Package 'kcmeans'

November 30, 2023

Title Conditional Expectation Function Estimation with K-Conditional-Means

Version 0.1.0

Date 2023-11-28

Description Implementation of the KCMeans regression estimator studied by Wiemann (2023) <arXiv:2311.17021> for expectation function estimation conditional on categorical variables. Computation leverages the unconditional KMeans implementation in one dimension using dynamic programming algorithm of Wang and Song (2011) <doi:10.32614/RJ-2011-015>, allowing for global solutions in time polynomial in the number of observed categories.

License GPL (>= 3)

URL https://github.com/thomaswiemann/kcmeans

BugReports https://github.com/thomaswiemann/kcmeans/issues

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 3.6)

Imports stats, Ckmeans.1d.dp, MASS, Matrix

Suggests testthat (>= 3.0.0), covr, knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Thomas Wiemann [aut, cre]

Maintainer Thomas Wiemann <wiemann@uchicago.edu>

Repository CRAN

Date/Publication 2023-11-30 10:50:02 UTC

R topics documented:

`kcmeans` *K-Conditional-Means Estimator*

Description

Implementation of the K-Conditional-Means estimator.

Usage

```
kcmeans(y, X, which_is_cat = 1, K = 2)
```

Arguments

`y`	The outcome variable, a numerical vector.
`X`	A (sparse) feature matrix where one column is the categorical predictor.
`which_is_cat`	An integer indicating which column of `X` corresponds to the categorical predictor.
`K`	The number of support points, an integer greater than 2.

Value

`kcmeans` returns an object of S3 class `kcmeans`. An object of class `kcmeans` is a list containing the following components:

`cluster_map` A matrix that characterizes the estimated predictor of the residualized outcome $\tilde{Y} \equiv Y - X_{2:}^{\top}\hat{\pi}$. The first column `x` denotes the value of the categorical variable that corresponds to the unrestricted sample mean `mean_x` of $\tilde{Y}$, the sample share `p_x`, the estimated cluster `cluster_x`, and the estimated restricted sample mean `mean_xK` of $\tilde{Y}$ with just `K` support points.

`mean_y` The unconditional sample mean of $\tilde{Y}$.

`pi` The best linear prediction coefficients of Y on X corresponding to the non-categorical predictors $X_{2:}$.

`which_is_cat`,`K` Passthrough of user-provided arguments. See above for details.

References

Wang H and Song M (2011). "Ckmeans.1d.dp: optimal k-means clustering in one dimension by dynamic programming." The R Journal 3(2), 29–33.

Wiemann T (2023). "Optimal Categorical Instruments." https://arxiv.org/abs/2311.17021

Examples

```
# Simulate simple dataset with n=800 observations
X <- rnorm(800) # continuous predictor
Z <- sample(1:20, 800, replace = TRUE) # categorical predictor
Z0 <- Z %% 4 # lower-dimensional latent categorical variable
y <- Z0 + X + rnorm(800) # outcome
# Compute kcmeans with four support points
kcmeans_fit <- kcmeans(y, cbind(Z, X), K = 4)
# Print the estimated support points of the categorical predictor
print(unique(kcmeans_fit$cluster_map[, "mean_xK"]))
```

`predict.kcmeans` *Prediction Method for the K-Conditional-Means Estimator.*

Description

Prediction method for the K-Conditional-Means estimator.

Usage

```
## S3 method for class 'kcmeans'
predict(object, newdata, clusters = FALSE, ...)
```

Arguments

`object`	An object of class `kcmeans`.
`newdata`	A (sparse) feature matrix where the first column corresponds to the categorical predictor.
`clusters`	A boolean indicating whether estimated clusters should be returned.
`...`	Currently unused.

Value

A numerical vector with predicted values (if `clusters = FALSE`) or predicted clusters (if `clusters = FALSE`).

References

Wiemann T (2023). "Optimal Categorical Instruments." https://arxiv.org/abs/2311.17021

Examples

```
# Simulate simple dataset with n=800 observations
X <- rnorm(800) # continuous predictor
Z <- sample(1:20, 800, replace = TRUE) # categorical predictor
Z0 <- Z %% 4 # lower-dimensional latent categorical variable
y <- Z0 + X + rnorm(800) # outcome
# Compute kcmeans with four support points
```

```
kcmeans_fit <- kcmeans(y, cbind(Z, X), K = 4)
# Calculate in-sample predictions
fitted_values <- predict(kcmeans_fit, cbind(Z, X))
# Print sample share of estimated clusters
clusters <- predict(kcmeans_fit, cbind(Z, X), clusters = TRUE)
table(clusters)
```

Index