# Package 'RcppDPR'

March 19, 2025

**Type** Package

**Title** 'Rcpp' Implementation of Dirichlet Process Regression

**Version** 0.1.10

**Description** 'Rcpp' reimplementation of the the Bayesian non-parametric Dirichlet Process Regression model for penalized regression first published in Zeng and Zhou (2017) <doi:10.1038/s41467-017-00470-2>. A full Bayesian version is implemented with Gibbs sampling, as well as a faster but less accurate variational Bayes approximation.

**License** GPL-3

**Encoding** UTF-8

**Imports** Rcpp (>= 1.0.13)

**LinkingTo** Rcpp, RcppArmadillo, RcppGSL

**Suggests** testthat (>= 3.0.0), snpStats

**Config/testthat/edition** 3

**RoxygenNote** 7.3.2

**NeedsCompilation** yes

**Author** Mohammad Abu Gazala [cre, aut],
Daniel Nachun [ctb],
Ping Zeng [ctb]

**Maintainer** Mohammad Abu Gazala <abugazalamohammad@gmail.com>

**Repository** CRAN

**Date/Publication** 2025-03-19 15:00:08 UTC

## Contents

`fit_model` *Fit Dirichlet Process Regression model*

## Description

Fit a Dirichlet Process Regression model using a specified fitting method. Outcome (y) should be Gaussian and scaled and centered; predictors (x) and covariates (w) should also be scaled and centered but may be of any distribution

## Usage

```
fit_model(
  y,
  w,
  x,
  rotate_variables = FALSE,
  covariance_matrix = NULL,
  fitting_method = "VB",
  ...
)
```

## Arguments

| | |
|---|---|
| `y` | Numeric vector of outcome |
| `w` | Numeric matrix of covariates (default = rep(1, length(y))) |
| `x` | Numeric matrix of predictors |
| `rotate_variables` | Logical value indicating whether to rotate y, w and x using covariance_matrix (default = FALSE) |
| `covariance_matrix` | Numeric sample covariance matrix used for rotation of y, w and x - if NULL and rotate_variables is TRUE then the sample covariance matrix is computed from x |
| `fitting_method` | Character string indicating the method used for fitting the data - possible values are:<br>• 'Gibbs' - full Bayesian inference with Gibbs sampler with a fixed n_k<br>• 'Adaptive_Gibbs' - adaptive version of Gibbs sample that automatically chooses n_k<br>• 'VB' - variational Bayes inference with a fixed n_k |
| `...` | arguments to pass through to internal methods. |

### Details

fit_model() can pass a number of additional parameters to the different fitting methods. These parameters are used for all modes:

- n_k: number of mixture components in scale mixture of normals prior (default = 4)
- l_min: minimum value of log-likelihood for initial parameter search (default = 1e-7, only modify if you know what you are doing)
- l_max: maximum value of log-likelihood for initial parameter search (default = 1e5, only modify if you know what you are doing)
- n_regions: number of regions over which to search for maximum log-likelihood (default = 10, only modify if you know what you are doing)

These parameters are only used for the Gibbs and Adaptive Gibbs modes:

- w_step: number of burn-in steps for Gibbs sampler (default = 1000)
- s_step: number of inference steps for Gibbs sampler (default = 1000)
- m_n_k: maximum number of mixture components in scale mixture of normals prior (default = 6, Adaptive Gibbs only)

### Value

returns an object of class 'DPR_Model'

### Examples

```
file_path_x <- system.file("extdata", "data/in/x.rds", package = "RcppDPR")
file_path_y <- system.file("extdata", "data/in/y.rds", package = "RcppDPR")
file_path_w <- system.file("extdata", "data/in/w.rds", package = "RcppDPR")
x = readRDS(file_path_x)
y = readRDS(file_path_y)
w = readRDS(file_path_w)
dpr_model <- fit_model(y, w, x, fitting_method = "VB")
```

---

## predict.DPR_Model *Use a DPR model to predict results from new data*

---

### Description

Use a DPR model to predict results from new data

### Usage

```
## S3 method for class 'DPR_Model'
predict(object, newdata, ...)
```

## Arguments

| | |
|---|---|
| object | an object of class DPR_Model |
| newdata | Numeric matrix representing the input to the model |
| ... | ignored args. |

## Value

returns Numeric vector of predictions

## Examples

```
n <- 500
p <- 10775
file_path_x <- system.file("extdata", "data/in/x.rds", package = "RcppDPR")
file_path_y <- system.file("extdata", "data/in/y.rds", package = "RcppDPR")
file_path_w <- system.file("extdata", "data/in/w.rds", package = "RcppDPR")
x = readRDS(file_path_x)
y = readRDS(file_path_y)
w = readRDS(file_path_w)
dpr_model <- fit_model(y, w, x, fitting_method = "VB")
new_x <- matrix(rnorm(n = n * p, mean = 0, sd = 1), nrow = n, ncol = p)
new_y <- predict(dpr_model, new_x)
```

# Index