Sick dataset analysis 2

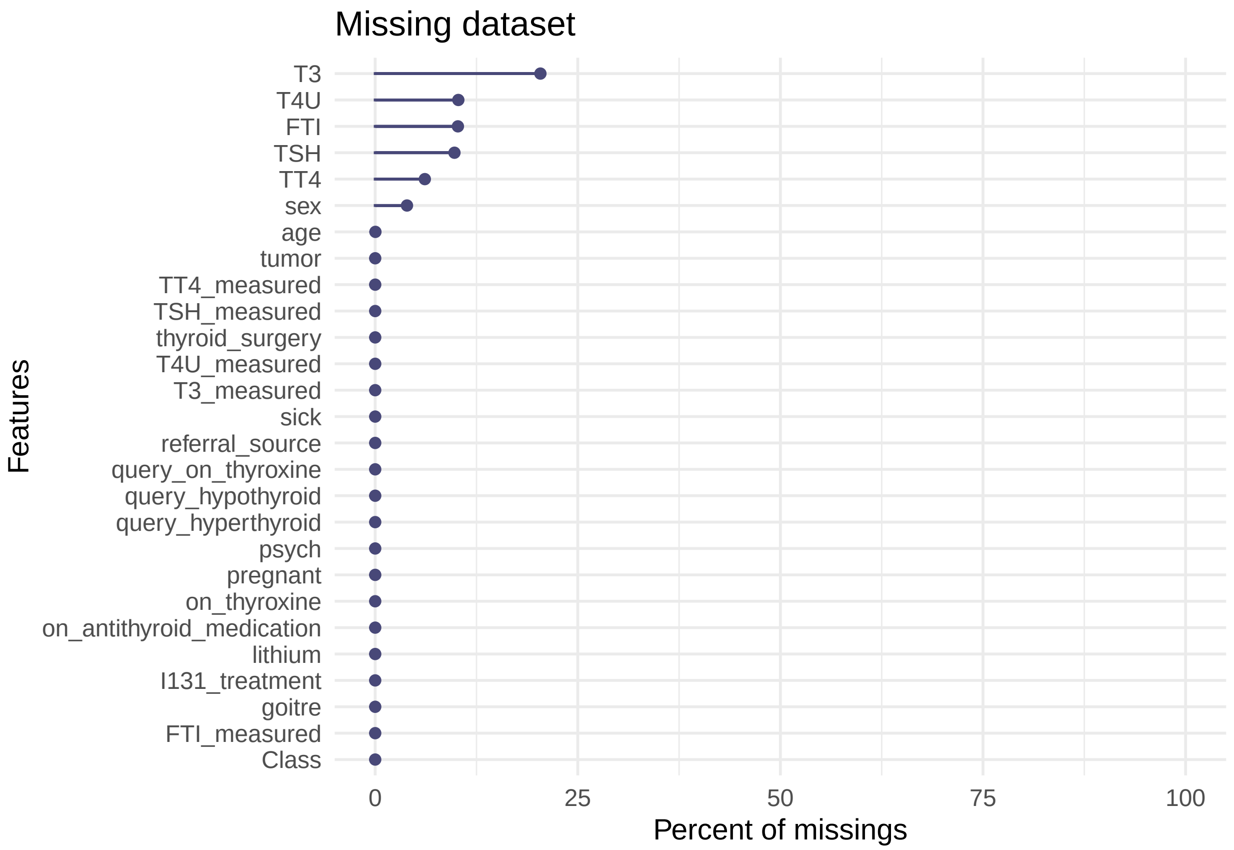
Malgorzata Wachulec

29 04 2020

Preprocessing

Some factor columns do not have sufficient number of observations in each of the levels. That means some sets will have observations with only one feature value. These variables will not be useful when training the model and might cause errors. For this reason I am deleting columns 'TBG', 'hypopituitary' and 'TBG_measured'. I have also decided to set limits to hormones, but this will be done after dividing the data into train, validation and test sets.

After these columns are deleted, we can now look at the missing values. Since in every column there is less than 10% of missing data, I will imput it as opposed to deleting the entire column or observation for which the column is empty.



As I am planning use the mice package to imput missing data, which is predicting the missing values through model built on other observations, I will first divide data into training, validation and test sets. This will prevent for training set to have values inputed based on observations from test set and vice versa.

```r
# dividing data into train and test
training_indicies <- read.table("indeksy_treningowe.txt")
training_indicies <- training_indicies$x
trainset_to_be_divided <- dataset[ training_indicies,]
testset <-  dataset[-training_indicies,]

# dividing training set into 75% training set and 25% valdation set
# each set has the same distribution of the target class
set.seed(3456)
trainIndex <- createDataPartition(trainset_to_be_divided$Class, p = .75,
                                  list = FALSE,
                                  times = 1)
trainset <- trainset_to_be_divided[ trainIndex,]
validset <- trainset_to_be_divided[-trainIndex,]

# Limiting hormone levels
preprocess <- function(dataset) {
  dataset <- data.table(dataset)
  dataset <- dataset[TSH < 100 & T3 < 30 & TT4 < 400 & FTI < 400]
  dataset <- data.frame(dataset)
  ## For other models
  #dataset <- as.data.frame(one_hot(as.data.table(dataset), cols = "referral_source"))
  #for(i in 1:(ncol(dataset))) {
  #  if(is.factor(dataset[, i])) {
  #    dataset[, i] = as.numeric(dataset[, i])
  #  }
  #}
  dataset
}

trainset <- preprocess(trainset)
validset <- preprocess(validset)
testset <- preprocess(testset)
```

Imputation of missing values using mice() function from mice package. As to make sure, that the model for imputing missing values will not use target variable, it is taken away before imputing and added again to the training, validation and test set after the imputing process.

Building black box model

Here I am defining training, validation and test tasks, as well as the learner. Then I am checking it's performance on the validation set.

```r
# task and learner definition
complete_trainset$Class <- as.factor(complete_trainset$Class)
complete_validset$Class <- as.factor(complete_validset$Class)
complete_testset$Class <- as.factor(complete_testset$Class)
pos <- 'sick' # '1' for other models
trainTask <- TaskClassif$new(id = "train sick", backend = complete_trainset,
                             target = "Class", positive = pos)
validTask <- TaskClassif$new(id = "valid sick", backend = complete_validset,
                             target = "Class", positive = pos)
testTask <- TaskClassif$new(id = "test sick", backend = complete_testset,
                              target = "Class",positive = pos)

learner <- lrn("classif.ranger")
## Other models
# learner <- lrn("classif.xgboost")
# learner <- lrn("classif.svm")

learner$predict_type <- "prob"

# model and prediction
learner$train(trainTask)
result <- learner$predict(validTask)
cat("Contigency table: \n")
```

```
## Contigency table:
```

```
result$confusion
```

```
##           truth
## response   sick negative
##   sick       33        8
##   negative    8      492
```

Let's check recall and other measures of this prediction.

```
## Auc on validation set:  0.9931707
```

```
## Auprc on validation set:  0.9075903
```

```
## Recall on validation set:  0.804878
```

```
## Specificity on validation set:  0.984
```

Trying other models

I have also tried xgboost model from the mlr3 package that required changing factor variables to one-hot-encoding and the result obtained on the validation set was: auc measure equal to 0.93 and auprc equal to 0.85. Another model that I tried was SVM model from mlr3 package which gave even worse results on the validation set - only 0.74 of accuracy and very low auprc measure.

At the end I have again run ranger model on the model with the one-hot-encoding, and, to my surprise (I thought it would be at least as good as the one with regular values), it performed worse that the one shown in the previous section - results on validation set were: acc equal to 0.98 and auprc equal to 0.86. This is why the final model is ranger without one-hot-encoding which gives auprc equal to nearly 0.91 on validation set. Let's check how it will perform on the testset.

Final score on testset

These are the results obtained on the test set using the final version of the model.

```
## Contigency table:
```

```
##           truth
## response   sick negative
##   sick       34        8
##   negative   12      497
##
##
## Auc on test set:  0.9909169
## Auprc on test set:  0.8937523
## Recall on test set:  0.7391304
## Specificity on test set:  0.9841584
```