Package: TwoCutoff (via r-universe)

October 26, 2026

Type Package

Title Deriving Clinically Interpretable Cutoffs for Disease Biomarkers

Version 0.1.0

Maintainer Bhrigu Kumar Rajbongshi <kumarbhrigu536@gmail.com>

Description Provides a reproducible pipeline for deriving two clinically meaningful cutoffs for disease biomarkers using a unified two-stage framework. The package integrates finite mixture modeling with risk prediction using biomarker plus clinical features, followed by decision curve analysis to evaluate clinical utility. Outputs include biomarker density plots, risk calibration curves, decision curves, and summary tables of diagnostic performance. Designed for researchers in bio-statistics, neurology, and data science, this package emphasizes reproducibility, transparency, and clear clinical relevance.

URL https://github.com/kumarbhrigu/TwoCutoff, https://kumarbhrigu.github.io/TwoCutoff/

BugReports https://github.com/kumarbhrigu/TwoCutoff/issues

Imports dplyr, mixtools, stats, pROC, xgboost, ggplot2, caret, patchwork, gridExtra

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 4.0.0)

RoxygenNote 7.3.2

NeedsCompilation no

Suggests rmarkdown, DiagrammeR, knitr, devtools

VignetteBuilder knitr

Author Bhrigu Kumar Rajbongshi [aut, cre, ctb], Seyed Ehsan Saffari [aut, ctb], Nastaran Marzban [aut, ctb]

Config/pak/sysreqs cmake make libicu-dev libuv1-dev libssl-dev

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

Date/Publication 2026-06-11 18:19:35 UTC

RemoteUrl https://github.com/kumarbhrigu/twocutoff

RemoteRef HEAD

RemoteSha d67c2f34e7e86580578adb36e2489b72644bf102

Contents

adjust_score 2
alzheimer_data 4
compare_performance 5
dca_analysis 6
derive_cutoffs_percentile 8
derive_cutoffs_sensspec 10
evaluate_performance 12
plot_two_cutoff 14
print.cutoff.percentile 16
print.cutoff.sensspec 16
print.twocutoff.adjscore 17

Index **18**

`adjust_score` *Adjust biomarker score with confounder-adjusted risk model*

Description

Function for biomarker score with confounder-adjusted risk model

Usage

```
adjust_score(
  data_set,
  biomarker,
  outcome,
  confounders = NULL,
  k = 2,
  maxit = 1000,
  epsilon = 1e-08,
  CompCase = FALSE,
  xgb_params = NULL,
  validate = FALSE,
  test_size = 0.2,
  seed = 123
)
```

Arguments

`data_set`	Data frame containing biomarker, outcome, and confounders.
`biomarker`	Character, name of biomarker column.
`outcome`	Character, name of binary outcome column (0/1).
`confounders`	Character vector of confounder column names (default = NULL).
`k`	Integer, number of mixture components (default = 2).
`maxit`	Integer, maximum EM iterations.
`epsilon`	Numeric, convergence tolerance for EM.
`CompCase`	Logical, whether to perform complete case analysis (default = FALSE).
`xgb_params`	Optional list of hyperparameters passed to `xgboost::xgb.train`. Can override default settings such as `max_depth` = `3`, `eta` = `0.05`, `subsample` = `0.8`, and `colsample_bytree` = `0.8`.
`validate`	Logical, whether to perform a train/test split. If TRUE, model performance is evaluated on a held-out test set. Validation is only performed when confounders are provided and an xgboost model is fitted.
`test_size`	Numeric, proportion of data used for the test set. Only used when `validate` = `TRUE`. Must be between 0 and 1.
`seed`	Integer, random seed for reproducibility of the train/test split.

Details

This function implements a two-stage modeling pipeline:

Stage 1: Finite Mixture Model (FMM)

- Fits a 2-component Gaussian mixture model to the biomarker distribution using the Expectation-Maximization (EM) algorithm (up to `maxit` iterations, tolerance `epsilon`).
- Estimates component means (μ_1, μ_2), standard deviations (σ_1, σ_2), and mixing proportions (π).
- Computes the posterior probability of disease for each subject:

$$\hat{p}(x) = \frac{\hat{\pi} \cdot N(x|\hat{\mu}_2, \hat{\sigma}_2)}{\hat{\pi} \cdot N(x|\hat{\mu}_2, \hat{\sigma}_2) + (1 - \hat{\pi}) \cdot N(x|\hat{\mu}_1, \hat{\sigma}_1)}$$

  where $N(x|\mu, \sigma)$ is the Gaussian density.
- The resulting `p_disease` represents the latent probability of disease given the biomarker value.

Stage 2: Confounder-Adjusted Risk Model (xgboost)

- Trains a binary logistic xgboost model using predictors: `p_disease` (from FMM) plus specified confounders.
- Model complexity (number of boosting iterations) is selected via cross-validation with early stopping.
- Produces an `adjusted_risk` score in [0,1] for each subject, representing the confounder-adjusted probability of disease.

The disease component is identified using `which.max(mu)`, assuming that higher biomarker values correspond to disease. If this assumption does not hold for your dataset, results may be misleading. A warning is issued if the assumption may not hold.

Value

A list containing:

- Mixture parameters (`mu`, `sigma`, `lambda`).
- Fitted xgboost model object.
- Training AUC (`train_auc`).
- Cross-validated AUC from xgboost CV (`cv_auc`).
- Test/validation AUC on held-out data (`test_auc`, only if `validate = TRUE`).
- Augmented dataset with columns: `p_disease` (posterior probability from FMM) and `adjusted_risk` (confounder-adjusted risk score).

Examples

```
n <- 500
set.seed(123)

# outcome
DISEASE <- rbinom(n, 1, 0.35)

# biomarker depends on disease
PTAU <- rnorm(n, mean = 25 + 10 * DISEASE, sd = 8)

# confounders (optional noise)
AGE <- round(rnorm(n, 73, 7))
SEX <- sample(c(0, 1), n, replace = TRUE)
BMI <- rnorm(n, 25, 4)
HTN <- rbinom(n, 1, 0.3)
DM <- rbinom(n, 1, 0.2)

df <- data.frame(PTAU, AGE, SEX, BMI, HTN, DM, DISEASE)

res <- adjust_score(
df,
biomarker = "PTAU", outcome = "DISEASE",
confounders = c("AGE","SEX","BMI","HTN","DM"), validate = TRUE)
head(res$data[,c("PTAU","p_disease","adjusted_risk")])
```

---

alzheimer_data *Alzheimer's Biomarker Dataset*

Description

This dataset contains simulated patient-level data for Alzheimer's disease biomarker analysis. It includes demographic variables, clinical covariates, biomarker values, and disease status. The dataset is suitable for testing finite mixture models, cutoff derivation, and performance evaluation.

Usage

```
data(alzheimer_data)
```

Format

A data frame with 500 rows and 7 variables:

PTAU `numeric`: Plasma p-tau biomarker concentration (pg/mL).

AGE `numeric`: Age in years (approx. 57-96).

SEX `integer`: Sex (0 = Female, 1 = Male).

BMI `numeric`: Body Mass Index (kg/m^2).

HTN `integer`: Hypertension status (0 = no, 1 = yes).

DM `integer`: Diabetes status (0 = no, 1 = yes).

DISEASE `integer`: Alzheimer's disease status (0 = no disease, 1 = disease).

Details

The dataset mimics typical biomarker distributions observed in Alzheimer's cohorts. Covariates (AGE, SEX, BMI, HTN, DM) act as confounders in biomarker-disease associations. Useful for demonstrating the two-cutoff approach in biomarker classification.

`compare_performance` *Compare cutoff methods*

Description

Compares rule-out and rule-in cutoffs, classification results, and performance metrics across multiple cutoff methods.

Usage

```
compare_performance(percentile_result, sensspec_result, outcome)
```

Arguments

`percentile_result`	Output list from `derive_cutoffs_percentile()`.
`sensspec_result`	Output list from `derive_cutoffs_sensspec()`.
`outcome`	Character, name of binary disease outcome column (0/1).

Value

Data frame summarizing cutoffs and performance metrics for each method.

Examples

```
n <- 500
set.seed(123)

# outcome
DISEASE <- rbinom(n, 1, 0.35)

# biomarker depends on disease
PTAU <- rnorm(n, mean = 25 + 10 * DISEASE, sd = 8)

# confounders (optional noise)
AGE <- round(rnorm(n, 73, 7))
SEX <- sample(c(0,1), n, replace = TRUE)
BMI <- rnorm(n, 25, 4)
HTN <- rbinom(n, 1, 0.3)
DM <- rbinom(n, 1, 0.2)

df <- data.frame(PTAU, AGE, SEX, BMI, HTN, DM, DISEASE)

res <- adjust_score(
 df,
 biomarker = "PTAU",
 outcome = "DISEASE",
 confounders = c("AGE","SEX","BMI","HTN","DM")
)

cutoffs_p <- derive_cutoffs_percentile(res, biomarker = "PTAU")
cutoffs_r <- derive_cutoffs_sensspec(res, biomarker = "PTAU", outcome = "DISEASE")
compare_performance(cutoffs_p, cutoffs_r, outcome = "DISEASE")
```

---

dca_analysis *Decision Curve Analysis (DCA)*

Description

This function evaluates the clinical utility of the confounder-adjusted risk model by quantifying net benefit across a range of decision thresholds and comparing against two reference strategies: treating all patients or treating none.

Usage

```
dca_analysis(
  adjust_result,
  outcome,
```

```
  thresholds = seq(0.01, 0.6, by = 0.01),
  risk_col = "adjusted_risk",
  ylim_lower = NULL
)
```

Arguments

adjust_result	Output list from adjust_score(), containing the augmented dataset with risk scores and outcome.
outcome	Character, name of binary disease outcome column (0/1).
thresholds	Numeric vector of risk thresholds (default = 0.01 to 0.60 by 0.01).
risk_col	Character, name of the risk score column to use (default = "adjusted_risk"). Allows generalisation beyond the default adjusted risk output.
ylim_lower	Numeric, lower bound for the y-axis in the decision curve plot. If NULL (default), the bound is computed dynamically as min(NB_model, -0.05, na.rm = TRUE) - 0.01.

Details

Concept

- DCA is a method to assess whether a predictive model improves clinical decision-making compared to default strategies.
- Net benefit (NB) balances true positives against false positives, weighted by the risk threshold:

$$NB(t) = \frac{TP(t)}{N} - \frac{FP(t)}{N} \cdot \frac{t}{1-t}$$

  where $TP(t)$ and $FP(t)$ are counts at threshold t, and N is the total sample size.
- Interpretation: The model is clinically useful at thresholds where

$$NB(model) > \max(NB(treat\ all), NB(treat\ none))$$

  .

Implementation

- Input: adjusted risk scores from adjust_score() and true disease labels.
- Thresholds: defaults to 0.01-0.60 in steps of 0.01 (1% to 60% risk).
- Strategies compared:
  - Model: net benefit using adjusted risk predictions.
  - Treat All: assume all subjects are positive.
  - Treat None: assume all subjects are negative.
- Output: a data frame of net benefit values across thresholds and a ggplot decision curve showing the three strategies.

Value

A list with:

- `data`: Data frame of net benefit values for each strategy across thresholds.
- `plot`: ggplot object showing the decision curve.

Examples

```
n <- 500
set.seed(123)

# outcome
DISEASE <- rbinom(n, 1, 0.35)

# biomarker depends on disease
PTAU <- rnorm(n, mean = 25 + 10 * DISEASE, sd = 8)

# confounders (optional noise)
AGE <- round(rnorm(n, 73, 7))
SEX <- sample(c(0, 1), n, replace = TRUE)
BMI <- rnorm(n, 25, 4)
HTN <- rbinom(n, 1, 0.3)
DM <- rbinom(n, 1, 0.2)

df <- data.frame(PTAU, AGE, SEX, BMI, HTN, DM, DISEASE)
res <- adjust_score(df,
biomarker = "PTAU",
outcome = "DISEASE",
confounders = c("AGE","SEX","BMI","HTN","DM"))

dca_out <- dca_analysis(res, outcome = "DISEASE")
head(dca_out$data)
print(dca_out$plot)
```

---

derive_cutoffs_percentile

Derive percentile-based biomarker cutoffs using adjusted risk

---

Description

Implements a two-cutoff approach to classify subjects into rule-out, rule-in, or indeterminate zones using confounder-adjusted risk scores.

Usage

```
derive_cutoffs_percentile(
  adjust_result,
  biomarker,
```

```
  low_thresh = 0.05,
  high_thresh = 0.7,
  low_pct = 0.8,
  high_pct = 0.2
)
```

Arguments

adjust_result	Output list from adjust_score(), containing the dataset with biomarker, adjusted_risk, and outcome.
biomarker	Character, name of biomarker column.
low_thresh	Numeric, threshold for defining the low-risk group (default = 0.05). Very small datasets may produce too few low-risk subjects at extreme thresholds, triggering fallback cutoffs.
high_thresh	Numeric, threshold for defining the high-risk group (default = 0.70). Very small datasets may produce too few high-risk subjects at extreme thresholds, triggering fallback cutoffs.
low_pct	Numeric, percentile used to derive the rule-out cutoff within the low-risk group (default = 0.80).
high_pct	Numeric, percentile used to derive the rule-in cutoff within the high-risk group (default = 0.20).

Details

Method

- The function uses adjusted_risk (from adjust_score()) to define clinically meaningful risk groups.
- Low-risk group: subjects with adjusted risk < low_thresh (default 0.05), representing individuals unlikely to have disease. The rule-out cutoff (c_L) is defined as the 80th percentile of biomarker values within this group.
- High-risk group: subjects with adjusted risk greater than high_thresh (default 0.70), representing individuals likely to have disease. The rule-in cutoff (c_H) is defined as the 20th percentile of biomarker values within this group.
- If either group is too small, fallback cutoffs are used based on the global biomarker distribution (5th and 95th percentiles).
- Final classification: biomarker < c_L "Negative"; biomarker > c_H "Positive"; otherwise "Indeterminate".

Value

List with rule-out cutoff (c_L), rule-in cutoff (c_H), classification vector, and augmented dataset.

Author(s)

Bhrigu Kumar Rajbongshi, Seyed Ehsan Saffari and Nastaran Marzban

References

Giacomucci, G., Crucitti, C., Ingannato, A., et al. (2025). The two cut-offs approach for plasma p-tau217 in detecting Alzheimer's disease. Alzheimer's & Dementia: Diagnosis, Assessment & Disease Monitoring, 17:e70116.

Hazan, J., Liu, K.Y., Isaacs, J.D., Howard, R. (2025). Cut-points and gray zones: The challenges of integrating Alzheimer's disease plasma biomarkers into clinical practice. Alzheimer's & Dementia, 21:e70113.

Examples

```
n <- 500
set.seed(123)

# outcome
DISEASE <- rbinom(n, 1, 0.35)

# biomarker depends on disease
PTAU <- rnorm(n, mean = 25 + 10 * DISEASE, sd = 8)

# confounders (optional noise)
AGE <- round(rnorm(n, 73, 7))
SEX <- sample(c(0, 1), n, replace = TRUE)
BMI <- rnorm(n, 25, 4)
HTN <- rbinom(n, 1, 0.3)
DM <- rbinom(n, 1, 0.2)

df <- data.frame(PTAU, AGE, SEX, BMI, HTN, DM, DISEASE)

res <- adjust_score(
df,
biomarker = "PTAU",
outcome = "DISEASE",
confounders = c("AGE","SEX","BMI","HTN","DM")
)

cutoffs <- derive_cutoffs_percentile(res, biomarker = "PTAU")
cutoffs$c_L; cutoffs$c_H
table(cutoffs$class)
```

`derive_cutoffs_sensspec` *Derive ROC-based biomarker cutoffs using adjusted risk*

Description

Implements a ROC-guided two-cutoff approach to classify subjects into rule-out, rule-in, or indeterminate zones using confounder-adjusted risk scores.

Usage

```
derive_cutoffs_sensspec(
  adjust_result,
  biomarker,
  outcome,
  sens_target = 0.8,
  spec_target = 0.8,
  loess_span = 0.35
)
```

Arguments

adjust_result	Output list from adjust_score(), containing the augmented dataset with biomarker, adjusted_risk, and outcome.
biomarker	Character, name of biomarker column.
outcome	Character, name of binary disease outcome column (0/1). If missing, the outcome stored in adjust_result will be used.
sens_target	Numeric, target sensitivity for rule-out threshold (default = 0.80).
spec_target	Numeric, target specificity for rule-in threshold (default = 0.80).
loess_span	Numeric, smoothing parameter for LOESS regression (default = 0.35). Smaller values give more flexibility, larger values give smoother fits.

Details

Method

- Compute ROC curve with predictor = adjusted_risk and outcome = true disease status.
- Identify thresholds:
  - Rule-out threshold (T_L): lowest risk threshold with sensitivity$\geq$ sens_target (default 0.80).
  - Rule-in threshold (T_H): highest risk threshold with specificity$\geq$ spec_target (default 0.80).
- Map thresholds back to biomarker scale by fitting a LOESS regression (adjusted_risk ~ biomarker, default span = 0.35 (user-adjustable via loess_span)) and interpolating biomarker values at T_L and T_H.
- Classification:
  - biomarker < c_L= "Negative" (rule-out).
  - biomarker > c_H= "Positive" (rule-in).
  - otherwise = "Indeterminate".

Value

List with rule-out cutoff (c_L), rule-in cutoff (c_H), classification vector, and augmented dataset.

Author(s)

Bhrigu Kumar Rajbongshi, Seyed Ehsan Saffari and Nastaran Marzban

References

Giacomucci, G., Crucitti, C., Ingannato, A., et al. (2025). The two cut-offs approach for plasma p-tau217 in detecting Alzheimer's disease. Alzheimer's & Dementia: Diagnosis, Assessment & Disease Monitoring, 17:e70116.

Hazan, J., Liu, K.Y., Isaacs, J.D., Howard, R. (2025). Cut-points and gray zones: The challenges of integrating Alzheimer's disease plasma biomarkers into clinical practice. Alzheimer's & Dementia, 21:e70113.

Examples

```
n <- 500
set.seed(123)

# outcome
DISEASE <- rbinom(n, 1, 0.35)

# biomarker depends on disease
PTAU <- rnorm(n, mean = 25 + 10 * DISEASE, sd = 8)

# confounders (optional noise)
AGE <- round(rnorm(n, 73, 7))
SEX <- sample(c(0,1), n, replace = TRUE)
BMI <- rnorm(n, 25, 4)
HTN <- rbinom(n, 1, 0.3)
DM <- rbinom(n, 1, 0.2)

df <- data.frame(PTAU, AGE, SEX, BMI, HTN, DM, DISEASE)

res <- adjust_score(
 df,
 biomarker = "PTAU",
 outcome = "DISEASE",
 confounders = c("AGE","SEX","BMI","HTN","DM")
)

cutoffs <- derive_cutoffs_sensspec(res, biomarker = "PTAU", outcome = "DISEASE")
cutoffs$c_L; cutoffs$c_H
table(cutoffs$class)
```

---

evaluate_performance *Evaluate classification performance*

Description

Computes diagnostic performance metrics based on classification results from cutoff functions (percentile-based or ROC-based).

Usage

```
evaluate_performance(cutoff_result, outcome)
```

Arguments

cutoff_result	Output list from derive_cutoffs_percentile() or derive_cutoffs_sensspec(), containing classification vector.
outcome	Character, name of binary disease outcome column (0/1).

Details

The function uses the classification vector returned by derive_cutoffs_percentile() or derive_cutoffs_sensspec() and compares it against the true disease labels to compute:

- **NPV (Negative Predictive Value)**: TN / (TN + FN), proportion of true negatives among predicted negatives.
- **PPV (Positive Predictive Value)**: TP / (TP + FP), proportion of true positives among predicted positives.
- **Overall Sensitivity**: TP / (TP + FN) across the full cohort.
- **Overall Specificity**: TN / (TN + FP) across the full cohort.
- **% in Middle Zone**: Proportion of subjects classified as indeterminate (those with biomarker values between c_L and c_H).

Indeterminate subjects (those with biomarker values between the rule-out cutoff c_L and the rule-in cutoff c_H) are excluded from the sensitivity and specificity calculations. They do not contribute to the counts of true positives, true negatives, false positives, or false negatives. This design is intentional to reflect clinical practice, where patients in the gray zone are not classified as definitively diseased or non-diseased. As a result, sensitivity and specificity are computed only on subjects outside the indeterminate zone, while the proportion of indeterminate cases is reported separately.

Value

Data frame with performance metrics.

Examples

```
n <- 500
set.seed(123)

# outcome
DISEASE <- rbinom(n, 1, 0.35)

# biomarker depends on disease
PTAU <- rnorm(n, mean = 25 + 10 * DISEASE, sd = 8)

# confounders (optional noise)
AGE <- round(rnorm(n, 73, 7))
SEX <- sample(c(0, 1), n, replace = TRUE)
BMI <- rnorm(n, 25, 4)
```

```
HTN <- rbinom(n, 1, 0.3)
DM <- rbinom(n, 1, 0.2)

df <- data.frame(PTAU, AGE, SEX, BMI, HTN, DM, DISEASE)

res <- adjust_score(
df,
biomarker = "PTAU",
outcome = "DISEASE",
confounders = c("AGE","SEX","BMI","HTN","DM"))

cutoffs <- derive_cutoffs_percentile(res, biomarker = "PTAU")
# cutoffs <- derive_cutoffs_sensspec(res, biomarker = "PTAU")

perf <- evaluate_performance(cutoffs, outcome = "DISEASE")
```

`plot_two_cutoff` *Plot two-cutoff classification*

Description

Visualizes biomarker values with rule-out and rule-in cutoffs, using results from percentile-based or ROC-based cutoff functions.

Usage

```
plot_two_cutoff(
  cutoff_result,
  biomarker,
  outcome,
  panel_title = NULL,
  negative_label = "Control",
  positive_label = "Disease",
  ylab = NULL,
  add_table = NULL,
  point_alpha = 0.55
)
```

Arguments

`cutoff_result`	Output list from `derive_cutoffs_percentile()` or `derive_cutoffs_sensspec()`, containing: • `c_L`: lower cutoff (rule-out threshold) • `c_H`: upper cutoff (rule-in threshold) • `data`: dataset including biomarker and outcome
`biomarker`	Character, name of biomarker column.

outcome	Character, name of binary disease outcome column (0/1).
panel_title	Optional character vector of panel titles.
negative_label	Label for non-diseased group (default = "Control").
positive_label	Label for diseased group (default = "Disease").
ylab	Y-axis label (default = "Plasma biomarker").
add_table	Logical, whether to add confusion table below plot (default = NULL).
point_alpha	Numeric, transparency for points (default = 0.55).

Details

Subjects are classified as:

- Negative (rule-out): biomarker < c_L
- Positive (rule-in): biomarker > c_H
- Indeterminate: between c_L and c_H

The plot shows jittered biomarker values by true group, with horizontal dashed lines indicating the rule-out (c_L) and rule-in (c_H) thresholds. Points are colored by predicted classification.

Value

Confusion tables for each panel (invisible).

Author(s)

Bhrigu Kumar Rajbongshi, Seyed Ehsan Saffari and Nastaran Marzban

Examples

```
n <- 500
set.seed(123)

# outcome
DISEASE <- rbinom(n, 1, 0.35)

# biomarker depends on disease
PTAU <- rnorm(n, mean = 25 + 10 * DISEASE, sd = 8)

# confounders (optional noise)
AGE <- round(rnorm(n, 73, 7))
SEX <- sample(c(0, 1), n, replace = TRUE)
BMI <- rnorm(n, 25, 4)
HTN <- rbinom(n, 1, 0.3)
DM <- rbinom(n, 1, 0.2)

df <- data.frame(PTAU, AGE, SEX, BMI, HTN, DM, DISEASE)

res <- adjust_score(
df,
```

```
biomarker = "PTAU",
outcome = "DISEASE",
confounders = c("AGE","SEX","BMI","HTN","DM"))

cutoffs <- derive_cutoffs_percentile(res, biomarker = "PTAU")
plot_two_cutoff(cutoffs, biomarker = "PTAU", outcome = "DISEASE")
```

---

print.cutoff.percentile *Print method for derive_cutoffs_percentile objects*

Description

Provides a concise summary of percentile-based cutoff results.

Usage

```
## S3 method for class 'cutoff.percentile'
print(x, ...)
```

Arguments

x	An object returned by derive_cutoffs_percentile().
...	Additional arguments (ignored).

---

print.cutoff.sensspec *Print method for derive_cutoffs_sensspec objects*

Description

Provides a concise summary of ROC-based cutoff results.

Usage

```
## S3 method for class 'cutoff.sensspec'
print(x, ...)
```

Arguments

x	An object returned by derive_cutoffs_sensspec().
...	Additional arguments (ignored).

print.twocutoff.adjscore

Print Method for twocutoff.adjscore Objects

Description

The `print.twocutoff.adjscore` function provides a concise summary of the output from the `adjust_score` function. It displays mixture parameters, cutoff values, and basic classification counts.

Usage

```
## S3 method for class 'twocutoff.adjscore'
print(x, ...)
```

Arguments

`x`	An object of class `twocutoff.adjscore`, returned by the `adjust_score` function.
`...`	Additional arguments (not used).

Value

The object is returned invisibly after printing a summary.

Index

∗ **datasets**
alzheimer_data, 4

adjust_score, 2
alzheimer_data, 4

compare_performance, 5

dca_analysis, 6
derive_cutoffs_percentile, 8
derive_cutoffs_sensspec, 10

evaluate_performance, 12

plot_two_cutoff, 14
print.cutoff.percentile, 16
print.cutoff.sensspec, 16
print.twocutoff.adjscore, 17