Package ‘texteffect’

March 24, 2019

Version 0.3
Date 2019-03-23
Title Discovering Latent Treatments in Text Corpora and Estimating Their Causal Effects
Author Christian Fong <christianfong@stanford.edu>
Maintainer Christian Fong <christianfong@stanford.edu>
Depends R (>= 3.3), MASS, boot, ggplot2
Imports

Description Implements the approach described in Fong and Grimmer (2016) <https://aclweb.org/anthology/P/P16/P16-1151.pdf> for automatically discovering latent treatments from a corpus and estimating the average marginal component effect (AMCE) of each treatment. The data is divided into a training and test set. The supervised Indian Buffet Process (sibp) is used to discover latent treatments in the training set. The fitted model is then applied to the test set to infer the values of the latent treatments in the test set. Finally, Y is regressed on the latent treatments in the test set to estimate the causal effect of each treatment.

LazyLoad yes
LazyData yes
License GPL (>= 2)
NeedsCompilation no
Repository CRAN
Date/Publication 2019-03-24 05:30:06 UTC

R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>BioSample</td>
<td>2</td>
</tr>
<tr>
<td>infer_Z</td>
<td>2</td>
</tr>
<tr>
<td>sibp</td>
<td>4</td>
</tr>
</tbody>
</table>
BioSample  

Sample from the Fong and Grimmer Wikipedia Biography Data

Description

This data set gives a small sample of the data used in “Discovery of Treatments from Text Corpora” by Christian Fong and Justin Grimmer. This sample is intended as a toy data set for use in the examples of this package’s documentation. A real data set should include far more observations.

Usage

BioSample

Format

A data frame consisting of 51 columns (including an outcome measure and counts for each word in a 50 word vocabulary) and 250 observations.

Source

Data collected using the Wikipedia API and an original survey experiment by Fong and Grimmer.

References


infer_Z  

Infer Treatments on the Test Set

Description

infer_Z uses an sibp object fitted on a training set to infer the treatments in a test set.

Usage

infer_Z(sibp.fit, X, newX = FALSE)
**Arguments**

- `sibp.fit` A sibp object.
- `X` The covariates for the data set where Z is to be inferred. Usually, the user should use the same X used to call the sibp function.
- `newX` Set to TRUE if the X supplied is not the training and test set. Used primarily for followup validation experiments. Defaults to FALSE.

**Details**

This function applies the mapping from words to treatments discovered in the training set to infer which observations have which treatments in the test set. Usually, users will be better served by calling `sibp_amce`, which calls this function internally before returning estimates and confidence intervals for the average marginal component effects.

**Value**

- `nu` Informally, the probability that the row document has the column treatment. Formally, the parameter for the variational approximation of \( z_{i,k} \), which is a Bernoulli distribution.

**Author(s)**

Christian Fong

**References**

Fong, Christian and Justin Grimmer. 2016. “Discovery of Treatments from Text Corpora” Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. [https://aclweb.org/anthology/P/P16/P16-1151.pdf](https://aclweb.org/anthology/P/P16/P16-1151.pdf)

**See Also**

- `sibp`, `sibp_amce`

**Examples**

```r
# Load the Wikipedia biography data
data(BioSample)

# Divide into training and test sets
Y <- BioSample[,1]
X <- BioSample[,-1]
set.seed(1)
train.ind <- sample(1:nrow(X), size = 0.5*nrow(X), replace = FALSE)

# Fit an sIBP on the training data
sibp.fit <- sibp(X, Y, K = 2, alpha = 4, sigmasq.n = 0.8,
                 train.ind = train.ind)

# Infer the latent treatments in the test set
```
Description

sibp discovers latent binary treatments within a corpus, as described by Fong and Grimmer (2016).

Usage

sibp(X, y, K, alpha, sigmasq.n, 
a = 0.1, b = 0.1, sigmasq.A = 5, 
train.ind, G = NULL, silent = FALSE )

Arguments

X The covariates for all observations in the training set, where each row is a document and each column is the count of a word.
Y The outcome for all observations in the training set.
K The number of treatments to be discovered.
alpha A parameter that influences how common the treatments are. When alpha is large, the treatments are common.
sigmasq.n A parameter determining the variance of the word counts conditional on the treatments. When sigmasq.n is large, the treatments must explain most of the variation in X.
a A parameter that, together with b, influences the variance of the treatment effects and the outcomes. a = 0.1 is a reasonably diffuse choice.
b A parameter that, together with a, influences the variance of the treatment effects and the outcomes. b = 0.1 is a reasonably diffuse choice.
sigmasq.A A parameter determining the variance of the effect of the treatments on word counts. A diffuse choice, such as 5, is usually appropriate.
train.ind The indices of the observations in the training set, usually obtained from get_training_set().
G An optional group membership matrix. The AMCE for a given treatment is permitted to vary as a function of the individual’s group.
silent If TRUE, prints how much the parameters have moved every 10 iterations of sIBP.

Details

Fits a supervised Indian Buffet Process using variational inference. Before running this function, the data should be divided into a training set and a test set. This function should be run on the training set to discover latent treatments in the data that seem to be correlated with the outcome. It is recommended to use linksibp_param_search instead of this function to search over multiple configurations of the most important parameters. So long as only the training data is used, the analyst can freely experiment with as many parameter configurations as he likes without corrupting his causal inferences. Once a parameter configuration is chosen, the user can then use sibp_amce on the test set to estimate the average marginal component effect (AMCE) for each treatment.
Value

\( \nu \) Informally, the probability that the row document has the column treatment. Formally, the parameter for the variational approximation of \( z_{i,k} \), which is a Bernoulli distribution.

\( m \) Informally, the effect of having each treatment on the outcome. Formally, the mean parameter for the variational approximation of the posterior distribution of beta, which is a normal distribution. Note that this is in the training sample, and it is inappropriate to use this posterior as the basis for causal inference. It is instead necessary to estimate effects using the test set, see `sibp_amce`.

S The variance parameter for the posterior distribution of beta, which is a normal distribution.

\( \lambda \) A matrix where the kth row contains the shape parameters for the variational approximation of the posterior distribution of \( \pi_k \), which is a beta distribution.

\( \phi \) Informally, the effect of the row treatment on the column word. Formally, the mean parameter for the variational approximation of the posterior distribution of A, which is a normal distribution.

\( \text{big} \cdot \Phi \) The variance parameter for the variational approximation of the posterior distribution of A, which is a normal distribution. The kth element of the list corresponds to a treatment k.

\( c \) The shape parameter for the variational approximation of the posterior distribution of \( \tau \), which is a gamma distribution.

\( d \) The rate parameter for the variational approximation of the posterior distribution of \( \tau \), which is a gamma distribution.

\( K \) The number of treatments.

\( D \) The number of words in the vocabulary.

\( \alpha \) The alpha used to call this function.

\( a \) The a used to call this function.

\( b \) The b used to call this function.

\( \text{sigmasq} \cdot A \) The sigmasq.A used to call this function.

\( \text{sigmasq} \cdot n \) The sigmasq.n used to call this function.

\( \text{train.ind} \) The indices of the observations in the training set.

\( \text{test.ind} \) The indices of the observations in the test set.

Author(s)

Christian Fong

References

See Also

sibp_param_search, sibp_top_words, sibp_amce

Examples

```r
# Load the Wikipedia biography data
data(BioSample)

# Divide into training and test sets
Y <- BioSample[,1]
X <- BioSample[,1:0]
set.seed(1)
train.ind <- sample(1:nrow(X), size = 0.5*nrow(X), replace = FALSE)

# Search sIBP for several parameter configurations; fit each to the training set
sibp.search <- sibp_param_search(X, Y, K = 2, alphas = c(2,4), sigmasq.ns = c(0.8, 1), iters = 1, train.ind = train.ind)

## Not run:
# Get metric for evaluating most promising parameter configurations
sibp_rank_runs(sibp.search, X, 10)

# Qualitatively look at the top candidates
sibp_top_words(sibp.search[["4"]][["0.8"]][[1]], colnames(X), 10, verbose = TRUE)
sibp_top_words(sibp.search[["4"]][["1"]][[1]], colnames(X), 10, verbose = TRUE)

# Select the most interest treatments to investigate
sibp.fit <- sibp.search[["4"]][["0.8"]][[1]]

# Estimate the AMCE using the test set
amce <- sibp_amce(sibp.fit, X, Y)

# Plot 95% confidence intervals for the AMCE of each treatment
sibp_amce_plot(amce)

## End(Not run)
```

---

### sibp_amce

**Infer Treatments on the Test Set**

**Description**

sibp_amce uses an sibp object fitted on a training set to estimate the AMCE with the test set.

**Usage**

```r
sibp_amce(sibp.fit, X, Y, G = NULL, seed = 0, level = 0.05, thresh = 0.5)
sibp_amce_plot(sibp.amce, xlab = "Feature", ylab = "Outcome", subs = NULL)
```
Arguments

- `sibp.fit` A sibp object.
- `X` The covariates for the full data set. The division between the training and test set is handled inside the function.
- `Y` The outcomes for the full data set. The division between the training and test set is handled inside the function.
- `G` A group membership matrix. The AMCE for a given treatment is permitted to vary as a function of the individual’s group.
- `seed` The seed
- `level` The level of the confidence intervals to be obtained.
- `thresh` The treatment will = 1 when nu >= thresh and 0 otherwise. This avoids problems due to misclassification error.
- `sibp.amce` The table returned by codesibp_amce.
- `xlab` The label for the x-axis of the plot.
- `ylab` The label for the y-axis of the plot.
- `subs` The subset of the coefficients to plot. By default, plots all coefficients.

Details

Nothing

Value

- `sibp.amce` A table where the first column is the index of the treatment, the second column ("effect") is the estimated AMCE, the third column ("L") is the lower bound of the confidence interval, and the fourth column ("U") is the upper bound of the confidence interval.

Author(s)

Christian Fong

References

Fong, Christian and Justin Grimmer. 2016. “Discovery of Treatments from Text Corpora” Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. [https://aclweb.org/anthology/P/P16/P16-1151.pdf](https://aclweb.org/anthology/P/P16/P16-1151.pdf)

See Also

- `sibp`
Examples

```r
# Load the sample of Wikipedia biography data
data(BioSample)

# Divide into training and test sets
Y <- BioSample[,1]
X <- BioSample[,-1]
set.seed(1)
train.ind <- sample(1:nrow(X), size = 0.5*nrow(X), replace = FALSE)

# Fit an sIBP on the training data
sibp.fit <- sibp(X, Y, K = 2, alpha = 4, sigmasq.n = 0.8,
                 train.ind = train.ind)

sibp.amce <- sibp_amce(sibp.fit, X, Y)
sibp_amce_plot(sibp.amce)
```

---

`sibp_exclusivity` *Calculate Exclusivity Metric*

**Description**

`sibp_exclusivity` calculates the coherence metric for an `sibp` object fit on a training set. `sibp_rank_runs` runs `sibp_exclusivity` on each element in the list returned by `sibp_param_search`, and ranks the parameter configurations from most to least promising.

**Usage**

```r
sibp_exclusivity(sibp.fit, X, num.words = 10)
sibp_rank_runs(sibp.search, X, num.words = 10)
```

**Arguments**

- `sibp.fit` A `sibp` object.
- `sibp.search` A list of `sibp` object fit using the training set, obtained using `sibp_param_search`.
- `X` The covariates for the full data set. The division between the training and test set is handled inside the function.
- `num.words` The top words whose coherence will be evaluated.

**Details**

The metric is formally described at the top of page 1605 of [https://aclweb.org/anthology/P/P16/P16-1151.pdf](https://aclweb.org/anthology/P/P16/P16-1151.pdf). The purpose of this metric is merely to suggest which parameter configurations might contain the most interesting treatments to test if there are too many configurations to investigate manually. The choice of the parameter configuration should always be made on the basis of which treatments are substantively the most interesting, see `sibp_top_words`. 
Value

| exclusivity | An exclusivity matrix which quantifies the degree to which the top words in a treatment appear in documents that have that treatment but not in documents that lack that treatment. |
| exclusivity_rank | A table that ranks the treatments discovered by the various runs from sibp.search from most exclusive to least exclusive. |

Author(s)

Christian Fong

References


See Also

sibp_param_search, sibp_top_words

Examples

```r
# Load the sample of Wikipedia biography data
data(BioSample)

# Divide into training and test sets
Y <- BioSample[,1]
X <- BioSample[,-1]
set.seed(1)
train.ind <- sample(1:nrow(X), size = 0.5*nrow(X), replace = FALSE)

# Search sIBP for several parameter configurations; fit each to the training set
sibp.search <- sibp_param_search(X, Y, K = 2, alphas = c(2,4),
                                 sigmasq.ns = c(0.8, 1), iters = 1,
                                 train.ind = train.ind)

# Get metric for evaluating most promising parameter configurations
sibp_rank_runs(sibp.search, X, 10)
```

Description

`sibp_param_search` runs `sibp` for a variety of parameter configurations, so that the user can then test the effects for the most interesting treatments.
sibp_param_search

Usage

sibp_param_search(X, Y, K, alphas, sigmasq.ns, iters,
                   a = 0.1, b = 0.1, sigmasq.A = 5, train.ind = train.ind,
                   G = NULL, seed = 0)

Arguments

X             The covariates for the full data set. The division between the training and test
              set is handled inside the function.
Y             The outcomes for the full data set. The division between the training and test set
              is handled inside the function.
K             The number of treatments to be discovered.
alphas        A vector of values of alpha to try.
sigmasq.ns    A vector of values of sigmasq.n to try.
iters         The number of starting values to attempt for each combination of alpha and
              sigmasq.n.
a             A parameter.
b             A parameter.
sigmasq.A    A parameter.
train.ind    The indices of the observations in the training set, usually obtained from get_training_set().
G             An optional group membership matrix. The AMCE for a given treatment is
              permitted to vary as a function of the individual’s group.
seed         The seed to be used, so the result can be replicated.

Details

Fits a supervised Indian Buffet Process using variational inference for combinations of alpha and
sigmasq.n. alpha influences how common the treatments are (where larger alphas imply more com-
mon treatments) and sigmasq.n influences how much of the variation of the outcome must be ex-
plained by the treatments. These parameters are the most important for determining the quality
of the treatments discovered, so it is usually a good idea to experiment with many combinations.
Because the treatments discovered can be sensitive to starting values, it is also usually a good idea
to try each combination of alpha and sigmasq.n several times by setting iters > 1.
Because this function uses only the training data, the user can experiment with many parameter
configurations without corrupting the inferences made with the test set. The choice of parameters
is equivalent to the choice of hypotheses to test, so the analyst should choose the parameter config-
uration that leads to the most substantively interesting treatments. sibp_top_words can be applied
to each element of the list returned by this function to determine which parameter configurations
lead to interesting treatments. Often, it will be impractical to manually investigate every parameter
configuration. In such cases, sibp_rank_runs can be used to automatically identify some of the most
promising candidates.

Value

paramslist
**sibp_top_words**

**Description**

`sibp_top_words` returns a data frame of the words most associated with each treatment.

**Usage**

`sibp_top_words(sibp.fit, words, num.words = 10, verbose = FALSE)`
Arguments

- **sibp.fit**: A sibp object.
- **words**: The actual words, usually obtained through colnames(X).
- **num.words**: The number of top words to report.
- **verbose**: If set to true, reports how common each treatment is (so that the analyst can focus on the common treatments) and how closely associated each word is with each treatment.

Details

The choice of the parameter configuration should always be made on the basis of which treatments are substantively the most interesting. This function provides one natural way of discovering which words are most associated with each treatment (the mean parameter for the posterior distribution of phi, where phi is the effect of the treatment on the count of word w) and therefore helps to determine which treatments are most interesting.

Value

- **top.words**: A data frame where each column consists of the top ten words (in order) associated with a given treatment.

Author(s)

Christian Fong

References

Fong, Christian and Justin Grimmer. 2016. “Discovery of Treatments from Text Corpora” Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. [https://aclweb.org/anthology/P/P16/P16-1151.pdf](https://aclweb.org/anthology/P/P16/P16-1151.pdf)

See Also

- **sibp**

Examples

```r
# Load the Wikipedia biography data
data(BioSample)

# Divide into training and test sets
Y <- BioSample[,1]
X <- BioSample[,-1]
set.seed(1)
train.ind <- sample(!:nrow(X), size = 0.5*nrow(X), replace = FALSE)

# Fit an sIBP on the training data
sibp.fit <- sibp(X, Y, K = 2, alpha = 4, sigmasq.n = 0.8, train.ind = train.ind)
```
sibp_top_words

sibp_top_words(sibp.fit, colnames(X))
Index

*Topic datasets
  BioSample, 2

BioSample, 2

infer_Z, 2

sibp, 3, 4, 7, 12
sibp_amce, 3–6, 6, 11
sibp_amce_plot (sibp_amce), 6
sibp_exclusivity, 8
sibp_param_search, 6, 9, 9
sibp_rank_runs, 10, 11
sibp_rank_runs (sibp_exclusivity), 8
sibp_top_words, 6, 8–11, 11