Testing COPE

Alexandros Rekkas

2021-02-25

Contents

1 Testing supporting functions

This set of tests evaluates the functions used to support the server function. These functions are used for calculating relevant quantities, e.g. 28-day mortality risk.

Function `createModelMatrix` creates the model matrix based on a set of covariates and a list of transformations.

```
transformationsMortality <- list(
    age             = identity,
    respiratoryRate = log,
    crp             = log,
    ldh             = log,
    albumin         = log,
    urea            = log
)


testthat::test_that(
    "Creation of model matrix works",
    {
        testthat::expect_equal(
            createModelMatrix(
                covariates      = rep(1, 6),
                transformations = transformationsMortality
            ),
            c(1, rep(0, 5))
        )
    }
)
```

```
## Test passed
```

Function `createLinearPredictor` calculates the linear predictor of a prediction based on a provided model matrix, a vector of β coefficients and an intercept

```
testthat::test_that(
    "Creation of linear predictor works",
    {
        testthat::expect_equal(
            createLinearPredictor(
                modelMatrix = rep(1, 6),
                beta        = rep(1, 6),
                intercept   = 1
            ),
            matrix(7)
        )
    }
)
```

```
## Test passed
```

Function `logisticProbability` calculates the logistic probability (%) based on a provided linear predictor value.

```
testthat::test_that(
    "Logistic probability calculation from linear predictor works",
    {
        testthat::expect_equal(
            logisticProbability(0),
            50
        )
    }
)
```

```
## Test passed
```

Finally, function `extractQuantiles` extracts the required quantiles for the calibration plot, based on the stored dataframe of `calibrationQuantiles`, the outcome of interest and the hospital under consideration.

```
testthat::test_that(
    "Extraction of calibration quantiles works",
    {
        testthat::expect_equal(
            extractQuantiles(
                outcome              = 1,
                center               = 1,
                calibrationQuantiles = data.frame(
                    center  = 1,
                    outcome = 1,
                    quant20 = 20,
                    quant40 = 40,
                    quant60 = 60,
```

```
                    quant80 = 80
                )
            ),
            c(
                quant20 = 20, quant40 = 40,
                quant60 = 60, quant80 = 80
            )
        )
    }
)
```

```
## Test passed
```

2 Testing server functions

Here we perform a set of unit tests to ensure that server-side operations work the way they should. We start by looking at calculations tasks run within the server function.

2.1 Interactivity

Interactive values, are server-side variables that depend on the input and, therefore, need to be updated whenever the user alters their selection.

First, the `currentInputData` reactive dataframe should contain all the provided inputs and should update accordingly, when these are changed.

```
shiny::testServer(
    expr = {
        session$setInputs(
            age                       = 70,
            respiratoryRate           = 19,
            ldh                       = 244,
            crp                       = 48,
            albumin                   = 39,
            urea                      = 6.5,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "The reactive input dataframe is correct",
            {
                testthat::expect_equal(
                    currentInputData(),
                    data.frame(
                        age             = 70,
                        respiratoryRate = 19,
                        crp             = 48,
                        ldh             = 244,
                        albumin         = 39,
                        urea            = 6.5
                    )
```

```
                )
            }
        )

        session$setInputs(
            age                         = 70,
            respiratoryRate             = 19,
            ldh                         = 2440,
            crp                         = 48,
            albumin                     = 39,
            urea                        = 6.5,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "The reactive input dataframe is updated",
            {
                testthat::expect_equal(
                    currentInputData(),
                    data.frame(
                        age             = 70,
                        respiratoryRate = 19,
                        crp             = 48,
                        ldh             = 2440,
                        albumin         = 39,
                        urea            = 6.5
                    )
                )
            }
        )

    }
)
```

```
## Loading required package: shiny

##
## Attaching package: 'shinyalert'

## The following object is masked from 'package:shinyBS':
##
##     closeAlert

## The following object is masked from 'package:shiny':
##
##     runExample

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## Warning: `arrange_()` was deprecated in dplyr 0.7.0.
## Please use `arrange()` instead.
## See vignette('programming') for more help
```

```
## Test passed
## Test passed
```

Next, we do the same for the variables that keep track of the current prediction and its placement, relevant to the overall predicted risk fifths, for both outcomes (mortality and ICU admission).The first set of tests evaluates if the initial input is handled appropriately and the second set evaluates if the values are updated correctly.

```
shiny::testServer(
    expr = {
        session$setInputs(
            age                        = 70,
            respiratoryRate            = 19,
            ldh                        = 244,
            crp                        = 48,
            albumin                    = 39,
            urea                       = 6.5,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "Is the prediction for the starting values correct?",
            testthat::expect_equal(
                currentPrediction(),
                list(
                    mortality = 4.8,
                    icu       = 13.3
                )
            )
        )

        testthat::test_that(
            "Is the predicted mortality risk assigned to the correct stratum of risk?",
            testthat::expect_equal(
                riskFifthMortality(),
                4
            )
        )

        # Is the predicted ICU risk assigned to the correct stratum of risk?
        testthat::test_that(
            "Is the predicted mortality risk assigned to the correct stratum of risk?",
            testthat::expect_equal(
                riskFifthIcu(),
                4
```

```
            )
        )


        session$setInputs(
            age                       = 60,
            respiratoryRate           = 30,
            ldh                       = 400,
            crp                       = 100,
            albumin                   = 50,
            urea                      = 10,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "Current prediction is updated",
            testthat::expect_equal(
                currentPrediction(),
                list(
                    mortality = 10.8,
                    icu       = 20.6
                )
            )
        )

        testthat::test_that(
            "Is the predicted mortality risk assigned to the correct stratum of risk?",
            testthat::expect_equal(
                riskFifthMortality(),
                5
            )
        )

        # Is the predicted ICU risk assigned to the correct stratum of risk?
        testthat::test_that(
            "Is the predicted mortality risk assigned to the correct stratum of risk?",
            testthat::expect_equal(
                riskFifthIcu(),
                5
            )
        )
    }
)
```

```
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
```

2.2 Handling input admissibility

There are constraints on the input a user can provide. More specifically, all inputs need to be numeric and should be between the following limits:

Predictor	Minimum	Maximum
Age	0	100
Respiratory rate	10	60
LDH	50	4000
CRP	1	500
Albumin	10	60
Urea	1	80

For each input we test ensure that input is accepted when the above rules are followed and that it is ruled as non-admissible if the submitted number is outside the boundaries or if it is a character string.

```
shiny::testServer(
    expr = {
        session$setInputs(
            age                       = -1,
            respiratoryRate           = 19,
            ldh                       = 244,
            crp                       = 48,
            albumin                   = 39,
            urea                      = 6.5,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "Non admissible age (lower)",
            testthat::expect_equal(
                admissibleInput(),
                FALSE
            )
        )

        # age above admissible input
        session$setInputs(
            age                       = 101,
            respiratoryRate           = 19,
            ldh                       = 244,
            crp                       = 48,
            albumin                   = 39,
            urea                      = 6.5,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "Non admissible age (higher)",
            testthat::expect_equal(
                admissibleInput(),
                FALSE
```

```
    )
)

# age as character input
session$setInputs(
    age                       = "seventy",
    respiratoryRate           = 19,
    ldh                       = 244,
    crp                       = 48,
    albumin                   = 39,
    urea                      = 6.5,
    calculatePredictionButton = "click"
)

testthat::test_that(
    "Non admissible age (character)",
    testthat::expect_equal(
        admissibleInput(),
        FALSE
    )
)

# respiratory rate below admissible input
session$setInputs(
    age                       = 70,
    respiratoryRate           = 9,
    ldh                       = 244,
    crp                       = 48,
    albumin                   = 39,
    urea                      = 6.5,
    calculatePredictionButton = "click"
)

testthat::test_that(
    "Non admissible respiratory rate (lower)",
    testthat::expect_equal(
        admissibleInput(),
        FALSE
    )
)

# respiratory rate above admissible input
session$setInputs(
    age                       = 70,
    respiratoryRate           = 61,
    ldh                       = 244,
    crp                       = 48,
    albumin                   = 39,
    urea                      = 6.5,
    calculatePredictionButton = "click"
)

testthat::test_that(
```

```
        "Non admissible respiratory rate (higher)",
        testthat::expect_equal(
            admissibleInput(),
            FALSE
        )
    )

    # respiratory rate as character input
    session$setInputs(
        age                       = 70,
        respiratoryRate           = "ten",
        ldh                       = 244,
        crp                       = 48,
        albumin                   = 39,
        urea                      = 6.5,
        calculatePredictionButton = "click"
    )

    testthat::test_that(
        "Non admissible respiratory rate (character)",
        testthat::expect_equal(
            admissibleInput(),
            FALSE
        )
    )

    # ldh above admissible input
    session$setInputs(
        age                       = 70,
        respiratoryRate           = 45,
        ldh                       = 4001,
        crp                       = 48,
        albumin                   = 39,
        urea                      = 6.5,
        calculatePredictionButton = "click"
    )

    testthat::test_that(
        "Non admissible LDH (higher)",
        testthat::expect_equal(
            admissibleInput(),
            FALSE
        )
    )

    # ldh below admissible input
    session$setInputs(
        age                       = 70,
        respiratoryRate           = 45,
        ldh                       = 49,
        crp                       = 48,
        albumin                   = 39,
        urea                      = 6.5,
```

```
        calculatePredictionButton = "click"
    )

    testthat::test_that(
        "Non admissible LDH (lower)",
        testthat::expect_equal(
            admissibleInput(),
            FALSE
        )
    )
    # ldh as character input
    session$setInputs(
        age                       = 70,
        respiratoryRate           = 45,
        ldh                       = "two hundred",
        crp                       = 48,
        albumin                   = 39,
        urea                      = 6.5,
        calculatePredictionButton = "click"
    )

    testthat::test_that(
        "Non admissible LDH (character)",
        testthat::expect_equal(
            admissibleInput(),
            FALSE
        )
    )

    # crp below admissible input
    session$setInputs(
        age                       = 70,
        respiratoryRate           = 45,
        ldh                       = 244,
        crp                       = 0,
        albumin                   = 39,
        urea                      = 6.5,
        calculatePredictionButton = "click"
    )

    testthat::test_that(
        "Non admissible CRP (lower)",
        testthat::expect_equal(
            admissibleInput(),
            FALSE
        )
    )

    # crp above admissible input
    session$setInputs(
        age                       = 70,
        respiratoryRate           = 45,
        ldh                       = 244,
```

```
        crp                         = 501,
        albumin                     = 39,
        urea                        = 6.5,
        calculatePredictionButton = "click"
    )

    testthat::test_that(
        "Non admissible CRP (higher)",
        testthat::expect_equal(
            admissibleInput(),
            FALSE
        )
    )

    # crp as character input
    session$setInputs(
        age                         = 70,
        respiratoryRate             = 45,
        ldh                         = 244,
        crp                         = "two hundred",
        albumin                     = 39,
        urea                        = 6.5,
        calculatePredictionButton = "click"
    )

    testthat::test_that(
        "Non admissible CRP (character)",
        testthat::expect_equal(
            admissibleInput(),
            FALSE
        )
    )

    # albumin below admissible input
    session$setInputs(
        age                         = 70,
        respiratoryRate             = 45,
        ldh                         = 244,
        crp                         = 48,
        albumin                     = 9,
        urea                        = 6.5,
        calculatePredictionButton = "click"
    )

    testthat::test_that(
        "Non admissible albumin (lower)",
        testthat::expect_equal(
            admissibleInput(),
            FALSE
        )
    )

    # albumin above admissible input
```

```
        session$setInputs(
            age                       = 70,
            respiratoryRate           = 45,
            ldh                       = 244,
            crp                       = 48,
            albumin                   = 61,
            urea                      = 6.5,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "Non admissible albumin (higher)",
            testthat::expect_equal(
                admissibleInput(),
                FALSE
            )
        )

        # albumin as character input
        session$setInputs(
            age                       = 70,
            respiratoryRate           = 45,
            ldh                       = 244,
            crp                       = 48,
            albumin                   = "twenty",
            urea                      = 6.5,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "Non admissible albumin (character)",
            testthat::expect_equal(
                admissibleInput(),
                FALSE
            )
        )

        # urea below admissible input
        session$setInputs(
            age                       = 70,
            respiratoryRate           = 45,
            ldh                       = 244,
            crp                       = 48,
            albumin                   = 20,
            urea                      = 0,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "Non admissible albumin (lower)",
            testthat::expect_equal(
                admissibleInput(),
                FALSE
```

```
            )
        )

        # urea above admissible input
        session$setInputs(
            age                       = 70,
            respiratoryRate           = 45,
            ldh                       = 244,
            crp                       = 48,
            albumin                   = 20,
            urea                      = 81,
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "Non admissible albumin (higher)",
            testthat::expect_equal(
                admissibleInput(),
                FALSE
            )
        )

        # urea as character input
        session$setInputs(
            age                       = 70,
            respiratoryRate           = 45,
            ldh                       = 244,
            crp                       = 48,
            albumin                   = 20,
            urea                      = "four",
            calculatePredictionButton = "click"
        )

        testthat::test_that(
            "Non admissible albumin (character)",
            testthat::expect_equal(
                admissibleInput(),
                FALSE
            )
        )
    }
)
```

```
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
```

```
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
## Test passed
```