Package 'varppRule'

August 20, 2021

Type Package

Title Variant Prioritisation and Predictive Rule Modelling for rare and other genetic disorders

Version 0.1.0

Maintainer Sebastian Rauschert <Sebastian.Rauschert@telethonkids.org.au>

Description This package is an extension of the VARPP (https://github.com/deniando/VARPP) model. Whole genome and exome sequencing are now standard tools in the diagnostic process of
patients suffering from rare and other genetic disorders. The bottleneck for succesfull diagnosis is finding the disease causing variants amongst tens of thousands of genetic variants returned by such tests. One step in this process is to pre-filter the variants based on known benign/non-disease causing variants.
This package, similar to the original VARPP code, focuses on the task of prioritising variants in respect to the observed disease phenotype(s), after applying the pre-filtering step.
This package links gene expression across multiple tissues and cell types to the phenotypes, hence the name (VAR)iant (P)rioritisation by (P)henotype. It can prioritise potential disease causing
variants in a personalised manner.
On top of the original task of prioritising variants, this version 2 of VARPP also returns a set of Rules that led to the prioritisation of the variants. This is based on the work by Fiedman and Popescu (Friedman JH, Popescu BE. Predictive learning via rule ensembles. The Annals of Applied Statistics. 2008;2(3):916-54.).

License GPL-3 + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports ranger,
glmnet,
doMC,
doParallel,
parallel,
foreach,
tidyverse,
precrec,
data.table,
caret,
tidypredict,
progress,

stringr,
dplyr,
magrittr,
plyr,
iterators,
lattice,
grid,
ggplot2,
precrec,
rmarkdown,
knitr,
plotly,
DT,
pander,
lubridate,
pROC,
tidyr,
tidyselect (>= 1.1.0),
rclipboard,
shiny

RoxygenNote 7.1.1

R topics documented:

.class_by_threshold *Return model metrics for RuleFit*

Description

Return model metrics for RuleFit

Usage

```
.class_by_threshold(actual, predicted)
```

Arguments

actual	values and predicted values of the model

.extract_ranger_rules *Extract rules from ranger trees*

Description

This function returns rules based on the decision trees built in ranger. It depends on the function varpp and is only meant to be executed internally in rule_fit()

Usage

```
.extract_ranger_rules(rf_results)
```

Arguments

rf_results	the results fro mthe ranger tree generation within the varpp function

Value

a named vector of rules

.sample_benign_variants *Sampling and sub-setting for the benign variant data*

Description

This is an internal function, not to be executed outside of the varpp() and rule_fit() functions

Usage

```
.sample_benign_variants(benign_data, sampled_genes)
```

Arguments

benign_data	the subset of benign variants
sampled_genes	the sampled genes (with replacement) from the sampling step

Value

a list of randomly sampled gene variants, without replacement

---

.threshold *Return model metrics for RuleFit*

Description

Return model metrics for RuleFit

Usage

```
.threshold(actual, predicted)
```

Arguments

actual	values and predicted values of the model

---

auPRC *Area under the precision recall curve plot for Class* varppRule

---

Description

Area under the precision recall curve plot for Class varppRule

Usage

```
auPRC(x)
```

Arguments

x	an object of class varppRule

---

auROC *Area under the receiver operator curve plot for Class* varppRule

---

Description

Area under the receiver operator curve plot for Class varppRule

Usage

```
auROC(x)
```

Arguments

x	an object of class varppRule

density_plot *Density plot for the rule predictions*

Description

Density plot for the rule predictions

Usage

```
density_plot(rulefit_results)
```

Arguments

rulefit_results	an object of class varppRule

Value

a density plot for the predictions based on the final rules

genePanelTop *Return a gene panel based on the tissues in the top rule*

Description

Return a gene panel based on the tissues in the top rule

Usage

```
genePanelTop(varppRuleObject)
```

Arguments

varppRuleObject	the results from varppRule

getCADDcutOff *Function to extract CADD score including cut-off*

Description

This is an internal function, not to be used by itself.

Usage

```
getCADDcutOff(varppRuleObject)
```

Arguments

varppRuleObject	the results from varppRule

kappa_stats *Calculate kappa statistic*

Description

Function to calculate kappa statistic; only meant to be used internal to the rule_fit() function.

Usage

```
kappa_stats(cross_table)
```

Arguments

cross_table	the confusion Matrix of predictions and actual data

lasso_ensemble *LASSO cross validation of rules*

Description

This function performs nested cross validation on the generated rule data set. It is the final step in the rule_fit algorithm and returns the predictions. This function is an internal function and is not meant to be executed on its own.

Usage

```
lasso_ensemble(data, rules, bootstrap.rounds, cores)
```

Arguments

data	is a list of data with the rules added. The benign and the pathogenic variants files are necessary for the sampling.
rules	is the list of rules that were generated in the varpp function. This is necessary for the annotation of the final results.
bootstrap.rounds	number of bootstrap rounds for the outer loop of the LASSO cross-validation, defaults to 100.
cores	number of cores for parallel, defaults to 4

Value

A list of predictions for the CADD raw rank score and the tissue/cell specific expression added. Further, a variable importance list for all rules and variables tested.

metrics *Return model metrics for RuleFit*

Description

Return model metrics for RuleFit

Usage

```
metrics(x)
```

Arguments

x	an object of class varppRule

performance *Return a table with model names, auPRC, PP100 and ntree of the model: only for two level bootstrap model*

Description

Return a table with model names, auPRC, PP100 and ntree of the model: only for two level bootstrap model

Usage

```
performance(x, ntree = x$ntree)
```

Arguments

x	an object of class varppRule

performance_varpp *Return a table with model names, auPRC, PP100 and ntree of the model: only for two level bootstrap model*

Description

Return a table with model names, auPRC, PP100 and ntree of the model: only for two level bootstrap model

Usage

```
performance_varpp(x)
```

Arguments

x	an object of class varpp

predict.varppRule *Predict function for varppRule*

Description

This function requires extra data and can not be executed by itself. We need to download the genome_files, prepare a patient data file from the .vcf file and can then apply this function.

Usage

```
## S3 method for class 'varppRule'
predict(patient_data, model_results, predict = c("probability", "class"))
```

Arguments

patient_data	based on a patient .vcf file, a preprocessed input file that is annotated with GTEx and CADD scores
hpo_term	patient hpo terms

print.varpp *Class specific functions*

Description

Class specific functions

Usage

```
## S3 method for class 'varpp'
print(x)
```

Arguments

x,	an object of class varpp

print.varppRule *Class specific functions*

Description

Class specific functions

Usage

```
## S3 method for class 'varppRule'
print(x)
```

Arguments

x,	an object of class varppRule

`removeCADD` *Function to remove the CADD score variable including '>' , '<' and '=' from the rules*

Description

This is an internal function, not to be used by itself.

Usage

```
removeCADD(varppRuleObject)
```

Arguments

`varppRuleObject`	the results from varppRule

`ruleVariantPlot` *Scatterplot of # of rules that predict correctly versus # of variants per gene*

Description

Scatterplot of # of rules that predict correctly versus # of variants per gene

Usage

```
ruleVariantPlot(rulefit_results)
```

Arguments

`rulefit_results`	an object of class varppRule
`y`	the outcome variable

Value

a scatterplot of # of rules and # of variants per gene

ruleVarImp *Function to return all rules ranked by variable importance*

Description

Function to return all rules ranked by variable importance

Usage

```
ruleVarImp(x)
```

Arguments

x	an object of class varppRule

rule_fit *The RuleFit function*

Description

RuleFit creates variant predictions and human interpretable rules

Usage

```
rule_fit(
  HPO_genes,
  HPO_term_name = "custom",
  type = c("gtex", "hcl", "custom"),
  user_patho = NULL,
  user_benign = NULL,
  ntree = 200,
  max.depth = 3,
  rule.filter = 10,
  bootstrap.rounds = 100,
  rule.extract.cores = 4,
  kappa.cores = 2,
  lasso.cores = 4
)
```

Arguments

HPO_genes	HPO term associated list of genes, or any list of patient genes.
HPO_term_name	In case the model is for one specific HPO term, this can be provided, otherwise it is assigned as "custom"
type	the prediction data; either hcl (single cell), gtex (tissue specific) or custom (requires the user to provide custom_patho and custom_benign).
user_patho	a user provided file for the pathogenic variants. This needs to have the following first few columns:Gene, GeneVariant, CADD_raw_rankscore, CADD_PHRED_SCORE, Pathogenic, gene_id,gene_biotype

`user_benign` a user provided file for the benign variants. This needs to have the following first few columns:Gene, GeneVariant, CADD_raw_rankscore, CADD_PHRED_SCORE, Pathogenic, gene_id,gene_biotype

`ntree` number of trees to be built, defaults to 200.

`max.depth` maximum tree depth, defaults to 3.

`rule.filter` filter the top n rules based on kappa statistic. If NULL, the rules are filter above a kappa of 0.05.

`bootstrap.rounds` number of bootstrap rounds for the outer loop of the LASSO cross-validation, defaults to 100.

`rule.extract.cores` number of cores for parallel, defaults to 4. This is specifically for the varpp rule extract step (less memory hungry than the cv.glmnet step).

`kappa.cores` number of cores used for the rule filtering by kappa. This needs to be separate, as it is quite memory intensive when the input + rule data is very large. Defaults to 2.

`lasso.cores` number of cores for the cv.glmnet step, as this is quite memory hungry, it is separated.

Value

A list of predictions for the outcome. Further, a variable importance list for all rules and variables tested.

`selected_rule_performance`

Prediction of single rules

Description

Prediction of single rules

Usage

```
selected_rule_performance(rulename, rulefit_results_object)
```

Arguments

`rulename` is the name of one of the rules as returned by the varppRule model

`rulefit_results_object,` a varppRule object

varIMP *varIMP: Function to extract the variable importance of the expression data variables*

Description

This function is provided on top of ruleVarImp. It Re-weights the rule kappas by the variables selected per rule and returns a 0 to 1 scaled importance value per tissue. The most important variable will have a value of 1. This is based on the variable importance described in the RuleFit publication by Friedman and Popescue

Usage

```
varIMP(rule_model = NULL, HPOterm)
```

Arguments

rule_model	This is the RuleFit model object
HPOterm	Add the HPO term name for the model

varpp *Based on the original VARPP paper, this algorithm is the parallelised and updated version of the model*

Description

Based on the original VARPP paper, this algorithm is the parallelised and updated version of the model

Usage

```
varpp(
  HPO_genes,
  type = c("gtex", "hcl", "custom"),
  user_patho = NULL,
  user_benign = NULL,
  ntree = 500,
  max.depth = NULL,
  cores = 4
)
```

Arguments

HPO_genes	HPO term associated list of genes
type	the prediction data; either hcl (single cell), gtex (tissue specific) or custom (requires the user to provide custom_patho and custom_benign).
user_patho	a user provided file for the pathogenic variants. This needs to have the following first few columns:Gene, GeneVariant, CADD_raw_rankscore, CADD_PHRED_SCORE, Pathogenic, gene_id,gene_biotype

user_benign	a user provided file for the benign variants. This needs to have the following first few columns:Gene, GeneVariant, CADD_raw_rankscore, CADD_PHRED_SCORE, Pathogenic, gene_id,gene_biotype
ntree	is the number of trees that should be built for ranger. It defaults to 1000
max.depth	is the maximum tree depth for the ranger trees. IT defaults to 3.
cores	number of cores for parallel, defaults to 4

`varpp_for_rulefit` *varpp: extract rules from ranger trees*

Description

This function is meant to only be used internally for the rule_fit function

Usage

```
varpp_for_rulefit(dat, ntree, max.depth, cores)
```

Arguments

dat	this is a data list returned from the function load_gtex_or_hcl. It is either GTEx tissue specific gene expression or HCL cell specific expression
ntree	is the number of trees that should be built for ranger. It defaults to 1000
max.depth	is the maximum tree depth for the ranger trees. IT defaults to 3.
cores	number of cores for parallel, defaults to 4

`varpp_report` *Report function for rule_fit() results*

Description

This function will create a report based on the results from the rulef_fit() function.'

Usage

```
varpp_report(results, report_filename)
```

Arguments

results	the results from the rule_fit() function
report_filename	The path, including the filename for the resulting report.

Index