Package: QualityMeasure (via r-universe)

November 7, 2026

Title Methods for Analyzing Quality Measure Performance

Version 2.0.2

Date 2026-07-09

Description Quality of care is compared across accountable entities, including hospitals, provider groups, and insurance plans, using standardized quality measures. However, observed variations in quality measure performance might be the result of chance sampling or measurement errors. Contains functions for estimating the reliability of unadjusted and risk-standardized quality measures.

License GPL (>= 3)

Depends R (>= 4.0), ggplot2, doParallel, lme4

Imports foreach, parallel, rlang

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Encoding UTF-8

LazyData true

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Kenneth Nieser [aut, cre, cph]

Maintainer Kenneth Nieser <nieser@stanford.edu>

Config/pak/sysreqs cmake make

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

Date/Publication 2026-07-09 21:40:02 UTC

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

RemoteRef HEAD

RemoteSha 6159c832acc340157af08438521106186b4ab29d

Contents

calcAOV 2
calcBetaBin 4
calcHLGMRel 6
calcHLMRel 9
calcPerformance 10
calcReliability 11
calcResamplingIUR 12
calcSSR 14
colonoscopy 16
controlPerf 16
controlRel 17
example_df_1 18
example_df_2 18
misclassification_analysis 19
model_performance 20
plotCalibration 21
plotEstimates 22
plotN 23
plotPerformance 23
plotPredictedDistribution 24
plotReliability 25
plotSSR 26
psychreadmission 27
simulateData 27
tutorial_BB_agg_results 28
tutorial_BB_results 29
tutorial_profiling_results 29
tutorial_reliability_results_1 30
tutorial_reliability_results_2 30

Index **31**

`calcAOV` *Calculate reliability using one-way ANOVA method*

Description

This function estimates reliability using the one-way ANOVA method.

Usage

```
calcAOV(
  df,
  entity = "entity",
  y = "y",
  df.aggregate = FALSE,
```

```
  n = "n",
  mean = "mean",
  std.dev = "sd",
  ctrPerf = controlPerf()
)
```

Arguments

df	dataframe (assumed to be observation-level unless df.aggregate is changed below)
entity	data column containing the accountable entity identifier
y	data column containing the outcome variable
df.aggregate	set this to TRUE if the data have already been aggregated to include only summary data (sample size, means, and standard deviations) for each entity; default is FALSE.
n	if using aggregated data, data column containing the sample sizes for each entity; default is n.
mean	if using aggregated data, data column containing the sample means for each entity; default is mean.
std.dev	if using aggregated data, data column containing the sample standard deviations for each entity entity; default is sd.
ctrPerf	parameters to control performance measure calculation

Details

This function uses the aov() function from the stats package to calculate mean squares between and mean squares within entities.

Value

A list containing:

- MSB: mean squares between entities
- MSW: mean squares within entities
- F.stat: F-statistic from one-way ANOVA
- entity: list of entities
- n: sample sizes for each entity
- n0: aggregate sample size used in reliability calculation
- var.b.aov: between-entity variance
- var.w.aov: within-entity variance
- est.aov: entity-level reliability estimates

Author(s)

Kenneth Nieser (nieser@stanford.edu)

References

Nieser KJ, Harris AH. Comparing methods for assessing the reliability of health care quality measures. Statistics in Medicine. 2024 Oct 15;43(23):4575-94.

Examples

```
# Simulate data
df <- simulateData(n.entity = 50, n.obs = 100, mu = 25, r = .7, data.type = 'normal')

# Calculate reliability
out <- calcAOV(df = df, entity = 'entity', y = 'y')
summary(out$est.aov)

# Plot reliability by entity sample size
plot(out$n, out$est.aov)


## Reliability can also be calculated with data aggregated by entity
df.agg <- data.frame(
          entity = aggregate(y ~ entity, data = df, length)$entity,
          n = aggregate(y ~ entity, data = df, length)$y,
          mean = aggregate(y ~ entity, data = df, mean)$y,
          sd = aggregate(y ~ entity, data = df, sd)$y
          )

out2 <- calcAOV(df = df.agg, df.aggregate = TRUE, n = 'n', mean = 'mean', std.dev = 'sd')
summary(out2$est.aov)
```

---

calcBetaBin *Calculate reliability using a Beta-Binomial model*

Description

This function estimates reliability using a Beta-Binomial model. **NOTE:** currently, Beta-Binomial reliability estimates do not take risk-adjustment into account.

Usage

```
calcBetaBin(
  df = NULL,
  model = NULL,
  entity = "entity",
  y = "y",
  df.aggregate = FALSE,
  n = "n",
  x = "x",
  show.all = FALSE,
  ctrPerf = controlPerf()
)
```

Arguments

df	dataframe (assumed to be observation-level unless df.aggregate is changed below); if null, will use the dataframe from the model object
model	model; if null, will use an unadjusted model (NOTE: currently, Beta-Binomial reliability estimates do not take risk-adjustment into account.)
entity	data column containing the accountable entity identifier
y	data column containing the outcome variable
df.aggregate	set this to TRUE if the data have already been aggregated to include only summary data (sample sizes and numerators) for each entity; default is FALSE.
n	if using aggregated data, data column containing the sample size by entity
x	if using aggregated data, data column containing the number of observations that met measure criteria by entity
show.all	logical parameter indicating whether all variations of reliability estimates should be calculated; default is FALSE.
ctrPerf	parameters to control performance measure calculation

Details

To fit the Beta-Binomial model, the function first calculates method-of-moments estimates for the alpha and beta parameters which are used as starting values. Then, we use the optim() function to calculate maximum likelihood estimates with method = 'L-BFGS-B'. Reliability estimates are calculated used the maximum likelihood estimates of alpha and beta.

Value

A list containing:

- alpha: estimated alpha from Beta-Binomial model
- beta: estimated beta from Beta-Binomial model
- entity: list of entities
- n: sample sizes for each entity
- var.b.BB: between-entity variance
- var.w.BB: within-entity variance
- est.BB: entity-level reliability estimates

If show.all is set to TRUE, then the outputted list will also contain:

- var.w.FE: within-entity variance using fixed effect estimates of entity-specific outcome probabilities
- var.w.RE: within-entity variance using random effect estimates of entity-specific outcome probabilities
- var.w.J: within-entity variance using Bayesian estimates of entity-specific outcome probabilities, with Jeffrey's prior
- est.BB.FE: entity-level reliability estimates using fixed effect estimates of entity-specific outcome probabilities

- `est.BB.RE`: entity-level reliability estimates using random effect estimates of entity-specific outcome probabilities
- `est.BB.J`: entity-level reliability estimates using Bayesian estimates of entity-specific outcome probabilities, with Jeffrey's prior

Author(s)

Kenneth Nieser (nieser@stanford.edu)

References

Adams JL. The Reliability of Provider Profiling: A Tutorial. 2009.

Nieser KJ, Harris AH. Comparing methods for assessing the reliability of health care quality measures. Statistics in Medicine. 2024 Oct 15;43(23):4575-94.

Zhou G, Lin Z. Improved beta-binomial estimation for reliability of healthcare quality measures. medRxiv. 2023 Jan 9:2023-01.

Examples

```
# Simulate data
df <- simulateData(n.entity = 50, n.obs = 100, mu = .2, r = .7)

# Calculate reliability
out <- calcBetaBin(df = df, entity = 'entity', y = 'y')
summary(out$est.BB)

# Plot entity-level reliability by sample size
plot(out$n, out$est.BB)


## Reliability can also be calculated with data aggregated by entity
df.agg <- data.frame(
          entity = aggregate(y ~ entity, data = df, length)$entity,
          n = aggregate(y ~ entity, data = df, length)$y,
          x = aggregate(y ~ entity, data = df, sum)$y
          )

out2 <- calcBetaBin(df = df.agg, df.aggregate = TRUE, n = 'n', x = 'x')
summary(out2$est.BB)
```

`calcHLGMRel` *Calculate reliability using a hierarchical logistic regression model*

Description

This function estimates reliability using a hierarchical logistic regression model with random intercepts for each accountable entity.

Usage

```
calcHLGMRel(
  df = NULL,
  model = NULL,
  entity = "entity",
  y = "y",
  show.all = FALSE,
  ctrPerf = controlPerf(),
  ctrRel = controlRel()
)
```

Arguments

df	observation-level data; if null, will use the dataframe from the model object
model	model; if null, will use an unadjusted model
entity	data column containing the accountable entity identifier
y	data column containing the outcome variable
show.all	logical parameter indicating whether all variations of reliability estimates should be calculated; default is FALSE.
ctrPerf	parameters to control performance measure calculation
ctrRel	parameters to control reliability estimation

Details

Hierarchical logistic regression models are fit using `lme4::glmer()` with `control = lme4::glmerControl(optimizer = "bobyqa")` and `nAGQ = 0`.

Value

A list containing:

- `fit`: fitted model
- `marg.p`: marginal probability of the outcome
- `entity`: list of entities
- `n`: entity sample sizes
- `p`: entity-level sample proportions
- `p.re`: predicted entity-level outcome probabilities (i.e., shrunken estimates)
- `var.b`: between-entity variance on the outcome scale
- `var.w`: within-entity variance on the outcome scale
- `est.HLGM.delta`: reliability estimates on the outcome scale

If `show.all` is set to `TRUE`, then the outputted list will also contain:

- `var.b.HLGM.latent`: between-entity variance on the latent log-odds scale

- `var.b.HLGM.delta`: between-entity variance on the outcome scale using delta method approximation
- `var.b.MC`: between-entity variance on the outcome scale using Monte Carlo approximation
- `var.w.latent`: within-entity variance on the latent log-odds scale
- `var.w.delta`: within-entity variance on the outcome scale using delta method approximation
- `var.w.MC`: within-entity variance on the outcome scale using Monte Carlo approximation
- `var.w.FE`: within-entity variance calculated using fixed effect estimates of entity-level performance
- `var.w.RE`: within-entity variance calculated using random effect estimates of entity-level performance
- `est.HLGM.latent`: reliability estimates on latent log-odds scale
- `est.HLGM.delta`: reliability estimates on outcome scale using delta approximation
- `est.HLGM.MC`: reliability estimates on outcome scale using Monte Carlo approximation
- `est.HLGM.FE`: reliability estimates on outcome scale using fixed effect estimates
- `est.HLGM.RE`: reliability estimates on outcome scale using random effect estimates

Author(s)

Kenneth Nieser (nieser@stanford.edu)

References

Goldstein H, Browne W, Rasbash J. Partitioning variation in multilevel models. Understanding statistics: statistical issues in psychology, education, and the social sciences. 2002 Dec 2;1(4):223-31.

He K, Kalbfleisch JD, Yang Y, Fei Z, Kim S, Kang J, Li Y. Inter-unit reliability for quality measure testing. Journal of hospital administration. 2019 Jan 8;8(2):1.

Hwang J, Adams JL, Paddock SM. Defining and estimating the reliability of physician quality measures in hierarchical logistic regression models. Health Services and Outcomes Research Methodology. 2021 Mar;21(1):111-30.

Nieser KJ, Harris AH. Comparing methods for assessing the reliability of health care quality measures. Statistics in Medicine. 2024 Oct 15;43(23):4575-94.

Examples

```
# Simulate data
df <- simulateData(n.entity = 50, n.obs = 100, mu = .2, r = .7)

# Calculate reliability
out <- calcHLGMRel(df = df, entity = 'entity', y = 'y')
summary(out$est.HLGM.delta)

# Plot reliability
plot(out$n, out$est.HLGM.delta)

## Reliability estimates from additional methods can be obtained by toggling show.all parameter
```

```
out.all <- calcHLGMRel(df = df, entity = 'entity', y = 'y', show.all = TRUE)
summary(out.all$est.HLGM.latent)
summary(out.all$est.HLGM.delta)
summary(out.all$est.HLGM.MC)
summary(out.all$est.HLGM.FE)
summary(out.all$est.HLGM.RE)
```

`calcHLMRel` *Calculate reliability using a hierarchical linear regression model*

Description

This function estimates reliability using a hierarchical linear regression model with random intercepts for each accountable entity.

Usage

```
calcHLMRel(
  df = NULL,
  model = NULL,
  entity = "entity",
  y = "y",
  ctrPerf = controlPerf()
)
```

Arguments

`df`	observation-level data; if null, will use the dataframe from the model object
`model`	model; if null, will use an unadjusted model
`entity`	data column containing the accountable entity identifier
`y`	data column containing the outcome variable
`ctrPerf`	parameters to control performance measure calculation

Details

Hierarchical linear regression models are fit using `lme4::lmer()`.

Value

A list containing:

- `fit`: fitted model
- `entity`: list of entities
- `n`: entity sample sizes
- `var.b`: between-entity variance
- `var.w`: within-entity variance
- `est.HLM`: entity-level reliability

Author(s)

Kenneth Nieser (nieser@stanford.edu)

References

Nieser KJ, Harris AH. Comparing methods for assessing the reliability of health care quality measures. Statistics in Medicine. 2024 Oct 15;43(23):4575-94.

Examples

```
# Simulate data
df <- simulateData(n.entity = 50, n.obs = 100, mu = 12, r = .7, data.type = 'normal')

# Calculate reliability
out <- calcHLMRel(df = df, entity = 'entity', y = 'y')
summary(out$est.HLM)

# Plot reliability
plot(out$n, out$est.HLM)
```

---

calcPerformance *Calculate measure performance by accountable entity*

Description

This function calculates measure performance by accountable entity.

Usage

```
calcPerformance(
  df = NULL,
  model = NULL,
  entity = "entity",
  y = "y",
  data.type = "binary",
  ctrPerf = controlPerf()
)
```

Arguments

df	observation-level data; if null, will use the dataframe from the model object
model	model; if null, will use an unadjusted model
entity	data column containing the accountable entity identifier
y	data column containing the outcome variable
data.type	acceptable values are binary for 0/1 data (default: binary)
ctrPerf	parameters to control performance measure calculation

Value

A list including: *df: a cleaned dataframe used to calculate measure performance *model: the model used to calculate measure performance *fit: the fitted model results *marg.p: overall, unadjusted average performance across all entities *perf.results: performance results by entity

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
# simulate data
df <- simulateData(n.entity = 50, n.obs = 100, mu = .2, r = .7)

# calculate measure performance
out <- calcPerformance(df = df, entity = 'entity', y = 'y')

# plot performance
plotPerformance(out$perf.results)
```

---

calcReliability *Calculate reliability of quality measure performance*

Description

This function calculates several estimates of quality measure performance.

Usage

```
calcReliability(
  df = NULL,
  model = NULL,
  entity = "entity",
  y = "y",
  data.type = "binary",
  show.all = FALSE,
  ctrPerf = controlPerf(),
  ctrRel = controlRel()
)
```

Arguments

df	observation-level data; if null, will use the dataframe from the model object
model	model; if null, will use an unadjusted model
entity	data column containing the accountable entity identifier
y	data column containing the outcome variable

data.type	acceptable values are binary for 0/1 data and continuous for continuous data (default: binary)
show.all	logical indicator for whether full list of reliability method estimates should be calculated (default: FALSE)
ctrPerf	parameters to control performance measure calculation
ctrRel	parameters to control reliability estimation

Value

A list with reliability estimates. rel.results is a dataframe summarizing estimates from the various methods. Output from each method's respective function is also included. More details on output from each method can be found within the help documentation for the respective function for that method. For example, see calcSSR() for more detail on SSR.out.

Author(s)

Kenneth Nieser (nieser@stanford.edu)

References

Nieser KJ, Harris AH. Comparing methods for assessing the reliability of health care quality measures. Statistics in Medicine. 2024 Oct 15;43(23):4575-94.

See Also

calcAOV(), calcBetaBin(), calcHLGMRel(), calcHLMRel(), calcResamplingIUR(), calcSSR()

Examples

```
### Simulate data with binary outcome
df <- simulateData(n.entity = 50, n.obs = 100, mu = .2, r = .7)

# Calculate reliability
out <- calcReliability(df = df, entity = 'entity', y = 'y', ctrRel = controlRel(n.resamples = 10))

# Plot estimates
plotReliability(out)
```

---

calcResamplingIUR *Calculate reliability using resampling inter-unit reliability method*

Description

This function estimates reliability using the resampling inter-unit reliability method described in He et al. 2018.

Usage

```
calcResamplingIUR(
  df = NULL,
  model = NULL,
  entity = "entity",
  y = "y",
  ctrPerf = controlPerf(),
  ctrRel = controlRel()
)
```

Arguments

df	observation-level data; if null, will use the dataframe from the model object
model	model; if null, will use an unadjusted model
entity	data column containing the accountable entity identifier
y	data column containing the outcome variable
ctrPerf	parameters to control performance measure calculation
ctrRel	parameters to control reliability estimation

Details

In the current version, this function assumes that the measure is a simple mean of outcome values within each entity. However, this method is more flexible as described in He et al. 2018.

Value

A list containing:

- entity: list of entities
- n: entity sample sizes
- var.b: between-entity variance
- var.w: within-entity variance
- var.total: total variance
- IUR: entity-level reliability

Author(s)

Kenneth Nieser (nieser@stanford.edu)

References

He K, Kalbfleisch JD, Yang Y, Fei Z. Inter unit reliability for nonlinear models. Stat Med. 2019 Feb 28;38(5):844-854.

Examples

```
# Simulate data
df <- simulateData(n.entity = 50, n.obs = 100, mu = .2, r = .7)

# Calculate reliability
out <- calcResamplingIUR(df = df, entity = 'entity', y = 'y', ctrRel = controlRel(n.resamples = 10))
out$IUR
```

`calcSSR` *Calculate reliability using split-sample method*

Description

This function estimates reliability using the split-sample method.

Usage

```
calcSSR(
  df = NULL,
  model = NULL,
  entity = "entity",
  y = "y",
  data.type = "binary",
  ctrPerf = controlPerf(),
  ctrRel = controlRel()
)
```

Arguments

`df`	observation-level data; if null, will use the dataframe from the model object
`model`	model; if null, will use an unadjusted model
`entity`	data column containing the accountable entity identifier
`y`	data column containing the outcome variable
`data.type`	acceptable values are `binary` for 0/1 data and `continuous` for continuous data (default: `binary`)
`ctrPerf`	parameters to control performance measure calculation
`ctrRel`	parameters to control reliability estimation

Value

A list containing:

- `entity`: list of entities
- `n`: entity sample sizes

- icc: Spearman-Brown-adjusted intraclass correlation coefficients for each resample
- icc.lb: lower bound on confidence interval for Spearman-Brown-adjusted intraclass correlation coefficients for each resample
- icc.ub: upper bound on confidence interval for Spearman-Brown-adjusted intraclass correlation coefficients for each resample
- est.SSR: reliability estimate based on a single split
- est.PSSR: mean reliability estimate across resamples

If a risk-adjustment model is included then, the outputted list will contain:

- entity: list of entities
- n: entity sample sizes
- icc.oe: Spearman-Brown-adjusted intraclass correlation coefficients for OE ratios for each resample
- icc.oe.lb: lower bound on confidence interval for Spearman-Brown-adjusted intraclass correlation coefficients for OE ratios for each resample
- icc.oe.ub: upper bound on confidence interval for Spearman-Brown-adjusted intraclass correlation coefficients for OE ratios for each resample
- icc.pe: Spearman-Brown-adjusted intraclass correlation coefficients for PE ratios for each resample
- icc.pe.lb: lower bound on confidence interval for Spearman-Brown-adjusted intraclass correlation coefficients for PE ratios for each resample
- icc.pe.ub: upper bound on confidence interval for Spearman-Brown-adjusted intraclass correlation coefficients for PE ratios for each resample
- est.SSR.oe: reliability estimate for OE ratio based on a single split
- est.PSSR.oe: mean reliability estimate for OE ratio across resamples
- est.SSR.pe: reliability estimate for PE ratio based on a single split
- est.PSSR.pe: mean reliability estimate for PE ratio across resamples

Author(s)

Kenneth Nieser (nieser@stanford.edu)

References

Nieser KJ, Harris AH. Comparing methods for assessing the reliability of health care quality measures. Statistics in Medicine. 2024 Oct 15;43(23):4575-94.

Examples

```
# Simulate data
df <- simulateData(n.entity = 50, n.obs = 100, mu = .2, r = .7)

# Calculate reliability
out <- calcSSR(df = df, entity = 'entity', y = 'y', ctrRel = controlRel(n.resamples = 10))
out$est.PSSR
```

```
# Distribution of estimates obtained from the permutation sampling.
hist(out$icc)
summary(out$icc)
```

`colonoscopy` *Colonoscopy follow-up data*

Description

These data are from the Centers for Medicare and Medicaid Services (CMS). They contain the percentage of patients receiving appropriate recommendation for follow-up screening colonoscopy in calendar year 2023.

Usage

```
colonoscopy
```

Format

A data frame with five variables: `entity`, `p`, `n`, `x`.

Details

`entity` indicates the accountable health care entity; `p` indicates the follow-up rate; `n` indicates the denominator count for the measure; `x` indicates the numerator count for the measure.

`controlPerf` *Parameters for performance calculations*

Description

This function stores parameters for performance calculations

Usage

```
controlPerf(min.n = 2, alpha = 0.05, n.boots = 1000, n.cores = 2)
```

Arguments

`min.n`	minimum number of observations per entity
`alpha`	statistical significance level to use for confidence intervals
`n.boots`	number of bootstraps to use for confidence interval estimation for P/E ratios
`n.cores`	number of cores to use for parallel processing

Value

control parameters for performance calculations

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
str(controlPerf())
```

`controlRel` *Parameters for reliability calculations*

Description

This function stores parameters for reliability calculations.

Usage

```
controlRel(
  n.resamples = 100,
  n.cores = 2,
  SSRmethod = "permutation",
  fn = NA,
  MC.reps = 1000,
  d.steps = 10
)
```

Arguments

`n.resamples`	number of resamples for split-sample reliability method
`n.cores`	number of cores to use for parallel processing
`SSRmethod`	use either the `permutation` (default) or the `bootstrap` method for the split-sample reliability calculation
`fn`	aggregation function for observations within entities, default is `NA` and will produce entity-level means
`MC.reps`	number of Monte Carlo simulations to produce reliability estimates for data modeled with hierarchical logistic regression
`d.steps`	number of percentiles removed to check for misclassification probabilities

Value

control parameters for performance calculations

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
str(controlRel())
```

`example_df_1` *Example data set used in tutorial*

Description

This is an example simulated data frame generated by `simulateData()` with 100 accountable entities, an average of 50 observations per entity, a marginal outcome rate of 0.2, a median reliability of 0.7, and a regression coefficient of 0 for the covariate.

Usage

```
example_df_1
```

Format

A data frame with six variables: `entity`, `z`, `x1`, `lp`, `p`, `y`.

`example_df_2` *Example data set with a covariate used in tutorial*

Description

This is an example simulated data frame generated by `simulateData()` with 100 accountable entities, an average of 50 observations per entity, a marginal outcome rate of 0.2, and a median reliability of 0.7, and a regression coefficient of log(1.5) for the covariate.

Usage

```
example_df_2
```

Format

A data frame with six variables: `entity`, `z`, `x1`, `lp`, `p`, `y`.

`misclassification_analysis` *Calculate misclassification probabilities (in progress)*

Description

This function runs the misclassification functions

Usage

```
misclassification_analysis(
  df = NULL,
  model = NULL,
  entity = "entity",
  y = "y",
  ctrPerf = controlPerf(),
  ctrRel = controlRel()
)
```

Arguments

`df`	observation-level data; if null, will use the dataframe from the model object
`model`	model; if null, will use an unadjusted model
`entity`	data column containing the accountable entity identifier
`y`	data column containing the outcome variable
`ctrPerf`	parameters to control performance measure calculation
`ctrRel`	parameters to control reliability estimation

Value

Estimated misclassification probabilities

Author(s)

Kenneth Nieser (nieser@stanford.edu)

References

None

Examples

```
# TBD
```

`model_performance` *Calculate model performance*

Description

This function calculates risk model performance.

Usage

```
model_performance(
  df,
  model,
  entity = "entity",
  y = "y",
  data.type = "binary",
  predictor.clean = NULL,
  ctrPerf = controlPerf()
)
```

Arguments

`df`	observation-level data; if null, will use the dataframe from the model object
`model`	model; if null, will use an unadjusted model
`entity`	data column containing the accountable entity identifier
`y`	data column containing the outcome variable
`data.type`	acceptable values are `binary` for 0/1 data and `continuous` for continuous data (default: `binary`)
`predictor.clean`	optional list of formatted names of predictors in the model
`ctrPerf`	parameters to control performance measure calculation

Value

A list containing: *`data.type`: type of data: binary or continuous *`model`: risk-adjustment model *`fit`: fitted model results *`marg.p`: overall, unadjusted outcome rate (for binary outcome data only) *`c.statistic`: c-statistic, a measure of discrimination *`model.results`: a dataframe with one row for each predictor in the model

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
# Simulate data
df <- simulateData(n.entity = 100, n.obs = 80, mu = 0.2, r = 0.6, beta1 = log(1.6))

# Calculate risk-adjustment model performance
model.perf <- model_performance(df = df, model = 'y ~ x1 + (1|entity)')

# Plot estimated effects of predictors
plotEstimates(model.perf)
```

`plotCalibration` *Plot calibration curve for risk-adjustment model*

Description

This function creates a plot of the model calibration curve

Usage

```
plotCalibration(model.performance, quantiles = 10)
```

Arguments

`model.performance`	results from `model_performance()`
`quantiles`	number of quantiles to bin data; default is 10.

Details

This function only works for binary outcome data.

Value

A ggplot figure

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
# Simulate data
df <- simulateData(n.entity = 100, n.obs = 80, mu = 0.2, r = 0.6, beta1 = log(1.6))

# Calculate risk-adjustment model performance
model.perf <- model_performance(df = df, model = 'y ~ x1 + (1|entity)')

# Calibration plots
plotCalibration(model.perf)
plotCalibration(model.perf, quantiles = 5)
```

`plotEstimates` *Plot estimates from regression model used for risk-adjustment*

Description

This function creates a plot of model results

Usage

```
plotEstimates(model.performance)
```

Arguments

`model.performance`
: results from `model_performance()`

Value

A ggplot figure

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
# Simulate data
df <- simulateData(n.entity = 100, n.obs = 80, mu = 0.2, r = 0.6, beta1 = log(1.6))

# Calculate risk-adjustment model performance
model.perf <- model_performance(df = df, model = 'y ~ x1 + (1|entity)')

# Plot estimated effects of predictors
plotEstimates(model.perf)
```

plotN *Plot sample size distribution across accountable entities*

Description

This function creates a histogram of entity sample sizes.

Usage

```
plotN(n, bin.width = 10)
```

Arguments

n	vector of sample sizes
bin.width	width of bins; default is 10

Value

A ggplot figure

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
# plot sample sizes from colonoscopy dataset
plotN(colonoscopy$n)

# plot sample sizes from psychiatric readmissions dataset
plotN(psychreadmission$n, bin.width = 100)
```

plotPerformance *Plot measure performance across accountable entities*

Description

This function creates a plot of measure performance across accountable entities, using the perf.results dataframe from calcPerformance() output.

Usage

```
plotPerformance(df, plot.type = "p")
```

Arguments

df	perf.results dataframe from `calcPerformance()` output
plot.type	specifies which plot to return: • p: plots the entity-level unadjusted outcome rates • oe: plots the entity-level observed-to-expected risk-standardized rates • pe: plots the predicted-to-expected risk-standardized rates • OR: plots the odds ratios comparing each entity with the average entity; i.e., exponentiated random intercepts from the hierarchical logistic regression model. • correlation: plot of standardization ratios against the entity random intercepts • multiple: plot of measure performance across different risk-adjustment methods

Value

A ggplot figure

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
# simulate data
df <- simulateData(n.entity = 50, n.obs = 100, mu = .2, r = .7)

# calculate measure performance
out <- calcPerformance(df = df, entity = 'entity', y = 'y')

# plot performance
plotPerformance(out$perf.results, plot.type = 'p')
```

---

plotPredictedDistribution

Plot densities of predicted values by outcome group

Description

This function creates a plot of the distributions of predicted values by outcome group

Usage

```
plotPredictedDistribution(model.performance)
```

Arguments

model.performance
: results from model_performance()

Details

This function only works for binary outcome data.

Value

A ggplot figure

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
# Simulate data
df <- simulateData(n.entity = 100, n.obs = 80, mu = 0.2, r = 0.6, beta1 = log(1.6))

# Calculate risk-adjustment model performance
model.perf <- model_performance(df = df, model = 'y ~ x1 + (1|entity)')

# Plot predicted distributions
plotPredictedDistribution(model.perf)
```

---

plotReliability *Plot distributions of reliability estimates across entities*

Description

This function creates boxplots of reliability estimates across entities and different methods

Usage

```
plotReliability(rel.out)
```

Arguments

rel.out
: results from calcReliability()

Value

A ggplot figure

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
### Simulate data with binary outcome
df <- simulateData(n.entity = 50, n.obs = 100, mu = .2, r = .7)

# Calculate reliability
out <- calcReliability(df = df, entity = 'entity', y = 'y', ctrRel = controlRel(n.resamples = 10))

# Plot estimates
plotReliability(out)
```

---

plotSSR *Example plot of values used to calcualte split-sample reliability.*

Description

This function creates a plot of the measure performance for each entity split across two exclusive random samples of observations within each entity.

Usage

```
plotSSR(df, model = NULL, entity = "entity", y = "y", ctrPerf = controlPerf())
```

Arguments

df	observation-level data; if null, will use the dataframe from the model object
model	model; if null, will use an unadjusted model
entity	data column containing the accountable entity identifier
y	data column containing the outcome variable
ctrPerf	parameters to control performance measure calculation

Author(s)

Kenneth Nieser (nieser@stanford.edu)

References

None

Examples

```
# TBD
```

psychreadmission *Psychiatric 30-day, unplanned readmission data*

Description

These data are from the Centers for Medicare and Medicaid Services (CMS) and show the percentage of patients who return to a hospital for an unplanned inpatient stay after discharge. Data are from 07/01/2021 to 06/30/2023

Usage

```
psychreadmission
```

Format

A data frame with six variables: `entity`, `category`, `n`, `rate`, `rate.lwr`, `rate.upr`.

simulateData *Simulate data*

Description

This function simulates some data.

Usage

```
simulateData(
  n.entity,
  n.obs,
  mu,
  sd = 1,
  r,
  beta1 = 0,
  data.type = "binary",
  dist = "normal"
)
```

Arguments

`n.entity`	total number of entities to simulate
`n.obs`	average number of observations per entity; entity sample sizes are simulated from a Poisson distribution with mean given by n.obs OR a vector of length n.entity with entity sample sizes
`mu`	average probability of the outcome for binary data OR average outcome value for Normal data

sd	within-entity standard deviation for Normal data (default is 1).
r	median reliability
beta1	regression coefficient for covariate added to the linear predictor; default is 0. Note that for binary data, beta1 is on the log odds scale (e.g., beta1 = 0.4 corresponds to an odds ratio of about 1.5).
data.type	type of data to simulate. Valid options include: binary (default) and normal.
dist	specifies the distribution family to use to simulate provider performance. Valid options include: normal (default) and beta.

Value

A dataframe of simulated data.

Author(s)

Kenneth Nieser (nieser@stanford.edu)

Examples

```
# number of accountable entities
n.entity = 100

# average number of patients or cases per accountable entity
n.obs = 50

# marginal probability of the outcome
mu = 0.1

# approximate reliability for entity with a median number of patients
r = 0.6

# parameter for risk-adjustment model (i.e., coefficient for x1)
beta1 = log(1.5)

df <- simulateData(n.entity = n.entity, n.obs = n.obs, mu = mu, r = r, beta1 = beta1)
head(df)
```

---

tutorial_BB_agg_results

Beta-Binomial reliability results for example aggregated data set used in tutorial

---

Description

This is the pre-computed reliability results used in the tutorial for the Beta-Binomial method applied to an aggregated data set.

Usage

```
tutorial_BB_agg_results
```

Format

An object of class `list` of length 8.

---

tutorial_BB_results *Beta-Binomial reliability results for example data set used in tutorial*

Description

This is the pre-computed reliability results used in the tutorial for the Beta-Binomial method.

Usage

```
tutorial_BB_results
```

Format

An object of class `list` of length 8.

---

tutorial_profiling_results *Risk-adjusted profiling results for example data set used in tutorial*

Description

This is the pre-computed profiling results used in the tutorial for the risk-adjusted example.

Usage

```
tutorial_profiling_results
```

Format

An object of class `data.frame` with 100 rows and 35 columns.

tutorial_reliability_results_1

Reliability results for example data set used in tutorial

Description

This is the pre-computed reliability results used in the tutorial for example 1.

Usage

```
tutorial_reliability_results_1
```

Format

An object of class `list` of length 7.

tutorial_reliability_results_2

Reliability results for example data set with risk adjustment used in tutorial

Description

This is the pre-computed reliability results used in the tutorial for example 2.

Usage

```
tutorial_reliability_results_2
```

Format

An object of class `list` of length 7.

Index

∗ **datasets**
colonoscopy, 16
example_df_1, 18
example_df_2, 18
psychreadmission, 27
tutorial_BB_agg_results, 28
tutorial_BB_results, 29
tutorial_profiling_results, 29
tutorial_reliability_results_1, 30
tutorial_reliability_results_2, 30

calcAOV, 2
calcAOV(), *12*
calcBetaBin, 4
calcBetaBin(), *12*
calcHLGMRel, 6
calcHLGMRel(), *12*
calcHLMRel, 9
calcHLMRel(), *12*
calcPerformance, 10
calcReliability, 11
calcResamplingIUR, 12
calcResamplingIUR(), *12*
calcSSR, 14
calcSSR(), *12*
colonoscopy, 16
controlPerf, 16
controlRel, 17

example_df_1, 18
example_df_2, 18

misclassification_analysis, 19
model_performance, 20

plotCalibration, 21
plotEstimates, 22
plotN, 23
plotPerformance, 23
plotPredictedDistribution, 24
plotReliability, 25
plotSSR, 26
psychreadmission, 27

simulateData, 27

tutorial_BB_agg_results, 28
tutorial_BB_results, 29
tutorial_profiling_results, 29
tutorial_reliability_results_1, 30
tutorial_reliability_results_2, 30