Supplementary File 1: Functional Elements of the FORESEE Pipeline

L. K. Turnhoff, A. H. Esfahani, M. Montazeri, N. Kusch and A. Schuppert

February 2019

The functional elements of the modeling pipeline are implemented as independent, individually changeable modules. This Supplementary File aims to explain in more detail the elements that FORESEE offers and demonstrates which functional arguments can be chosen to select from the pre-implemented methods.

After choosing the format for model output values by using the function argument *CellResponseType*, the module CellResponseProcessor [1, 2, 3] uses the function argument *CellResponseTransformation* to distinguish between methods for output transformation, such as power transformation or binarization. The function argument *InputDataTypes* is utilized to specify the molecular data type used for the model inputs. Before the module FeatureCombiner summarizes the chosen features, FORESEE employs multiple functional blocks to prepare the training set with respect to the subset of samples and features to be utilized. The function argument *TrainingTissue* of the module SampleSelector selects the samples according to their tissue of origin, while the function argument *GeneFilter* in the FeatureSelector restricts the features that are used to either predefined gene sets or a subset based on variance measures in the training set. Moreover, the module DuplicationHandler uses the function argument *DuplicationHandling* to distinguish between methods for dealing with duplicated gene names in the FORESEE objects. If the model includes gene expression data, the function argument *HomogenizationMethod* of the module Homogenizer [4, 5, 6, 7, 8] determines how batch effects between the training object ForeseeCell and the testing object ForeseePatient are to be removed, whereas the function argument *FeaturePreprocessing* of the module FeaturePreprocesser transforms the input values using for example principal component analysis or PhysioSpace [9]. Finally, the module BlackBoxFilter [10, 11, 12, 13, 14, 15, 16] trains the model applying a regression algorithm that is specified by the function argument *BlackBox*. In the course of this, the function argument *nfoldCrossvalidation* enables the user to choose whether all samples of the training object are used at once or whether the training process is executed doing a n-fold cross-validation, extracting the best performing model to consequently apply to the independent testing object. For model validation, the function argument *Evaluation* offers various methods that measure model performance with the module Validator [17, 18], where the predicted Foreseen values are compared with the actual annotations of the ForeseePatient object.

Across all main steps of the pipeline, user-defined functions can substitute the pre-implemented methods to enable a more flexible use of the package. An overview of all functional elements and their respective arguments is depicted in table 1.

Module	**Function Argument**	**Options**
CellResponseProcessor	CellResponseTransformation	binarization_cutoff [2] binarization_kmeans [1] logarithm none powertransform [3] user-defined function
FeatureSelector	GeneFilter	all landmarkgenes ontology pathway pvalue variance user-defined function
DuplicationHandler	DuplicationHandling	first mean none user-defined function
Homogenizer	HomogenizationMethod	ComBat [4] limma [6] none quantile [5] RUV RUV4 [8] YuGene [7] user-defined function
FeaturePreprocesser	FeaturePreprocessing	none pca physio [9] zscore_samplewise zscore_genewise user-defined function
BlackBoxFilter	BlackBox	elasticnet [11, 12] lasso [11] linear rf [14] rf_ranger [15] ridge [10] svm [13] tandem [11, 16] user-defined function
Validator	Evaluation	fpvalue mse pearson prauc [18] rocauc [17] rocpvalue [17] rsquared rsquared_adjusted spearman user-defined function

Table 1: Input options for different modules of the FORESEE pipeline.

References

[1] S. Mundus et al. *Binarize: Binarization of one-dimensional data*, 2017. R package version 1.2.

[2] S. Epskamp et al. Estimating psychological networks and their accuracy: A tutorial paper. *Behav Res Methods*, 50:195–212, 2018.

[3] J. Fox and S. Weisberg. *An R companion to applied regression.* Sage, Thousand Oaks CA, second edition, 2011.

[4] J. T. Leek et al. *sva: Surrogate variable analysis*, 2017. R package version 3.26.0.

[5] B. Bolstad. *preprocessCore: A collection of pre-processing functions*, 2017. R package version 1.40.0.

[6] M. E. Ritchie et al. limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic Acids Res*, 43:e47, 2015.

[7] K.-A. Lê Cao et al. Yugene: A simple approach to scale gene expression data derived from different platforms for integrated analyses. *Genomics*, 103:239–251, 2014.

[8] J. Gagnon-Bartsch. *ruv: Detect and Remove Unwanted Variation using Negative Controls*, 2018. R package version 0.9.7.

[9] M. Lenz et al. Physiospace: Relating gene expression experiments from heterogeneous sources using shared physiological processes. *PLoS One*, 8:e77627, 2013.

[10] S. Moritz and E. Cule. *ridge: Ridge regression with automatic selection of the penalty parameter*, 2017. R package version 2.2.

[11] J. Friedman et al. Regularization paths for generalized linear models via coordinate descent. *J Stat Softw*, 33:1–22, 2010.

[12] Microsoft and H. Ooi. *glmnetUtils: Utilities for 'Glmnet'*, 2017. R package version 1.1.

[13] D. Meyer et al. *e1071: Misc functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2017. R package version 1.6-8.

[14] A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2:18–22, 2002.

[15] M. N. Wright and A. Ziegler. ranger: A fast implementation of random forests for high dimensional data in C++ and R. *J Stat Softw*, 77:1–17, 2017.

[16] N. Aben. *TANDEM: A two-stage approach to maximize interpretability of drug response models based on multiple molecular data types.*, 2017. R package version 1.0.2.

[17] X. Robin et al. proc: An open-source package for R and S+ to analyze and compare ROC curves. *BMC Bioinformatics*, 12, 2011.

[18] J. Grau et al. Prroc: Computing and visualizing precision-recall and receiver operating characteristic curves in R. *Bioinformatics*, 31:2595–2597, 2015.