# Package ‘forestControl’

July 20, 2018

**Type** Package

**Title** Approximate False Positive Rate Control in Selection Frequency for Random Forest

**Version** 0.2.0

**Date** 2018-07-20

**Description** Approximate false positive rate control in selection frequency for random forest using the methods described by Ender Konukoglu and Melanie Ganz (2014) <arXiv:1410.2838>. Methods for calculating the selection frequency threshold at false positive rates and selection frequency false positive rate feature selection.

**Imports** Rcpp, purrr, tibble, magrittr, dplyr

**Suggests** testthat, randomForest, ranger

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**URL** `https://github.com/aberHRML/forestControl`

**BugReports** `https://github.com/aberHRML/forestControl/issues`

**RoxygenNote** 6.0.1

**LinkingTo** Rcpp

**Roxygen** list(markdown = TRUE)

**NeedsCompilation** yes

**Author** Tom Wilson [aut, cre],
Jasen Finch [aut]

**Maintainer** Tom Wilson <`tpw2@aber.ac.uk`>

## R topics documented:

## forestControl-package *False Positive Rate Control in Selection Frequency for Random Forest*

### Description

This package is an implementation of the methods described by Ender Konukoglu and Melanie Ganz in *Konukoglu, E. and Ganz, M., 2014. Approximate false positive rate control in selection frequency for random forest. arXiv preprint arXiv:1410.2838* https://arxiv.org/abs/1410.2838.

## extract_params *Extract forest parameters*

### Description

For a randomForest or ranger classification object, extract the parameters needed to calculate an approximate selection frequency threshold

### Usage

```
extract_params(x)
```

### Arguments

| | |
|---|---|
| x | a randomForest or ranger object |

### Value

a list of four elements

- **Fn** The number of features considered at each internal node (mtry)
- **Ft** The total number of features in the data set
- **K** The average number of binary tests/internal nodes across the enitre forest
- **Tr** The total number of trees in the forest

### Author(s)

Tom Wilson <tpw2@aber.ac.uk>

## Examples

```
library(randomForest)
data(iris)
iris.rf <- randomForest(iris[,-5], iris[,5], forest = TRUE)

iris.params <- extract_params(iris.rf)
print(iris.params)
```

---

# `fpr_fs` *False Postivie Rate Feature Selection*

---

## Description

Calculate the False Positive Rate (FPR) for each feature using it's selection frequency

## Usage

```
fpr_fs(x)
```

## Arguments

| | |
|---|---|
| `x` | a `randomForest` or `ranger` object |

## Value

a `tibble` of selection frequencies and their false positive rate

## Author(s)

Jasen Finch <`jsf9@aber.ac.uk`>

## Examples

```
library(randomForest)
data(iris)
iris.rf <- randomForest(iris[,-5], iris[,5], forest = TRUE)

iris.features <- fpr_fs(iris.rf)
print(iris.features)
```

## selection_freqs *Variable Selection Frequencies*

### Description

Extract variable selection frequencies from randomForest and ranger model objects

### Usage

```
selection_freqs(x)
```

### Arguments

| | |
|---|---|
| x | a randomForest or ranger object |

### Value

tibble of variable selection frequencies

### Examples

```
library(randomForest)
data(iris)
iris.rf <- randomForest(iris[,-5], iris[,5], forest = TRUE)

iris.freqs <- selection_freqs(iris.rf)
print(iris.freqs)
```

## sft *Selection Frequency Threshold*

### Description

Determine the selecton frequency threshold of a model at a specified false positive rate

### Usage

```
sft(x, alpha)
```

### Arguments

| | |
|---|---|
| x | a randomForest or ranger object |
| alpha | a false positive rate (ie, 0.01) |

## Value

a list of two elements

- **sft** Tthe selection frequency threshold
- **probs_atsft** The esimated false positive rate

## Author(s)

Tom Wilson <tpw2@aber.ac.uk>

## Examples

```
library(randomForest)
data(iris)
iris.rf <- randomForest(iris[,-5], iris[,5], forest = TRUE)

# For a false positive rate of 1%
iris.sft <- sft(iris.rf, 0.01)
print(iris.sft)

# To iterate through a range of alpha values

alpha <- c(0.01,0.05, 0.1,0.15,0.2, 0.25)
threshold <- NULL
for(i in seq_along(alpha)){
    threshold[i] <- sft(iris.rf, alpha[i])$sft
}

plot(alpha, threshold, type = 'b')
```

# Index