Why do we need NADIA?

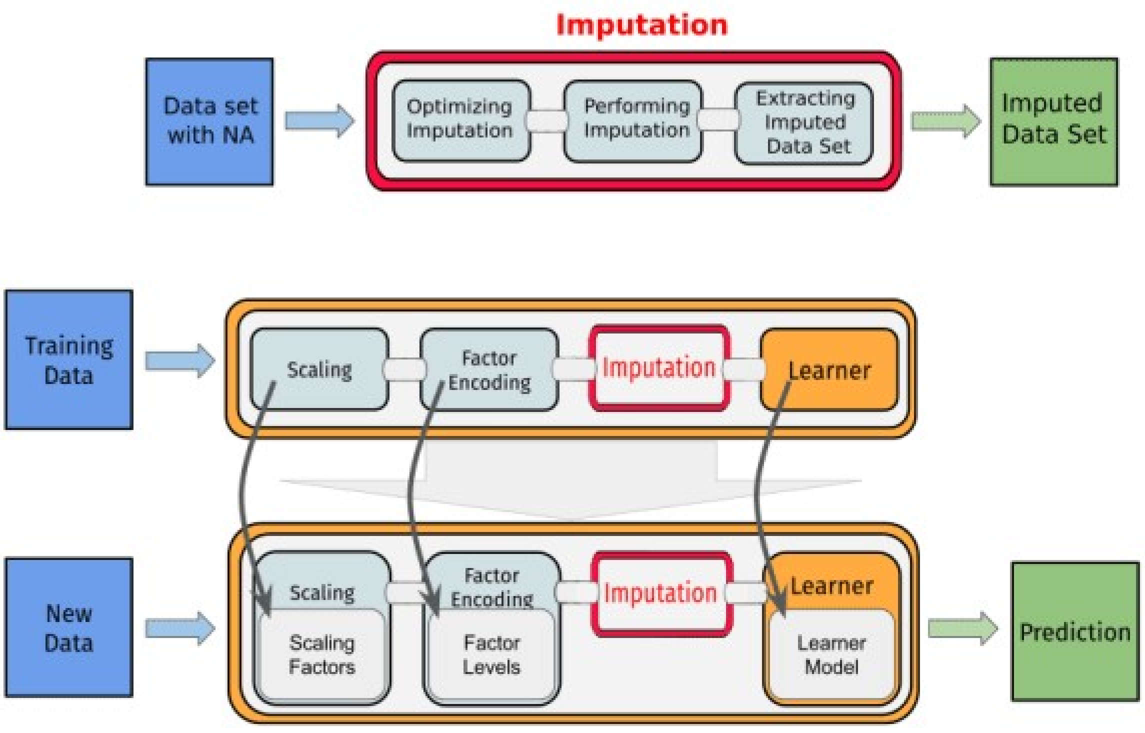
Machine learning is an important part of working in R. Packages like mlr3 simplify the whole process. Its no need to manually split data into training and test set, no need to manually fit linear models. Even more packages like mlr3pipelines let you crate complex pipelines and include feature engineering and transformation in your models. R is also used by statisticians, from statisticians we have advanced methods of imputing missing data like mice or Amelia. What happens when we want to connect machine learning with a statistical approach. This is where **NADIA** comes.

NADIA is available on CRAN: install.packages("NADIA")

How does it work?

At this point, maybe some of you wonder what **NADIA** is really doing? So **NADIA (NA Data Imputation Algorithms)** creates a unified interface for popular imputation packages using mlr3pipelines. If you want to use mice, missMDA, Amelia, or another popular package in your machine learning models you can do much easier **NADIA**.

Statistical approach standing behind all of the included packages generates problems when we want to use them in ML. Neither of them creates models. In a typical machine learning case, imputing missing data is treated as part of the model and should be trained on training data. This is not possible with included packages, to solve this problem we propose something a little different:



Approach to data imputation used in NADIA. Graphic inspire by mlr3book

We decided to exclude imputation from the normal ML workflow. In this case, imputation is basically trained and used separately for training and test sets. This allows to include any method of imputing missing data in **NADIA**. For some packages (for example, mice) the standard approach is also available. We are currently working to suit more packages to work in a normal approach.

Case study

The true power of the presented package comes from, how it simplifies the process of including advance imputation techniques in ML models. To show you what I mean I will present how to perform a simple tas with and without **NADIA**.

The task will be to evaluate imputation methods in ML using two folds cross-validation and compare results.

First, we must make certain assumptions:

- We will use a dataset called *Pima* containing Indians Diabetes data from mlr3
- I will try to maximally shortcode but include every necessary step
- As a result, we will compare the accuracy achieved by two packages
- I will not care to assure the same rows in test and training set between the two approaches (they will be the same for two packages in one approach)

With NADIA

```
# Creating Graph
graph_mice <-
PipeOpMice$new() %>>% lrn('classif.glmnet')
graph_missMDA <-
PipeOpMissMDA_PCA_MCA_FMAD$new() %>>% lrn('classif.glmnet')
# Creating Graph Learners
Learner_mice <-  GraphLearner$new(graph_mice)
Learner_missMDA <- GraphLearner$new(graph_missMDA)
# Cross-Validation
set.seed(1)
rr_mice <-
resample(tsk('pima'),Learner_mice,rsmp('cv',folds=2))
set.seed(1)
rr_missMDA <- resample(tsk('pima'),Learner_missMDA,rsmp('cv',folds=2))
# Compering accuracy
rr_mice$aggregate(msr('classif.acc'))
rr_missMDA$aggregate(msr('classif.acc'))
```

This task is simpler with **NADIA** because you have access to already written methods from mlr3.

Code, where I am doing it manually is long, so I included only a part of it. Without using **NADIA** this task involves a lot of operations, it’s more complicated and requires knowledge about used packages.

First, we need to extract data from the task.

```
data <-
  as.data.frame(tsk('pima')$data(cols = tsk('pima')$feature_names))
target <-
  as.data.frame(tsk('pima')$data(cols = tsk('pima')$target_names))
```

Before creating models I need to manually impute missing data.

```
# Creating split
ind <- sample(1:nrow(data),floor(nrow(data)/2))
# I have to impute data in 4 data frames
data1_mice <- complete(mice(data[ind,]))
data2_mice <- complete(mice(data[-ind,]))
# In the case of missMDA, I have to include the
# preparation step
ncp_1 <- estim_ncpPCA(data[ind,])
ncp_2 <- estim_ncpPCA(data[-ind,])
data1_MDA <- as.data.frame(imputePCA(data[ind,],ncp_1$ncp)$completeObs)
data2_MDA <-
  as.data.frame(imputePCA(data[-ind,],ncp_2$ncp)$completeObs)
# Adding back column with target
data1_mice$diabetes <- target[ind,]
data2_mice$diabetes <- target[-ind,]
data1_MDA$diabetes <- target[ind,]
data2_MDA$diabetes <- target[-ind,]
# I have to create all tasks manually
task1_mice <- TaskClassif$new('mice1',data1_mice,'diabetes')
task2_mice <- TaskClassif$new('mice2',data2_mice,'diabetes')
task1_MDA <- TaskClassif$new('MDA1',data1_MDA,'diabetes')
task2_MDA <- TaskClassif$new('MDA2',data1_MDA,'diabetes')
```

To finish the experiment, I would need to manually train and evaluate one model per fold (four models). I show how to do it on only one model.

```
# Crating Learner
Learner <- lrn('classif.glmnet')
# Fold 1 mice
Learner$train(task1_mice)
acc1 <- Learner$predict(task2_mice)$score(msrs('classif.acc'))
```

As you can see it's faster using **NADIA** but being faster and easier is not the only advantage of our package.

Firstly, you don't need to know about additional steps to use the full potential of the chosen methods. In this example, this step is pretty simple, but this is not always the case. In our package preparation steps are automated and can be done without the attention of the users. The presented package also implements any forms of improving imputation if such is included in the imputation package. For example using mice we can choose an imputation matrix (matrix indicating which variables will be imputed using which). In **NADIA**, the process of finding the best possible matrix is automatized.

Secondly, **NADIA** allows you to handle errors using functionality already implemented in mlr3. Error handling is important when you want to use advanced statistical methods on real data.

Conclusions

NADIA is a useful extension to mlr3pipelines and simplifies work with imputation packages especially in machine learning. Our package is also simple to use and if you already know mlr3 you shouldn’t have an
problem working with **NADIA**.