easy to
# print as separate lines with `gsub()` and `cat()`, e.g.:
cat(gsub("; ", "\n", res$input_problems))

res$input_problems |> gsub(pattern = "; ", replacement = "\n", x = _) |> cat()
# ... and could, of course, also do with the {magrittr} pipe `%>%` if desired
```

Index