Package 'pip'

November 3, 2021

Title Platelet Inventory Prediction

Version 0.3-8

VignetteBuilder knitr

URL https://github.com/okadak126/pip

BugReports https://github.com/okadak126/pip

Description Forecast inventory of platelet or any other item based on past history and projected usage. Facilities are provided for indicating forecast window and length of past history to use. A combination of time series analysis together with linear programming is used to optimize a criterion.

Depends R (>= 3.4.2)

License GPL (>= 2)

Encoding UTF-8

LazyData true

Suggests knitr,
rmarkdown,
testthat,
shiny

RoxygenNote 7.1.1.9001

Imports dplyr, lpSolveAPI, utils

Author Leying Guan [aut],
Robert J. Tibshirani [aut],
Saurabh Gombar [aut],
Xiaoying Tian [ctb],
Robert Scott [ctb],
Alison J. Zemek [ctb],
Tho Duc Pham [aut],
Balasubramanian Narasimhan [aut, cre],
Gomathi Krishnan [ctb]

Maintainer Balasubramanian Narasimhan <naras@stat.stanford.edu>

R topics documented:

`pip-package` *pip: Platelet Inventory Prediction using Linear Programming*

Description

Forecast inventory of platelet or any other item based on past history and projected usage. Facilities are provided for indicating forecast window and length of past history to use. A combination of time series analysis together with linear programming is used to optimize a criterion.

Author(s)

Leying Guan, Robert J. Tibshirani, Balasubramanian Narasimhan, Alison Zemek, Xiaoying Tian, Robert Scott, Saurabh Gombar, Tho Duc Pham, Gomathi Krishnan

Maintainer: Robert J. Tibshirani<tibs@stanford.edu>

`build_model` *Run a fit on a given dataset*

Description

The dataset should have one more column than the predictors and response. This additional column, which is typically a date or day number, has to be named "date" or "day"

Usage

```
build_model(
  data,
  c0 = 15,
  history_window = 200,
  penalty_factor = 5,
  rss_bias = 30,
  start = 10,
  l1_bounds = seq(from = 200, to = 0, by = -2),
  lag_bounds = -1,
  date_column = "day|date",
  response_column = "plt_used",
  show_progress = TRUE,
  plot_losses = TRUE
)
```

Arguments

data	the dataset
c0	the lower bound on remaining "fresh" inventory (used in optimization)
history_window	number of days to look back
penalty_factor	penalty for shortage specified by doctors
rss_bias	amount by which we should bias the RSS calculation upward for the loss
start	the first day in the dataset that the model's predictions are evaluated
l1_bounds	vector containing the possible values of the l1 bound on coefs aside from days of the week and seven day moving average
lag_bounds	vector containing possible values of the bound on the seven day moving average (lag) coefficient
date_column	the name of the date or day number column as a regex, default is "day|date" i.e. day or date
response_column	the name of the response column, default is "plt_used"
show_progress	a TRUE/FALSE flag for showing progress, default TRUE
plot_losses	a TRUE/FALSE flag for whether we plot loss values by hyperparameter (helpful for model diagnostics). Default true.

Examples

```
# runs single_lpSolve to produce the model with the lowest CV loss using
# hyperparameters l1_bounds and lag_bounds (LASSO penalty)
model <- build_model(lpdata, c0 = 15, history_window = 100, penalty_factor = 5,
                     rss_bias = 30, l1_bounds = l1_bounds, lag_bounds = lag_bounds)

# generate single total platelet usage prediction for the next 3 days.
pred <- predict_three_day_sum(model, newdata)
```

---

compute_loss *Compute the Cross-Validation Loss for each hyperparameter*

Description

Each column of preds, w, r1, r2, s, represent a single hyperparameter value. These inputs should be generated from the cross_validate function

Usage

```
compute_loss(preds, y, w, r1, r2, s, penalty_factor, rss_bias)
```

Arguments

preds	a vector of 3 day usage predictions
y	a vector of actual daily usage
w	a vector of waste determined by pip::compute_prediction_statistics
r1	a vector of remaining inventory determined by pip::compute_prediction_statistics
r2	a vector of remaining inventory determined by pip::compute_prediction_statistics
s	a vector of inventory shortage as determined by pip::compute_prediction_statistics
penalty_factor	factor to additionally penalize shortage terms over waste
rss_bias	amount by which we bias the RSS term upward to prevent shortage

Value

a vector of losses computed for each hyperparameter

---

compute_prediction_statistics *Compute the prediction statistics*

Description

Compute three-day predicted usage, waste, remaining and shortage using the actual usage and predicted usage.

Usage

```
compute_prediction_statistics(
  y,
  t_pred,
  initial_expiry_data = c(0, 0),
  initial_collection_data = c(60, 60, 60),
  start = 10,
  c = 30
)
```

Arguments

y	is the number of units used at the current day i
t_pred	is the sum of the predicted number of units for days i+1, i+2, i+3
initial_expiry_data	the number of units that can be used the current day and the next (2-length vector)
initial_collection_data	is the number of units to collect for days i, i+1, i+2
start	is when we start the clock. So for example, in this default invocation the shelf storage (initial_collection_data) was 60, 60, 60, on days 10, 11, and 12. So the evaluation of the model begins at day 11 but the decision to order collection starts on day 13.
c	the ideal minimum number of fresh units remaining at the end of the day. this is also the minimum number that should be collected (let's remove)

Value

a list with four components, `x` is the number of units to collect, `r` is a matrix of two columns indicating remaining units usable on day i+1 and day i+2, `w` is waste, and `s` is shortage.

`create_folds` *Create folds for time series cross validation. Folds are continguous and sequential.*

Description

Create folds for time series cross validation. Folds are continguous and sequential.

Usage

```
create_folds(n, fold_len)
```

Arguments

`n`	number of rows of a matrix/data_frame to which ids are assigned
`fold_len`	length of a fold

Value

a vector of integers assigning values in 1 - nfold to each row.

`cross_validate` *Run cross-validation method*

Description

Run cross-validation method

Usage

```
cross_validate(
  data,
  pred_var_indices,
  resp_var_index,
  l1_bounds,
  lag_bounds,
  c0,
  start,
  penalty_factor,
  seed_prop,
  fold_size,
  cv_type = "loo",
  show_progress = FALSE
)
```

Arguments

data	the full dataset (data.frame) used as input to the model
pred_var_indices	the column indices corresponding to the features in "data"
resp_var_index	the (single) column index corresponding to the response in "data
l1_bounds	vector containing the possible values of the l1 bound on coefs aside from days of the week and seven day moving average
lag_bounds	vector containing possible values of the bound on the seven day moving average (lag) coefficient
c0	the lower bound on remaining "fresh" inventory (used in optimization)
start	the first day in the dataset that the model's predictions are evaluated
penalty_factor	how much we penalize shortage over waste
seed_prop	the proportion of seed data (treated as fold 1 for training)
fold_size	the desired size of each fold (aside from the seed fold 1)
cv_type	the variety of cross_validation used. The model can either train on all previous folds (exp for exanding) or on all other folds besides the left-out fold (loo for leave-one-out, default)
show_progress	a TRUE/FALSE flag for showing progress, default FALSE

Value

a list containing matrices for waste, r1, r2, shortage, and predictions from cross-validation

---

evaluate_model *Evaluate the model on a held out validation set within the original training set*

Description

The dataset should have one more column than the predictors and response. This additional column, which is typically a date or day number, has to be named "date" or "day"

Usage

```
evaluate_model(
  data,
  c0 = 15,
  train_window = 200,
  test_window = 7,
  penalty_factor = 5,
  rss_bias = 30,
  start = 10,
  l1_bounds = seq(from = 200, to = 0, by = -2),
  lag_bounds = c(NA),
  date_column = "day|date",
  response_column = "plt_used"
)
```

Arguments

data	the dataset
c0	the c0 value
train_window	number of days to look back
test_window	number of days on which to evaluate model
penalty_factor	penalty for shortage specified by doctors
rss_bias	amount by which we bias the RSS term of the loss upward
start	the day you start evaluating the model (for CV)
l1_bounds	vector containing the possible values of the l1 bound on coefs aside from days of the week and seven day moving average
lag_bounds	vector containing possible values of the bound on the seven day moving average (lag) coefficient
date_column	the name of the date or day number column as a regex, default is "day|date" i.e. day or date
response_column	the name of the response column, default is "plt_used"

Value

the lowest CV loss obtained on the validation set

Examples

```
load("data/sample_data.Rdata")

# evaluate CV performance of model on last [test_window] days vs. best hyperparameter choice
# after training on the first [train_window] days of input data
min_loss <- evaluate_model(lpdata, c0 = 10, train_window = 86, test_window = 14, penalty_factor = 15,
                           rss_bias = 10, l1_bounds = l1_bounds, lag_bounds = lag_bounds)
```

---

pos *Return the positive part of a value*

Description

Return the positive part of a value

Usage

```
pos(x)
```

Arguments

x	the vector

Value

the maximum of x and 0

predict_three_day_sum *Predict the next three day sum and units to order if available*

Description

Predict the next three day sum and units to order if available

Usage

```
predict_three_day_sum(model, new_data)
```

Arguments

model	the trained model
new_data	a new data frame with the same predictor names as in the training data, even if one row

Value

the predicted three day total of how many units to collect

Examples

```
load("data/sample_data.Rdata")

# runs single_lpSolve to produce the model with the lowest CV loss using
# hyperparameters l1_bounds and lag_bounds (LASSO penalty)
model <- build_model(lpdata, c0 = 15, history_window = 100, penalty_factor = 5,
                     rss_bias = 30, l1_bounds = l1_bounds, lag_bounds = lag_bounds)

# generate single total platelet usage prediction for the next 3 days.
pred <- predict_three_day_sum(model, newdata)
```

single_lpSolve *Solve the LP problem for platelet usage prediction*

Description

Solve the LP problem for platelet usage prediction

Usage

```
single_lpSolve(
  d,
  l1_bounds,
  lag_bounds,
  num_vars,
  start = 10,
  c = 10,
```

```
  shortage_factor = 15,
  buffer = 10,
  ind = NULL,
  reset_basis = FALSE
)
```

Arguments

d	the data frame. We assume that the first column contains a date and the response is the last column.
l1_bounds	the upper bound on the l1 norm of coefficient vector (aside from lag and days of week)
lag_bounds	the upper bound on the seven day moving average of usage coefficient (lag)
num_vars	the number of features/covariates in the dataset.
start	the 0th index of the date when we begin evaluating the model
c	the value for c_0
shortage_factor	the amount by which we multiply shortage over waste (ratio)
buffer	a value to be added to account for training in the initial stages, default 10
ind	the indices for cross validation used only if non-null. (Should be sorted!!)
reset_basis	TRUE or FALSE for whether we reset the LP basis before each solve (for non-uniqueness)

Examples

```
# Note that SBCpip::create_dataset() produces data of the form lpdata from source files
load("data/sample_data.Rdata")
sol <- single_lpSolve(lpdata, l1_bounds, lag_bounds, num_vars)

# Add the coefficient names to the resulting coefficients. Each
# column represents a pairing of hyperparameter values from l1_bounds
# and lag_bounds. The effects of the L1 penalty should be apparent.
rownames(sol) <- c("Intercept", colnames(lpdata[2:(ncol(lpdata) - 1)]))
```

---

single_lpSolve_wasteOnly

Solve the LP problem for platelet usage prediction (ignoring shortage in the optimization)

Description

Solve the LP problem for platelet usage prediction (ignoring shortage in the optimization)

Usage

```
single_lpSolve_wasteOnly(
  d,
  l1_bounds,
  lag_bounds,
  num_vars,
  start = 10,
  c = 10,
  shortage_factor = 15,
  buffer = 10,
  ind = NULL,
  reset_basis = FALSE
)
```

Arguments

d	the data frame
l1_bounds	the upper bound on the l1 norm of coefficient vector (aside from lag and days of week)
lag_bounds	the upper bound on the seven day moving average of usage coefficient (lag)
num_vars	the number of features/covariates in the dataset.
start	the 0th index of the date when we begin evaluating the model
c	the value for c_0
shortage_factor	the amount by which we multiply shortage over waste (ratio)
buffer	a value to be added to account for training in the initial stages, default 10
ind	the indices for cross validation used only if non-null. (Should be sorted!!)
reset_basis	TRUE or FALSE for whether we reset the LP basis before each solve (for non-uniqueness)

Index