# Functional Machine Learning Workflows in R with funcml

**A methodological note on explicit, auditable machine learning workflows for health data science.**

Imad EL BADISY

2026-04-25

## Table of contents



A functional machine learning workflow treats the analysis as a sequence of explicit transformations. Instead of hiding modeling decisions inside one opaque object, the analyst moves through stable verbs such as fitting, resampling, evaluating, comparing, interpreting, and estimating effects.[1]

A compact supervised learning workflow can be written as

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}, \qquad f_\theta : \mathcal{X} \to \mathcal{Y},$$

where $f_\theta$ is fitted on training data and assessed on validation folds. For a metric $m$, $K$-fold cross-validation estimates performance as

$$\widehat{M}_{CV} = \frac{1}{K} \sum_{k=1}^{K} m\left(y_{V_k}, \hat{f}^{(-k)}(x_{V_k})\right),$$

where $\hat{f}^{(-k)}$ is trained without validation fold $V_k$. This formula is simple, but it is where many reproducibility errors enter: fold definitions, preprocessing, outcome encoding, metrics, and seeds must be visible.

[1] `funcml` package documentation, CRAN: https://CRAN.R-project.org/package=funcml.

### Example

The `funcml` package exposes these workflow steps as explicit functions.[2]

### Quick Example

The shortest useful workflow is to fit a learner and obtain predictions on new patients.

```r
library(funcml)

data <- pimadiabetes
data$diabetes <- factor(data$diabetes)

train <- subset(data, split == "train")[1:80, ]
test <- subset(data, split == "test")[1:5, ]

glm_fit <- fit(
  diabetes ~ glu + bmi + age,
  data = train,
  model = "glm",
  seed = 1
)

glm_fit
```

```
<funcml_fit> classification model: glm
Formula: diabetes ~ glu + bmi + age
Features: 3 | Obs: 80
```

The fitted object carries a stable prediction method.

```r
round(glm_fit$predict(test, type = "prob"), 3)
```

```
       No   Yes
201 0.288 0.712
202 0.947 0.053
203 0.956 0.044
204 0.950 0.050
205 0.078 0.922
```

[2] `funcml` R-universe builds and documentation: https://ielbadisy.r-universe.dev/funcml.

```r
glm_fit$predict(test, type = "class")
```

```
[1] Yes No  No  No  Yes
Levels: No Yes
```

### Cross-Validated Example

```r
library(funcml)

train <- subset(data, split == "train")[1:120, ]

list_metrics(columns = c("metric", "direction")) |>
  head(8)
```

```
     metric direction
1      rmse  minimize
2       mae  minimize
3       mse  minimize
4     medae  minimize
5      mape  minimize
6       rsq  maximize
7  accuracy  maximize
8 precision  maximize
```

We can evaluate a logistic regression classifier using three-fold cross-validation and two metrics: accuracy and AUC.

```r
set.seed(1)

res <- evaluate(
  diabetes ~ glu + bmi + age + bp,
  data = train,
  model = "glm",
  resampling = cv(3),
  metrics = c("accuracy", "auc"),
  seed = 1
)

res
```

```
<funcml_eval> model: glm | task: classification
    metric      mean         sd n  std_error conf_level  conf_low conf_high
1 accuracy 0.7002918 0.01751277 3 0.01011100       0.95 0.6567877 0.7437960
2      auc 0.7885502 0.04031014 3 0.02327307       0.95 0.6884143 0.8886862
```

The output reports the mean, standard deviation, standard error, and confidence interval across folds. In this run, the glucose/BMI/age/blood-pressure model reaches an AUC around 0.79 under three-fold cross-validation, while accuracy is around 0.70. For a health-data workflow, the point is not only the numeric result: it is that the resampling plan and metrics are explicit in the code.

## Methodological Use

This pattern is useful when the analysis needs to be audited:

1. The formula states the outcome and candidate predictors.
2. The model name states the learner.
3. The resampling object states how validation is performed.
4. The metric vector states how performance is judged.
5. The seed makes the split reproducible.

The same structure can be extended to compare multiple learners, tune hyperparameters, or estimate treatment effects. The important methodological constraint is that preprocessing should be declared before fitting and applied inside the resampling workflow when it is learned from data, otherwise information can leak from validation folds into training folds.[3]

[3] Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning.* Springer.