# mlrFDA: an R toolbox for functional data analysis

Laura Beggel, Xudong Sun, Florian Pfisterer, Bernd Bischl

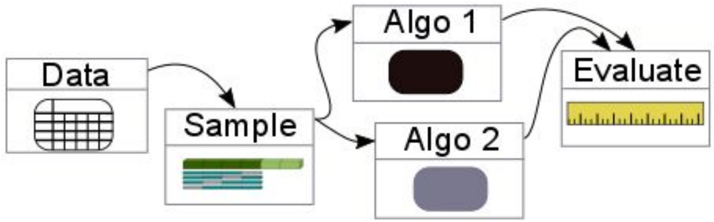
Ludwig-Maximilians-Universität München

Statistical Computing, 07/24/2017

BOSCH

# mlr

- How does your code for experiments look like?




-> mlr: unified interface for machine learning algorithms

- Supported tasks: classification, clustering, regression, survival
- for performance evaluation, resampling, preprocessing and feature selection, hyperparameter tuning,

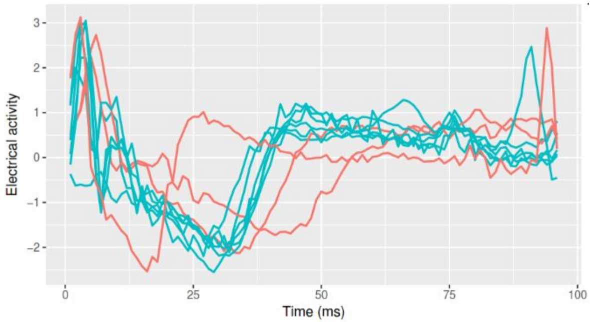


BOSCH

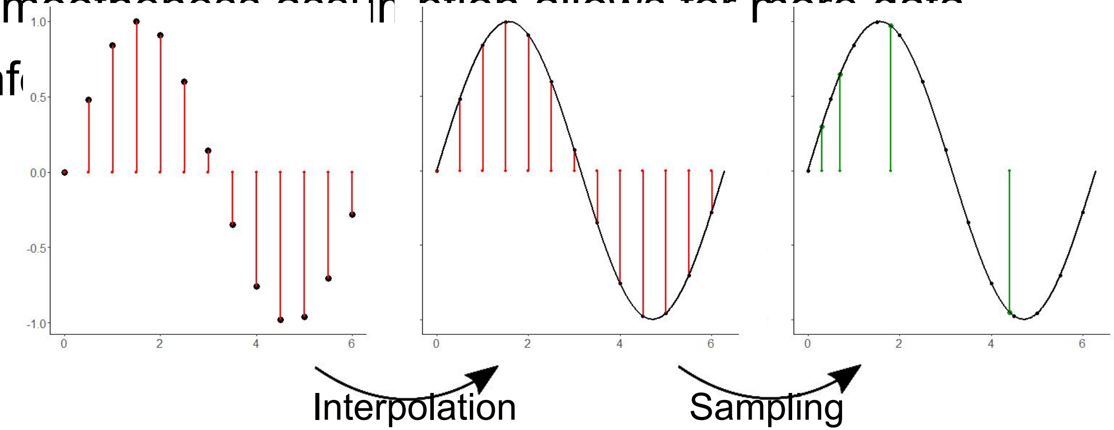
# Functional data

[1]

- Data is sampled over an ordered continuum
- Here: time



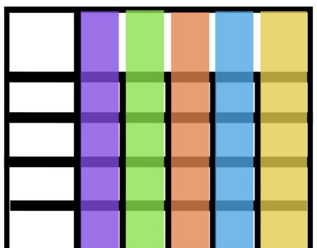

- Smoothness assumption allows for more data inf

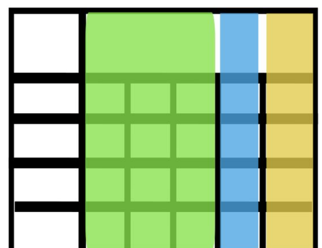



[1] Psych.mcgill.ca/misc/fda/index.html

# mlrFDA
## Data handling

- mlr internally: data frame with annotations
- NOW: Functional features

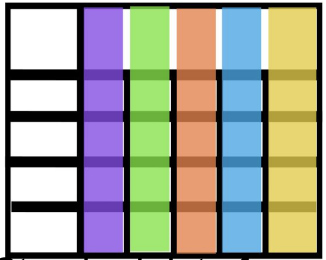

Standard data frame


Functional data frame

# mlrFDA
## Data handling

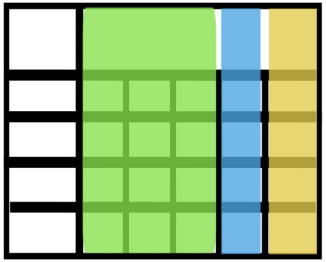
```
df = data.frame(matrix(rnorm(100), nrow = 10),
  "target" = as.factor(sample(1:2, 10, replace = TRUE)))
str(df)
```

```
## 'data.frame':    10 obs. of  11 variables:
##  $ X1    : num  -0.5605 -0.2302 1.5587 0.0705 0.1293 ...
##  $ X2    : num  1.224 0.36 0.401 0.111 -0.556 ...
##  $ X3    : num  -1.068 -0.218 -1.026 -0.729 -0.625 ...
##  $ X4    : num  0.426 -0.295 0.895 0.878 0.822 ...
##  $ X5    : num  -0.695 -0.208 -1.265 2.169 1.208 ...
##  $ X6    : num  0.2533 -0.0285 -0.0429 1.3686 -0.2258 ...
##  $ X7    : num  0.38 -0.502 -0.333 -1.019 -1.072 ...
##  $ X8    : num  -0.491 -2.309 1.006 -0.709 -0.688 ...
##  $ X9    : num  0.00576 0.38528 -0.37066 0.64438 -0.22049 ...
##  $ X10   : num  0.994 0.548 0.239 -0.628 1.361 ...
##  $ target: Factor w/ 2 levels "1","2": 1 2 2 2 1 2 1 1 1 1
```


Standard data frame

```
functionaldf = makeFunctionalData(df, fd.features = list("CO2" = 1:6, "NOX" = 8:10))
str(functionaldf)
```

```
## 'data.frame':    10 obs. of  4 variables:
##  $ X7    : num  0.38 -0.502 -0.333 -1.019 -1.072 ...
##  $ target: Factor w/ 2 levels "1","2": 1 2 2 2 1 2 1 1 1 1
##  $ CO2   : num [1:10, 1:6] -0.5605 -0.2302 1.5587 0.0705 0.1293 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr  "X1" "X2" "X3" "X4" ...
##  $ NOX   : num [1:10, 1:3] -0.491 -2.309 1.006 -0.709 -0.688 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr  "X8" "X9" "X10"
```


Functional data frame

# mlrFDA
## Task abstraction

- Supp
  orted

```
tsk = makeClassifTask(data = functionaldf, target = "target")
print(tsk)
```

```
## Supervised task: functionaldf
## Type: classif
## Target: target
## Observations: 10
## Features:
##    numerics     factors     ordered functionals
##           1           0           0           2
## Missings: FALSE
## Has weights: FALSE
## Has blocking: FALSE
## Classes: 2
## 1 2
## 6 4
## Positive class: 1
```

# mlrFDA
## Fda learning algorithms

- Currently implemented ... fdakernel
- Learner consists of training and prediction functions and parameter set specification

```r
lrn = makeLearner("classif.fdaknn")
print(lrn)
```

```
## Learner classif.fdaknn from package fda.usc
## Type: classif
## Name: fdaknn; Short name: fdaknn
## Class: classif.fdaknn
## Properties: twoclass,multiclass,weights,prob,single.functional
## Predict-Type: response
## Hyperparameters: draw=FALSE
```




BOSCH

# mlrFDA
## Fda learning algorithm: fdaknn

- Calculate distance between a pair of functional data

$$\|f(t) - g(t)\|^p = \left( \frac{1}{\int_a^b w(t)dt} \int_a^b |f(t) - g(t)|^p \, w(t)dt \right)^{1/p}$$


- Distance




BOSCH

# mlrFDA
## Fda learning algorithms

```r
modelfit = train(lrn, subsetTask(tsk, features = "CO2"), subset = 1:5)
print(modelfit)
```

```
## Model for learner.id=classif.fdaknn; learner.class=classif.fdaknn
## Trained on: task.id = functionaldf; obs = 5; features = 1
## Hyperparameters: draw=FALSE
```

```r
prediction = predict(modelfit, subsetTask(tsk, features = "CO2"), subset = 6:10)
print(prediction)
```

```
## Prediction: 5 observations
## predict.type: response
## threshold:
## time: 0.00
##    id truth response
## 6   6     2        2
## 7   7     1        2
## 8   8     1        2
## 9   9     1        2
## 10 10     1        2
```

# mlrFDA

## Fda feature extraction

- How can we apply non-functional ML algorithms to functional data?
  - Ignore functional structure
  - Extract

```r
feat.methods = list("CO2" = extractFDAFourier(), "NOX" = extractFDAMean())
newlrn = makeExtractFDAFeatsWrapper("classif.rpart",  feat.methods = feat.methods)
print(newlrn)
```

```
## Learner classif.rpart.extracted from package rpart
## Type: classif
## Name: ; Short name:
## Class: extractFDAFeatsWrapper
## Properties: twoclass,multiclass,missings,numerics,factors,ordered,prob,weights,featimp,functionals
## Predict-Type: response
## Hyperparameters: xval=0
```

```r
newmodel = train(newlrn, tsk, subset = 1:5)
print(newmodel)
```

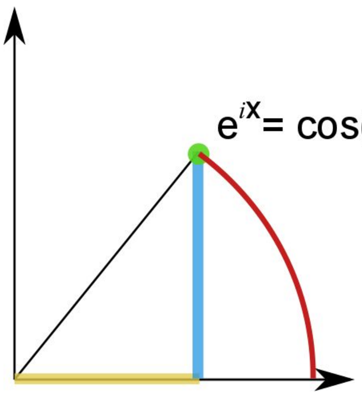
```
## Model for learner.id=classif.rpart.extracted; learner.class=extractFDAFeatsWrapper
## Trained on: task.id = functionaldf; obs = 5; features = 3
## Hyperparameters: xval=0
```

# mlrFDA

## Fda feature extraction: fourier transform

- Key idea: each signal (over time) can be filtered into combinations of circular paths, i.e.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N}$$




Signal over time functional

Signal over frequencies non-fu


CR/AEY2 | 07/24/2017

# Benchmark experiments

[2]

- Compare to UCR Time Series Classification Repository Bakeoff
- 83 classification data sets (univariate time series)
- Fixed train-test split, LOOCV <-> 3fold CV
- tuning via random search (100iters) and MBO (100iters)

## Bake Off

36 (non) time series learners

- Euclidean_1NN
- ST
- BOSS
- COTE
- …

## mlrFDA

4 functional learners

- fdanp
- fdaknn
- fdaglm
- fdakernel

default

6 standard ml learners

- xgboost
- glmnet
- ranger
- rpart
- ksvm
- cvglmnet

default

d +

tuned

t +

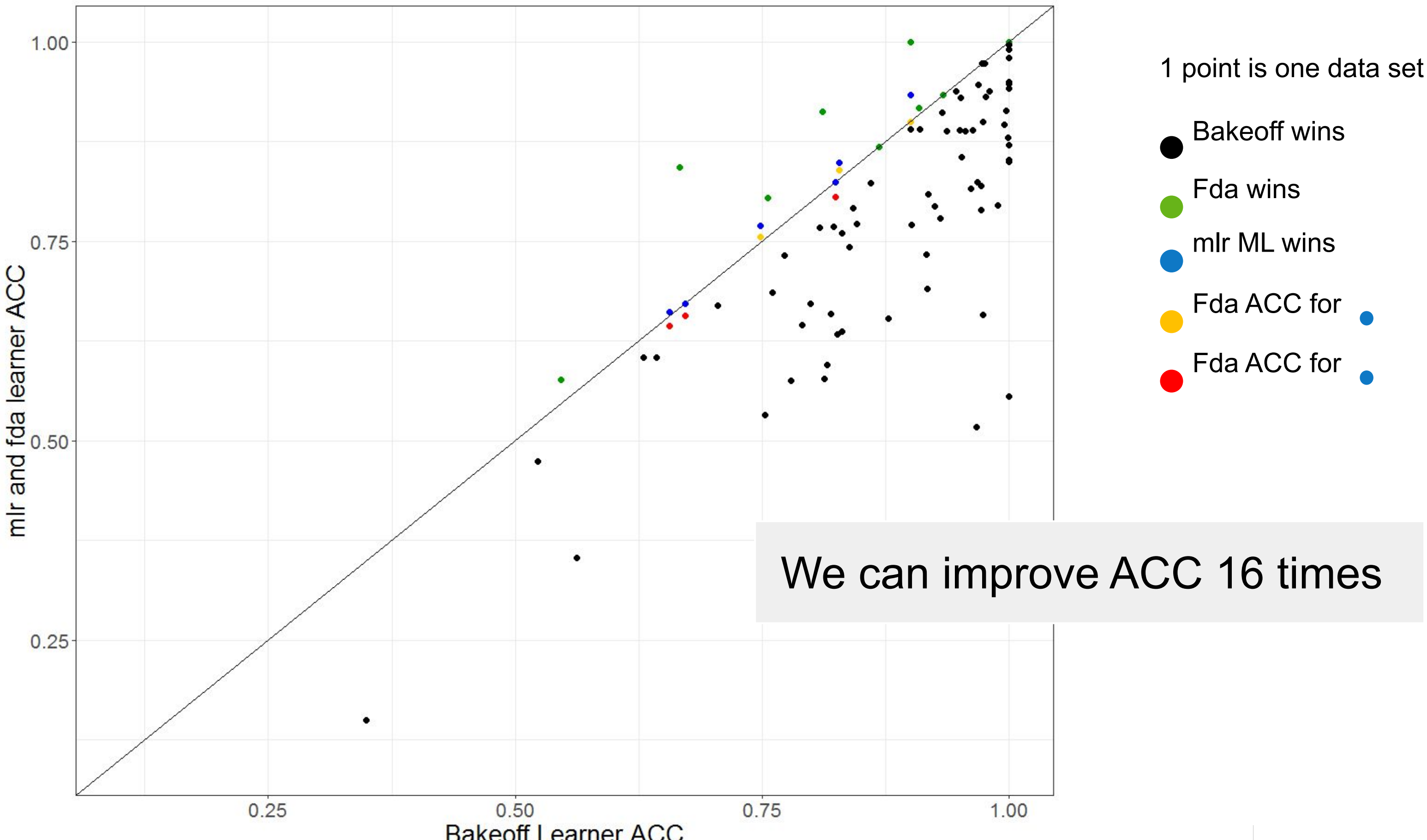
[2]

- Anthony Bagnall, et al., 2016, The Great Time Series Classification Bake Off: a Review and Experimental Evaluation of Recent Algorithmic Advances, Data Mining and

- Anthony Bagnall, et al., The

HEA & UCR Time Series Classification

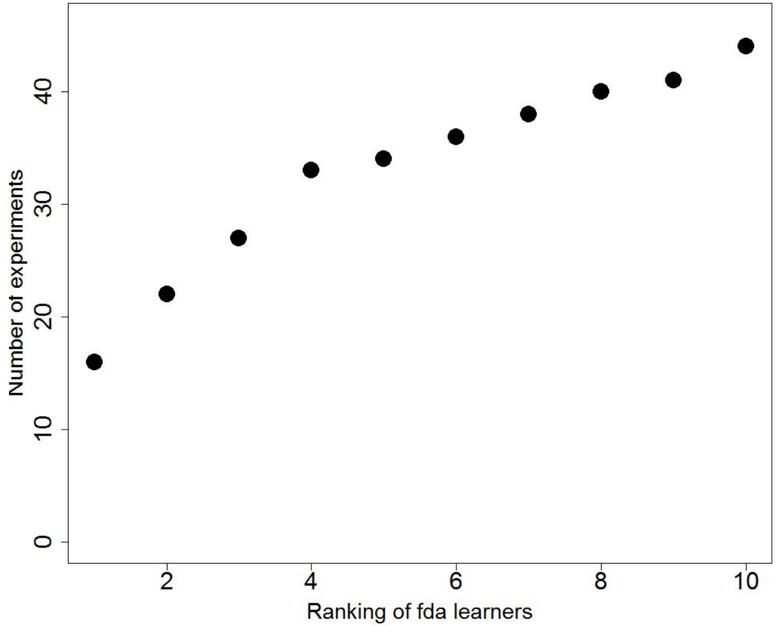
# Benchmark experiments
## Overview ACC



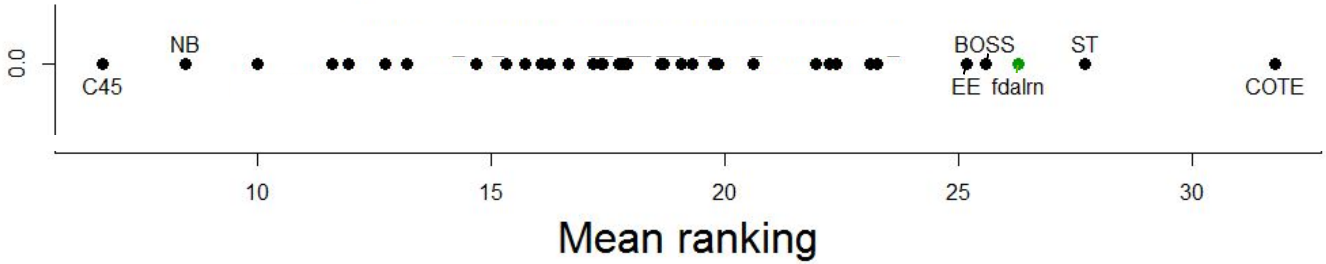

We can improve ACC 16 times


CR/AEY2 | 07/24/2017

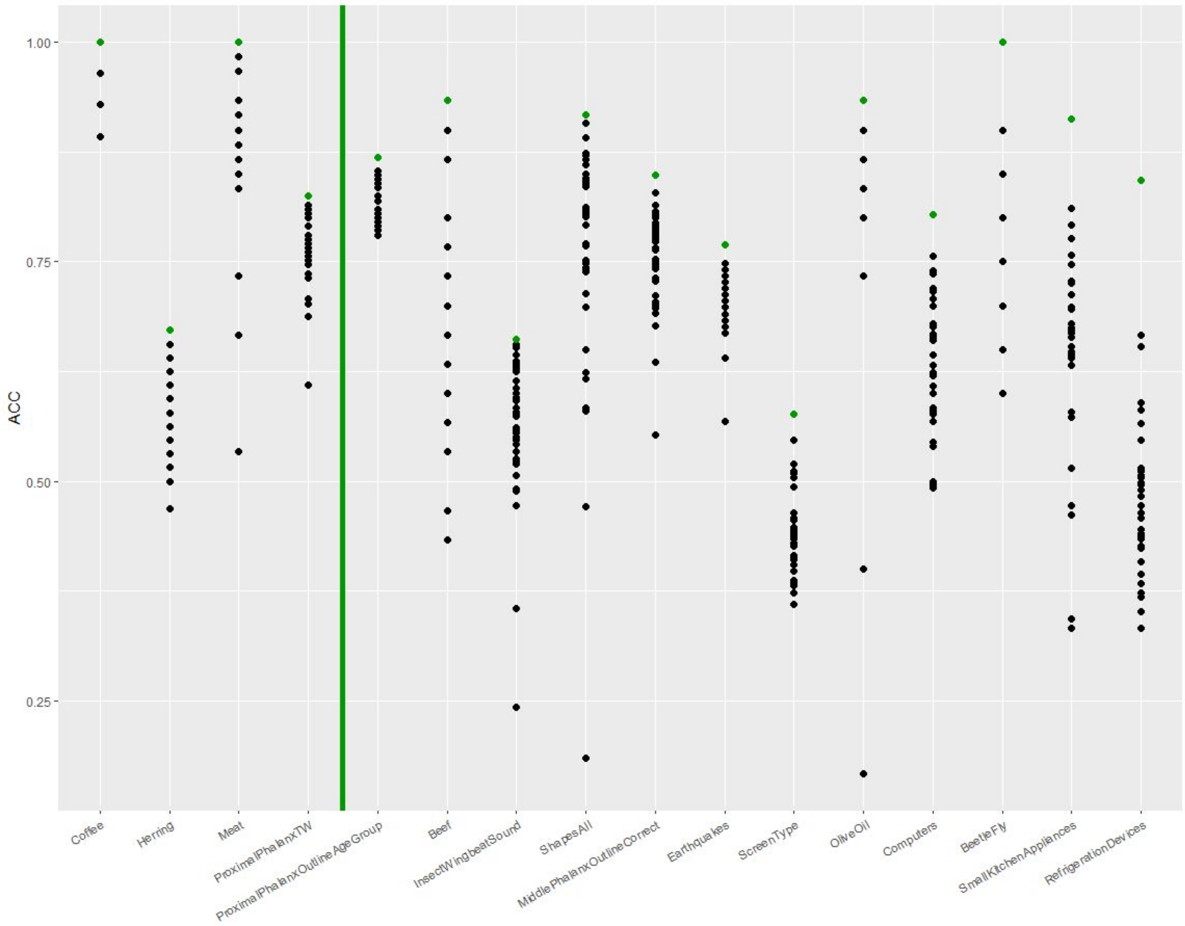
# Benchmark experiments
## Ranking




In more than 0.5 * 83 experiments we are under the 10 best performing learners



# Benchmark experiments
## Individual datasets

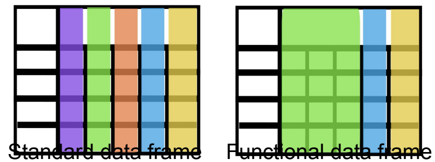




CR/AEY2 | 07/24/2017

# Summary

- How is functional data analysis integrated into mlr?



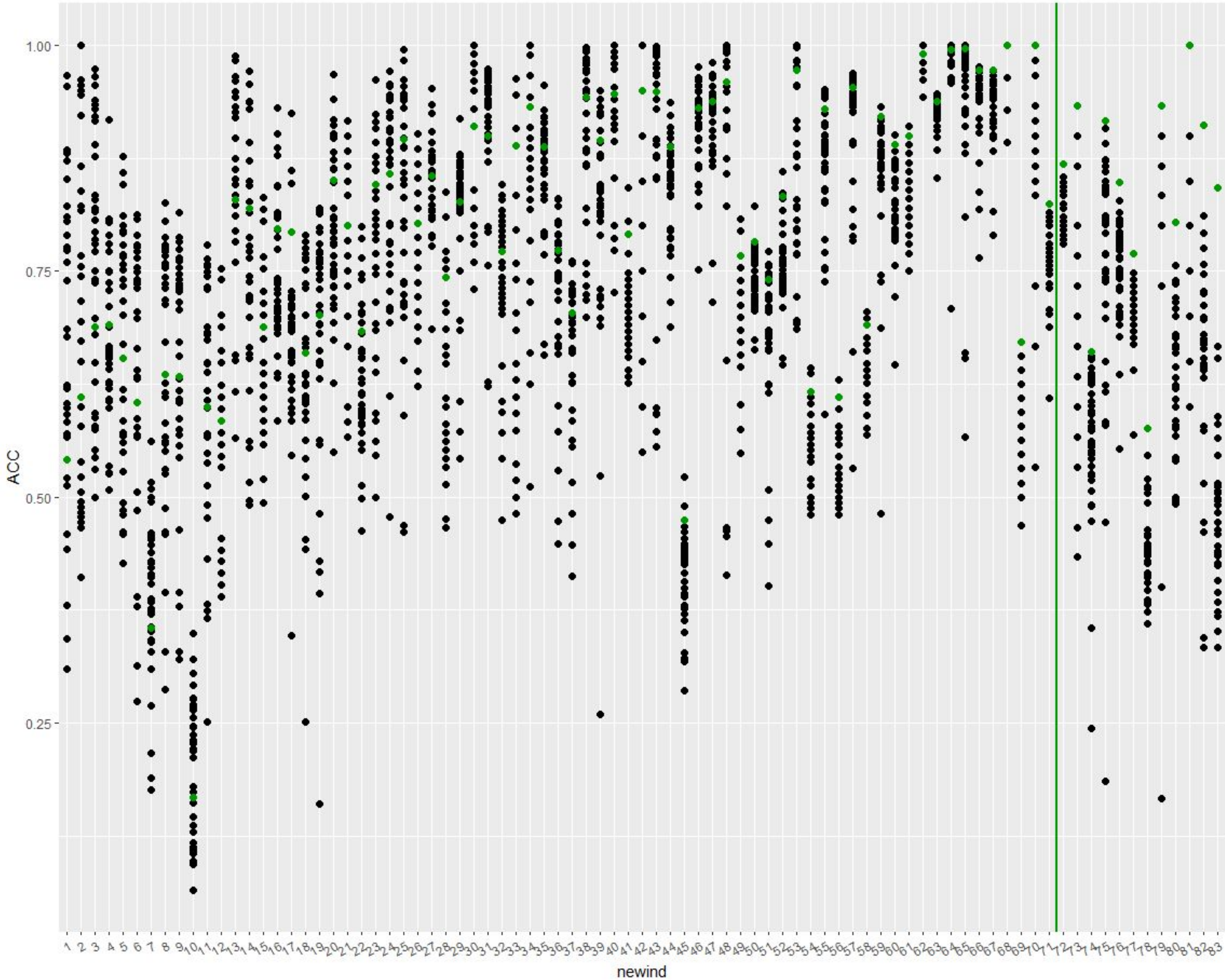

- What can you do with mlrFDA and how?
  - Functional classification and regression
  - Feature extraction for standard ML methods

- Considering functional learners in time series applications is worth a try

# Header of section
## fgam

- Functi
  onal
  gen
  alize
  additi
  ve
  model
  s for
  regre
  ssion
  of
  functi

$$g(E(Y_i)) = \beta_0 + \int_{T_1} F(X_{i1}, t)dt + \int_{T_2} \beta(t)X_{i2}dt + f(z_{i1}) + f(z_{i2}, z_{i3}) + \ldots$$


ACC
1.00
0.75
0.50
0.25
newind
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83