Package 'CRTConjoint'

July 21, 2025

Title Conditional Randomization Testing (CRT) Approach for Conjoint Analysis

Version 0.1.0

Maintainer Dae Woong Ham <daewoongham@g.harvard.edu>

Description Computes p-value according to the CRT using the HierNet test statistic. For more details, see Ham, Imai, Janson (2022) ``Using Machine Learning to Test Causal Hypotheses in Conjoint Analysis" <doi:10.48550/arXiv.2201.08343>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.0

Imports utils, methods, doSNOW, foreach, Rcpp, snow

Depends R (>= 2.10)

LazyData true

LinkingTo Rcpp

Suggests knitr, rmarkdown

VignetteBuilder knitr

URL https://github.com/daewoongham97/CRTConjoint

BugReports https://github.com/daewoongham97/CRTConjoint/issues

NeedsCompilation yes

Author Dae Woong Ham [aut, cre],
Kosuke Imai [aut],
Lucas Janson [aut],
Jacob Bien [ctb, cph]

Repository CRAN

Date/Publication 2022-06-09 08:00:05 UTC

Contents

CRT_carryovereffect 2
CRT_fatigueeffect 6
CRT_profileordereffect 8
CRT_pval 11
immigrationdata 15

Index **17**

`CRT_carryovereffect` *Testing carryover effect in Conjoint Experiments*

Description

This function takes a conjoint dataset and returns the p-value when using the CRT to test if the carryover effect holds using HierNet test statistic. The function requires user to specify the outcome, all factors used in the conjoint experiment, and the evaluation task number. By default, this function assumes a uniform randomization of factor levels. The function assumes the forced choice conjoint experiment and consequently assumes the data to contain the left and right profile factors in separate column in the dataset supplied.

Usage

```
CRT_carryovereffect(
  formula,
  data,
  left,
  right,
  task,
  design = "Uniform",
  supplyown_resamples = NULL,
  profileorder_constraint = TRUE,
  non_factor = NULL,
  B = 200,
  parallel = TRUE,
  num_cores = 2,
  nfolds = 3,
  lambda = c(20, 30, 40),
  tol = 0.001,
  seed = sample(c(1:1000), size = 1),
  verbose = TRUE
)
```

Arguments

`formula`	A formula object specifying the outcome variable on the left-hand side and factors of (X,Z) and respondent characteristics (V) in the right hand side. RHS

variables should be separated by + signs and should only contain either left or right for each (X,Z). For example Y ~ Country_left + Education_left is sufficient as opposed to Y ~ Country_left + Country_right + Education_left + Education_right

`data` A dataframe containing outcome variable and all factors (X,Z,V) (including both left and right profile factors). All (X,Z,V) listed in the formula above are expected to be of class factor unless explicitly stated in non_factor input.

`left` Vector of column names of data that corresponds to the left profile factors

`right` Vector of column names of data that corresponds to the right profile factors. NOTE: left and right are assumed to be the same length and the order should correspond to the same variables. For example left = c("Country_left", "Education_left") and right = c("Country_right", "Education_right")

`task` A character string indicating column of data that contains the task evaluation. IMPORTANT: The task variable is assumed to have no missing tasks, i.e., each respondent should have 1:J tasks. Please drop respondents with missing tasks.

`design` A character string of one of the following options: "Uniform" or "Manual". "Uniform" refers to a completely uniform design where all (X,Z) are sampled uniformly. "Manual" refers to more complex conjoint designs, where the user will supply their own resamples in supplyown_resamples input.

`supplyown_resamples`
List of length B that contains own resamples of X when design="Manual". Each element of list should contain a dataframe with the same number of rows of data and two columns for the left and right profile values of X.

`profileorder_constraint`
Boolean indicating whether to enforce profile order constraint (default = TRUE)

`non_factor` A vector of strings indicating columns of data that are not factors. This should only be used for respondent characteristics (V) that are not factors. For example non_factor = "Respondent_Age".

`B` Numeric integer value indicating the number of resamples for the CRT procedure. Default value is B=200.

`parallel` Boolean indicating whether parallel computing should be used. Default value is TRUE.

`num_cores` Numeric integer indicating number of cores to use when parallel=TRUE. num_cores should not exceed the number of cores the user's machine can handle. Default is 2.

`nfolds` Numeric integer indicating number of cross-validation folds. Default is 3.

`lambda` Numeric vector indicating lambda used for cross-validation for HierNet fit. Default lambda=c(20,30,40).

`tol` Numeric value indicating acceptable tolerance for terminating optimization fit for HierNet. Default is tol=1e-3. WARNING: Do not increase as it greatly increases computation time.

`seed` Seed used for CRT procedure

`verbose` Boolean indicating verbose output. Default verbose=TRUE

Value

A list containing:

p_val	A numeric value for the p-value testing carryover effect.
obs_test_stat	A numeric value for the observed test statistic.
resampled_test_stat	Matrix containing all the B resampled test statistics
tol	Tolerance used for HierNet
lam	Best cross-validated lambda
hiernet_fit	An object of class hiernet that contains the hiernet fit for the observed test statistic
seed	Seed used
elapsed_time	Elapsed time

References

Ham, D., Janson, L., and Imai, K. (2022) Using Machine Learning to Test Causal Hypotheses in Conjoint Analysis

Examples

```
# Subset of Immigration Choice Conjoint Experiment Data from Hainmueller et. al. (2014).
data("immigrationdata")
form = formula("Y ~ FeatEd + FeatGender + FeatCountry + FeatReason + FeatJob +
FeatExp + FeatPlans + FeatTrips + FeatLang + ppage + ppeducat + ppethm + ppgender")
left = colnames(immigrationdata)[1:9]
right = colnames(immigrationdata)[10:18]
# Each respondent evaluated 5 tasks
J = 5
carryover_df = immigrationdata
carryover_df$task = rep(1:J, nrow(carryover_df)/J)
# Since immigration conjoint experiment had dependent randomization for several factors
# we supply our own resamples
resample_func_immigration = function(x, seed = sample(c(0, 1000), size = 1), left_idx, right_idx) {
 set.seed(seed)
 df = x[, c(left_idx, right_idx)]
 variable = colnames(x)[c(left_idx, right_idx)]
 len = length(variable)
 resampled = list()
 n = nrow(df)
 for (i in 1:len) {
   var = df[, variable[i]]
   lev = levels(var)
   resampled[[i]] = factor(sample(lev, size = n, replace = TRUE))
 }

 resampled_df = data.frame(resampled[[1]])
 for (i in 2:len) {
   resampled_df = cbind(resampled_df, resampled[[i]])
```

```
}
colnames(resampled_df) = colnames(df)

#escape persecution was dependently randomized
country_1 = resampled_df[, "FeatCountry"]
country_2 = resampled_df[, "FeatCountry_2"]
i_1 = which((country_1 == "Iraq" | country_1 == "Sudan" | country_1 == "Somalia"))
i_2 = which((country_2 == "Iraq" | country_2 == "Sudan" | country_2 == "Somalia"))

reason_1 = resampled_df[, "FeatReason"]
reason_2 = resampled_df[, "FeatReason_2"]
levs = levels(reason_1)
r_levs = levs[c(2,3)]

reason_1 = sample(r_levs, size = n, replace = TRUE)

reason_1[i_1] = sample(levs, size = length(i_1), replace = TRUE)

reason_2 = sample(r_levs, size = n, replace = TRUE)

reason_2[i_2] = sample(levs, size = length(i_2), replace = TRUE)

resampled_df[, "FeatReason"] = reason_1
resampled_df[, "FeatReason_2"] = reason_2

#profession high skill fix
educ_1 = resampled_df[, "FeatEd"]
educ_2 = resampled_df[, "FeatEd_2"]
i_1 = which((educ_1 == "Equivalent to completing two years of college in the US" |
 educ_1 == "Equivalent to completing a college degree in the US" |
 educ_1 == "Equivalent to completing a graduate degree in the US"))
i_2 = which((educ_2 == "Equivalent to completing two years of college in the US" |
educ_2 == "Equivalent to completing a college degree in the US" |
educ_2 == "Equivalent to completing a graduate degree in the US"))


job_1 = resampled_df[, "FeatJob"]
job_2 = resampled_df[, "FeatJob_2"]
levs = levels(job_1)
# take out computer programmer, doctor, financial analyst, and research scientist
r_levs = levs[-c(2,4,5, 9)]

job_1 = sample(r_levs, size = n, replace = TRUE)

job_1[i_1] = sample(levs, size = length(i_1), replace = TRUE)

job_2 = sample(r_levs, size = n, replace = TRUE)

job_2[i_2] = sample(levs, size = length(i_2), replace = TRUE)

resampled_df[, "FeatJob"] = job_1
resampled_df[, "FeatJob_2"] = job_2
```

```
 resampled_df[colnames(resampled_df)] = lapply(resampled_df[colnames(resampled_df)], factor )

 return(resampled_df)
}


own_resamples = list()
B = 50
for (i in 1:B) {
 newdf = resample_func_immigration(carryover_df, left_idx = 1:9, right_idx = 10:18, seed = i)
 own_resamples[[i]] = newdf
}
carryover_test = CRT_carryovereffect(formula = form, data = carryover_df, left = left,
right = right, task = "task", supplyown_resamples = own_resamples, B = B)
carryover_test$p_val
```

CRT_fatigueeffect *Testing fatigue effect in Conjoint Experiments*

Description

This function takes a conjoint dataset and returns the p-value when using the CRT to test if the fatigue effect holds using HierNet test statistic. The function requires user to specify the outcome, all factors used in the conjoint experiment, and both the evaluation task number and respondent index. The function assumes the forced choice conjoint experiment and consequently assumes the data to contain the left and right profile factors in separate column in the dataset supplied.

Usage

```
CRT_fatigueeffect(
  formula,
  data,
  left,
  right,
  task,
  respondent,
  profileorder_constraint = TRUE,
  non_factor = NULL,
  B = 200,
  parallel = TRUE,
  num_cores = 2,
  nfolds = 3,
  lambda = c(20, 30, 40),
  tol = 0.001,
  speedup = TRUE,
  seed = sample(c(1:1000), size = 1),
  verbose = TRUE
)
```

Arguments

formula	A formula object specifying the outcome variable on the left-hand side and factors of (X,Z) and respondent characteristics (V) in the right hand side. RHS variables should be separated by + signs and should only contain either left or right for each (X,Z). For example Y ~ Country_left + Education_left is sufficient as opposed to Y ~ Country_left + Country_right + Education_left + Education_right
data	A dataframe containing outcome variable and all factors (X,Z,V) (including both left and right profile factors). All (X,Z,V) listed in the formula above are expected to be of class factor unless explicitly stated in non_factor input.
left	Vector of column names of data that corresponds to the left profile factors
right	Vector of column names of data that corresponds to the right profile factors. NOTE: left and right are assumed to be the same length and the order should correspond to the same variables. For example left = c("Country_left", "Education_left") and right = c("Country_right", "Education_right")
task	A character string indicating column of data that contains the task evaluation. IMPORTANT: The task variable is assumed to have no missing tasks, i.e., each respondent should have 1:J tasks. Please drop respondents with missing tasks.
respondent	A character string indicating column of data that contains the respondent index. The column should contain integers from 1:N indicating respondent index.
profileorder_constraint	Boolean indicating whether to enforce profile order constraint (default = TRUE)
non_factor	A vector of strings indicating columns of data that are not factors. This should only be used for respondent characteristics (V) that are not factors. For example non_factor = "Respondent_Age".
B	Numeric integer value indicating the number of resamples for the CRT procedure. Default value is B=200.
parallel	Boolean indicating whether parallel computing should be used. Default value is TRUE.
num_cores	Numeric integer indicating number of cores to use when parallel=TRUE. num_cores should not exceed the number of cores the user's machine can handle. Default is 2.
nfolds	Numeric integer indicating number of cross-validation folds. Default is 3.
lambda	Numeric vector indicating lambda used for cross-validation for HierNet fit. Default lambda=c(20,30,40).
tol	Numeric value indicating acceptable tolerance for terminating optimization fit for HierNet. Default is tol=1e-3. WARNING: Do not increase as it greatly increases computation time.
speedup	Boolean indicating whether to employ computational tricks to make function run faster. It is always recommended to use default speedup=TRUE.
seed	Seed used for CRT procedure
verbose	Boolean indicating verbose output. Default verbose=TRUE

Value

A list containing:

p_val	A numeric value for the p-value testing fatigue effect.
obs_test_stat	A numeric value for the observed test statistic.
resampled_test_stat	Matrix containing all the B resampled test statistics
tol	Tolerance used for HierNet
lam	Best cross-validated lambda
hiernet_fit	An object of class hiernet that contains the hiernet fit for the observed test statistic
seed	Seed used
elapsed_time	Elapsed time

References

Ham, D., Janson, L., and Imai, K. (2022) Using Machine Learning to Test Causal Hypotheses in Conjoint Analysis

Examples

```
# Subset of Immigration Choice Conjoint Experiment Data from Hainmueller et. al. (2014).
data("immigrationdata")
form = formula("Y ~ FeatEd + FeatGender + FeatCountry + FeatReason + FeatJob +
FeatExp + FeatPlans + FeatTrips + FeatLang + ppage + ppeducat + ppethm + ppgender")
left = colnames(immigrationdata)[1:9]
right = colnames(immigrationdata)[10:18]
# Each respondent evaluated 5 tasks
J = 5
fatigue_df = immigrationdata
fatigue_df$task = rep(1:J, nrow(fatigue_df)/J)
fatigue_df$respondent = rep(1:(nrow(fatigue_df)/J), each = J)

fatigue_test = CRT_fatigueeffect(formula = form, data = fatigue_df, left = left,
right = right, task = "task", respondent = "respondent", B = 50)
fatigue_test$p_val
```

CRT_profileordereffect

Testing profile order effect in Conjoint Experiments

Description

This function takes a conjoint dataset and returns the p-value when using the CRT to test if the profile order effect holds using HierNet test statistic. The function requires user to specify the outcome, all factors used in the conjoint experiment, and any additional respondent characteristics. The function assumes the forced choice conjoint experiment and consequently assumes the data to contain the left and right profile factors in separate column in the dataset supplied.

Usage

```
CRT_profileordereffect(
  formula,
  data,
  left,
  right,
  non_factor = NULL,
  B = 200,
  parallel = TRUE,
  num_cores = 2,
  nfolds = 3,
  lambda = c(20, 30, 40),
  tol = 0.001,
  speedup = TRUE,
  seed = sample(c(1:1000), size = 1),
  verbose = TRUE
)
```

Arguments

formula	A formula object specifying the outcome variable on the left-hand side and factors of (X,Z) and respondent characteristics (V) in the right hand side. RHS variables should be separated by + signs and should only contain either left or right for each (X,Z). For example Y ~ Country_left + Education_left is sufficient as opposed to Y ~ Country_left + Country_right + Education_left + Education_right
data	A dataframe containing outcome variable and all factors (X,Z,V) (including both left and right profile factors). All (X,Z,V) listed in the formula above are expected to be of class factor unless explicitly stated in non_factor input.
left	Vector of column names of data that corresponds to the left profile factors
right	Vector of column names of data that corresponds to the right profile factors. NOTE: left and right are assumed to be the same length and the order should correspond to the same variables. For example left = c("Country_left", "Education_left") and right = c("Country_right", "Education_right")
non_factor	A vector of strings indicating columns of data that are not factors. This should only be used for respondent characteristics (V) that are not factors. For example non_factor = "Respondent_Age".
B	Numeric integer value indicating the number of resamples for the CRT procedure. Default value is B=200.
parallel	Boolean indicating whether parallel computing should be used. Default value is TRUE.
num_cores	Numeric integer indicating number of cores to use when parallel=TRUE. num_cores should not exceed the number of cores the user's machine can handle. Default is 2.
nfolds	Numeric integer indicating number of cross-validation folds. Default is 3.

lambda	Numeric vector indicating lambda used for cross-validation for HierNet fit. Default lambda=c(20,30,40).
tol	Numeric value indicating acceptable tolerance for terminating optimization fit for HierNet. Default is tol=1e-3. WARNING: Do not increase as it greatly increases computation time.
speedup	Boolean indicating whether to employ computational tricks to make function run faster. It is always recommended to use default speedup=TRUE.
seed	Seed used for CRT procedure
verbose	Boolean indicating verbose output. Default verbose=TRUE

Value

A list containing:

p_val	A numeric value for the p-value testing profile order effect.
obs_test_stat	A numeric value for the observed test statistic.
resampled_test_stat	Matrix containing all the B resampled test statistics
tol	Tolerance used for HierNet
lam	Best cross-validated lambda
hiernet_fit	An object of class hiernet that contains the hiernet fit for the observed test statistic
seed	Seed used
elapsed_time	Elapsed time

References

Ham, D., Janson, L., and Imai, K. (2022) Using Machine Learning to Test Causal Hypotheses in Conjoint Analysis

Examples

```
# Subset of Immigration Choice Conjoint Experiment Data from Hainmueller et. al. (2014).
data("immigrationdata")
form = formula("Y ~ FeatEd + FeatGender + FeatCountry + FeatReason + FeatJob +
FeatExp + FeatPlans + FeatTrips + FeatLang + ppage + ppeducat + ppethm + ppgender")
left = colnames(immigrationdata)[1:9]
right = colnames(immigrationdata)[10:18]

# Testing if profile order effect is present or not in immigration data

profileorder_test = CRT_profileordereffect(formula = form, data = immigrationdata,
 left = left, right = right, B = 50)
profileorder_test$p_val
```

CRT_pval *Testing whether factor X matters in Conjoint Experiments*

Description

This function takes a conjoint dataset and returns the p-value when using the CRT to test if Y is independent of X given Z using HierNet test statistic. The function requires user to specify the outcome, all factors used in the conjoint experiment, and any additional respondent characteristics. By default, this function assumes a uniform randomization of factor levels. In addition, the function assumes the forced choice conjoint experiment and consequently assumes the data to contain the left and right profile factors in separate columns in the supplied dataset.

Usage

```
CRT_pval(
  formula,
  data,
  X,
  left,
  right,
  design = "Uniform",
  p = NULL,
  constraint_randomization = NULL,
  supplyown_resamples = NULL,
  profileorder_constraint = TRUE,
  in_levs = NULL,
  forced_var = NULL,
  non_factor = NULL,
  B = 200,
  parallel = TRUE,
  num_cores = 2,
  nfolds = 3,
  lambda = c(20, 30, 40),
  tol = 0.001,
  speedup = TRUE,
  seed = sample(c(1:1000), size = 1),
  analysis = 0,
  verbose = TRUE
)
```

Arguments

formula	A formula object specifying the outcome variable on the left-hand side and factors of (X,Z) and respondent characteristics (V) in the right hand side. RHS variables should be separated by + signs and should only contain either left or right for each (X,Z). For example Y ~ Country_left + Education_left is sufficient as opposed to Y ~ Country_left + Country_right + Education_left + Education_right

`data` A dataframe containing outcome variable and all factors (X,Z,V) (including both left and right profile factors). All (X,Z,V) listed in the formula above are expected to be of class factor unless explicitly stated in non_factor input.

`X` Character string specifying the variable of interest. This character should match column name in data. For example "Country_left" is sufficient.

`left` Vector of column names of data that corresponds to the left profile factors

`right` Vector of column names of data that corresponds to the right profile factors. NOTE: left and right are assumed to be the same length and the order should correspond to the same variables. For example left = c("Country_left", "Education_left") and right = c("Country_right", "Education_right")

`design` A character string of one of the following options: "Uniform", "Constrained Uniform", "Nonuniform", "Manual". "Uniform" refers to a completely uniform design where all (X,Z) are sampled uniformly. "Nonuniform" refers to a design where all (X,Z) are sampled independently but the levels of X are not sampled uniformly. If design="Nonuniform", then user should supply the non-uniform probability weights in p. If in_levs is not NULL, then length of p should match in_levs. "Constrained Uniform" refers to a dependent randomization design where some levels of X are only possible based on certain levels of Z. If design="Constrained Uniform" user should supply constraint_randomization list indicating the dependencies. See examples below. "Manual" refers to more complex conjoint designs, where the user will supply their own resamples in supplyown_resamples input. Default is design="Uniform"

`p` A vector of nonuniform probability weights used when design="Nonuniform". Length of p should match number of levels of X or length of in_levs.

`constraint_randomization`
List containing levels of X that can only be sampled with certain values of Z (used when design="Constrained Uniform"). The first element of constraint_randomization should contain the levels of X that can only be sampled with certain values of Z, which are included in the second element of the list. See example below.

`supplyown_resamples`
List of length B that contains own resamples of X when design="Manual". Each element of list should contain a dataframe with the same number of rows of data and two columns for the left and right profile values of X.

`profileorder_constraint`
Boolean indicating whether to enforce profile order constraint (default = TRUE)

`in_levs` A vector of strings which indicates a subset of the levels of X to test for. See example below.

`forced_var` A character string indicating column name of Z or V that user wishes to force an interaction with.

`non_factor` A vector of strings indicating columns of data that are not factors. This should only be used for respondent characteristics (V) that are not factors. For example non_factor = "Respondent_Age".

`B` Numeric integer value indicating the number of resamples for the CRT procedure. Default value is B=200.

`parallel` Boolean indicating whether parallel computing should be used. Default value is TRUE.

num_cores	Numeric integer indicating number of cores to use when parallel=TRUE. num_cores should not exceed the number of cores the user's machine can handle. Default is 2.
nfolds	Numeric integer indicating number of cross-validation folds. Default is 3.
lambda	Numeric vector indicating lambda used for cross-validation for HierNet fit. Default lambda=c(20,30,40).
tol	Numeric value indicating acceptable tolerance for terminating optimization fit for HierNet. Default is tol=1e-3. WARNING: Do not increase as it greatly increases computation time.
speedup	Boolean indicating whether to employ computational tricks to make function run faster. It is always recommended to use default speedup=TRUE.
seed	Seed used for CRT procedure
analysis	Numeric integer indicating whether to return the top x number of strongest interactions that contributed to the the observed test statistic. Default analysis = 0 to not return any top interactions. If analysis > 0, for example analysis = 2, then the top two strongest interactions contribution to the test statistic along with which interaction is returned. NOTE: this is purely for exploratory analysis.
verbose	Boolean indicating verbose output. Default verbose=TRUE

Value

A list containing:

p_val	A numeric value for the p-value testing Y independent of X given Z.
obs_test_stat	A numeric value for the observed test statistic. If analysis is > 0, obs_test_stat will contain a list detailing the contribution of the main effects interaction effects and the top interactions.
resampled_test_stat	Matrix containing all the B resampled test statistics
tol	Tolerance used for HierNet
lam	Best cross-validated lambda
hiernet_fit	An object of class hiernet that contains the hiernet fit for the observed test statistic
seed	Seed used
elapsed_time	Elapsed time

References

Ham, D., Janson, L., and Imai, K. (2022) Using Machine Learning to Test Causal Hypotheses in Conjoint Analysis

Examples

```
# Subset of Immigration Choice Conjoint Experiment Data from Hainmueller et. al. (2014).
data("immigrationdata")
form = formula("Y ~ FeatEd + FeatGender + FeatCountry + FeatReason + FeatJob +
FeatExp + FeatPlans + FeatTrips + FeatLang + ppage + ppeducat + ppethm + ppgender")
left = colnames(immigrationdata)[1:9]
right = colnames(immigrationdata)[10:18]


# Testing whether edcuation matters for immigration preferences
education_test = CRT_pval(formula = form, data = immigrationdata, X = "FeatEd",
 left = left, right = right, non_factor = "ppage", B = 100, analysis = 2)
education_test$p_val


# Testing whether job matters for immigration preferences
constraint_randomization = list() # (Job has dependent randomization scheme)
constraint_randomization[["FeatJob"]] = c("Financial analyst","Computer programmer",
"Research scientist","Doctor")
constraint_randomization[["FeatEd"]] = c("Equivalent to completing two years of
college in the US", "Equivalent to completing a graduate degree in the US",
 "Equivalent to completing a college degree in the US")

job_test = CRT_pval(formula = form, data = immigrationdata, X = "FeatJob",
left = left, right = right, design = "Constrained Uniform",
constraint_randomization = constraint_randomization, non_factor = "ppage", B = 100)
job_test$p_val



# Testing whether Mexican and European immigrants are treated indistinguishably
country_data = immigrationdata
country_data$FeatCountry = as.character(country_data$FeatCountry)
country_data$FeatCountry_2 = as.character(country_data$FeatCountry_2)
country_data$FeatCountry[country_data$FeatCountry %in% c("Germany", "France",
"Poland")] = "Europe"
country_data$FeatCountry_2[country_data$FeatCountry_2 %in% c("Germany", "France",
 "Poland")] = "Europe"
country_data$FeatCountry = factor(country_data$FeatCountry)
country_data$FeatCountry_2 = factor(country_data$FeatCountry_2)

mexico_Europe_test = CRT_pval(formula = form, data = country_data, X = "FeatCountry",
left = left, right = right, design = "Nonuniform",
in_levs = c("Mexico", "Europe"), p = c(0.25, 0.75), non_factor = "ppage", B = 100,
analysis = 2)


# example case with supplying own resamples
resample_Mexico_Europe = function(country_data) {
 resamples_1 = sample(c("Mexico", "Europe"), size = nrow(country_data),
 replace = TRUE, p = c(0.25, 0.75))
 resamples_2 = sample(c("Mexico", "Europe"), size = nrow(country_data),
```

```
 replace = TRUE, p = c(0.25, 0.75))
 resample_df = data.frame(resamples_1, resamples_2)
 return(resample_df)
}
own_resamples = list()
for (i in 1:100) {
   own_resamples[[i]] = resample_Mexico_Europe(country_data)
}
mexico_Europe_test = CRT_pval(formula = form, data = country_data, X = "FeatCountry",
left = left, right = right, design = "Manual",
in_levs = c("Mexico", "Europe"), supplyown_resamples = own_resamples,
non_factor = "ppage", B = 100, analysis = 2)

# example case with forcing with candidate gender

mexico_Europe_test_force = CRT_pval(formula = form, data = country_data,
X = "FeatCountry", left = left, right = right, design = "Nonuniform",
in_levs = c("Mexico", "Europe"), p = c(0.25, 0.75), forced_var = "FeatGender",
non_factor = "ppage", B = 100)
```

`immigrationdata` *Immigration Choice Conjoint Experiment Data from Hainmueller et. al. (2014).*

Description

This dataset is a subset of the first 1000 rows chosen from the original immigration dataset. Each row consists of a pair of immigrant candidates that were shown to respondents. The respondent then chooses either the left or right profile (main binary response) from the nine profile factors shown for each candidate. Respondent characteristics are also recorded. For example, the first row shows that the left immigrant candidate was a Male from Iraq who had a high school degree, etc. while the right immigrant candidate was a Female from France with no formal education, etc. The respondent who evaluated this task was a 20 year old college educated White Male, who voted for the left immigrant candidate.

Usage

```
immigrationdata
```

Format

A data frame with 1000 rows and 23 variables:

FeatEd Education of left candidate containing levels: "College Degree", "Graduate Degree", "Eighth Grade", "Fourth Grade", "High School", "Two Years of College", and "No Education"

FeatGender Gender of left candidate containing levels: "Female" and "Male"

FeatCountry Country of origin of left candidate containing levels: "Poland", "France", "Iraq", "Somalia", "Sudan", "China", "Mexico", "Germany", "Philippines", and "India"

FeatReason Reason for immigration of left candidate containing levels: "Escape political/religious persecution", "Reunite with family members", and "Seek better job"

FeatJob Occupation of left candidate containing levels: "Waiter", "Child care provider", "Teacher", "Nurse", "Construction worker", "Janitor", "Gardener", "Financial analyst", "Computer programmer", "Research scientist", and "Doctor"

FeatExp Job Experience of left candidate containing levels: "More than five years", "No job training", "One-two years", and "Three-five years"

FeatPlans Job Plans of left candidate containing levels: "Does not contract with U.S. employer but have job interview", "No Contract", "No plans", and "Will look for work after arriving"

FeatTrips Trips to U.S. of left candidate containing levels: "Entered U.S. once without legal authorization", "Entered U.S. once before on tourist visa", "Multiple visits on tourist visa", "Never visited", and "Spent six months with family in U.S."

FeatLang Language of left candidate containing levels: "Used interpreter", "Broken English", "Unable to speak English", and "Fluent English"

FeatEd_2 Education of right candidate containing levels: "College Degree", "Graduate Degree", "Eighth Grade", "Fourth Grade", "High School", "Two Years of College", and "No Education"

FeatGender_2 Gender of right candidate containing levels: "Female" and "Male"

FeatCountry_2 Country of origin of right candidate containing levels: "Poland", "France", "Iraq", "Somalia", "Sudan", "China", "Mexico", "Germany", "Philippines", and "India"

FeatReason_2 Reason for immigration of right candidate containing levels: "Escape political/religious persecution", "Reunite with family members", and "Seek better job"

FeatJob_2 Occupation of right candidate containing levels: "Waiter", "Child care provider", "Teacher", "Nurse", "Construction worker", "Janitor", "Gardener", "Financial analyst", "Computer programmer", "Research scientist", and "Doctor"

FeatExp_2 Job Experience of right candidate containing levels: "More than five years", "No job training", "One-two years", and "Three-five years"

FeatPlans_2 Job Plans of right candidate containing levels: "Does not contract with U.S. employer but have job interview", "No Contract", "No plans", and "Will look for work after arriving"

FeatTrips_2 Trips to U.S. of right candidate containing levels: "Entered U.S. once without legal authorization", "Entered U.S. once before on tourist visa", "Multiple visits on tourist visa", "Never visited", and "Spent six months with family in U.S."

FeatLang_2 Language of right candidate containing levels: "Used interpreter", "Broken English", "Unable to speak English", and "Fluent English"

ppage Respondent age (numeric variable)

ppeducat Respondent education containing levels

ppethm Respondent ethnicity

ppgender Respondent gender

Y Binary response variable Y: 1 if left profile is selected and 0 otherwise

Index

∗ **datasets**
immigrationdata, 15

CRT_carryovereffect, 2
CRT_fatigueeffect, 6
CRT_profileordereffect, 8
CRT_pval, 11

immigrationdata, 15