

# Show me your predictive model

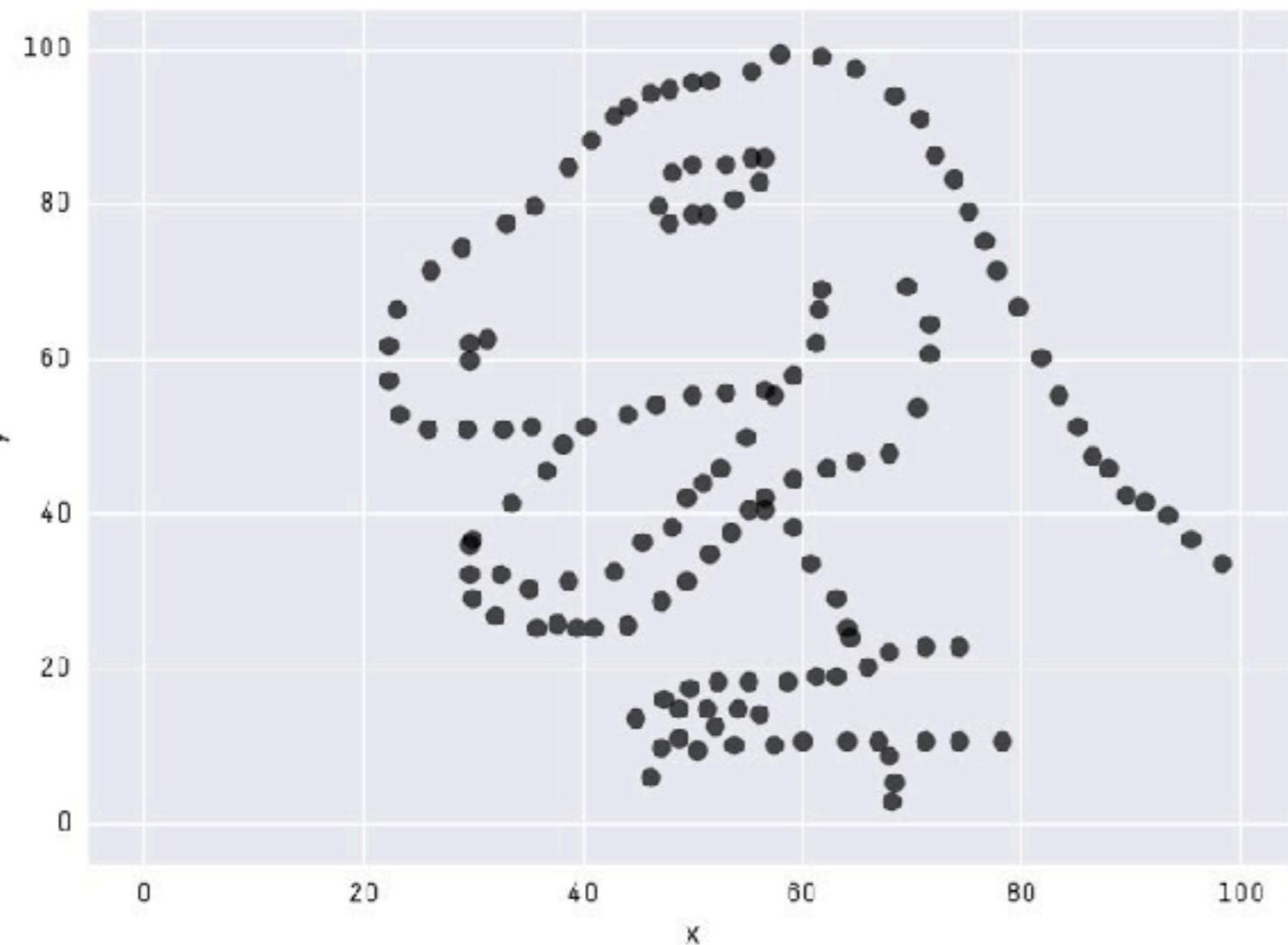
Does datavis help to understand models?  
Why it's important?  
What tools do we have?

**Przemysław Biecek**  
**University of Warsaw**  
**Warsaw University of Technology**



Do we really need plots to understand numbers?

# Package datasauRus

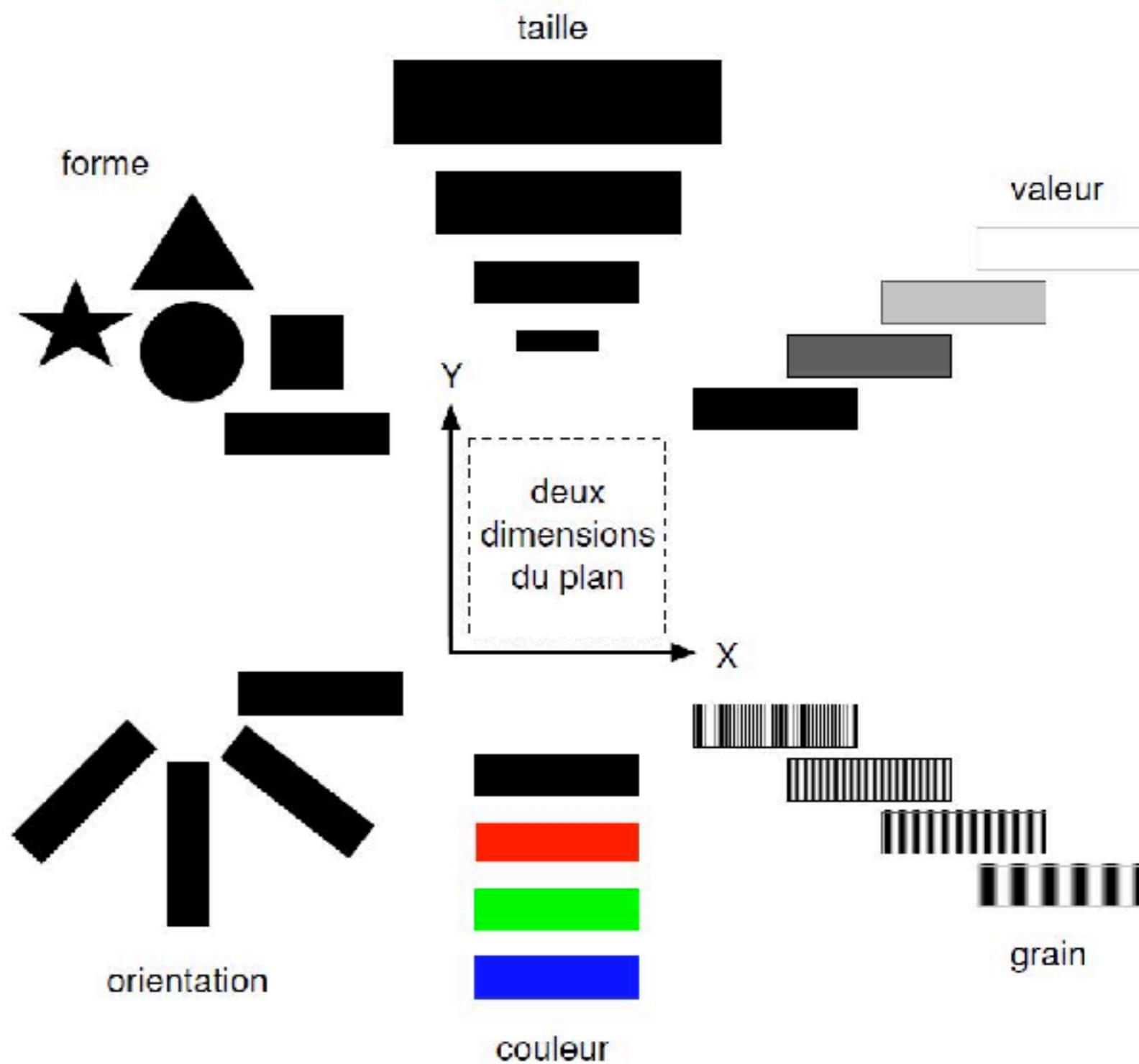


X Mean: 54.2632025  
Y Mean: 47.8315781  
X SD : 16.7650109  
Y SD : 26.9353144  
Corr. : -0.0645195

JACQUES BERTIN

# Semiology of Graphics

*diagrams  
networks  
maps*



Sémiologie Graphique. Jacques Bertin (1967)  
Translation: Semiology of Graphics (1983)

JACQUES BERTIN

# Semiology of Graphics

diagrams  
networks  
maps



John W. Tukey

# EXPLORATORY DATA ANALYSIS



Statistics and Computing

Leland Wilkinson

# The Grammar of Graphics

Second Edition

Springer

UseR!

Hadley Wickham

# ggplot2

Elegant Graphics for Data Analysis

Second Edition

Springer

JACQUES BERTIN

# Semiology of Graphics

diagrams  
networks  
maps



John W. Tukey

# EXPLORATORY DATA ANALYSIS



Statistics and Computing

Leland Wilkinson

# The Grammar of Graphics

Second Edition

Springer

UseR!

Hadley Wickham

# ggplot2

Elegant Graphics for Data Analysis

Second Edition

Springer



Przemysław Biecek

# Odkrywać! Ujawniać! Objaśniać!

Zbiór esejów o sztuce prezentowania danych



JACQUES BERTIN

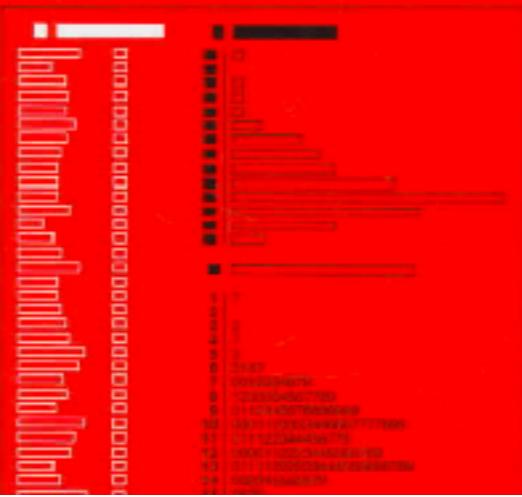
## Semiology of Graphics

diagrams  
networks  
maps



John W. Tukey

## EXPLORATORY DATA ANALYSIS



Statistics and Computing

Leland Wilkinson

## The Grammar of Graphics

Second Edition

UseR!

Hadley Wickham

## ggplot2

Elegant Graphics for Data Analysis

Second Edition

ggplot2

Do we need plots to understand predictive models?

1) ...there is **a lot of opportunity to do visualization for machine learning**. Even many of the people working in the field don't have good intuitions for how their systems work, and they need tools to inspect what they're doing, debug, etc...

<https://eagereyes.org/blog/2017/eurovis-2017-conference-report-part-1>

EuroVis 2017 Conference

Robert Kosara

Senior Research Scientist at Tableau

- 2) Understanding and trust - we need to understand models that makes important decisions for us.
- 3) Machine learning in regulated industry

"Why Should I Trust You?": Explaining the Predictions of Any Classifier  
Marco Tulio Ribeiro, Sameer Singh, Carlos Guestrin (2016)

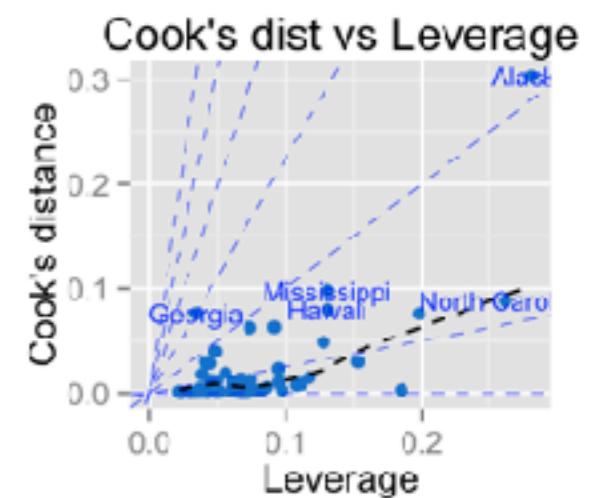
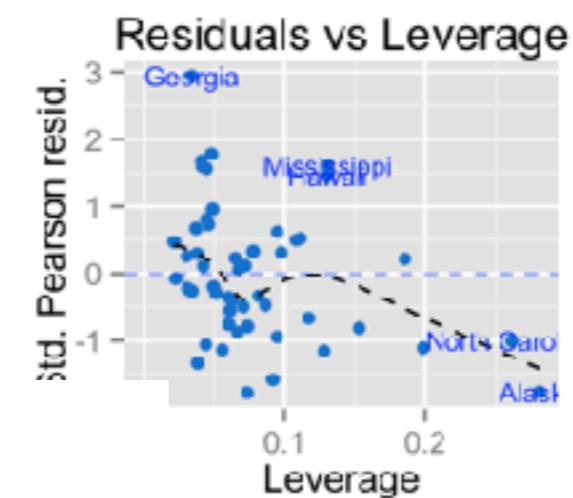
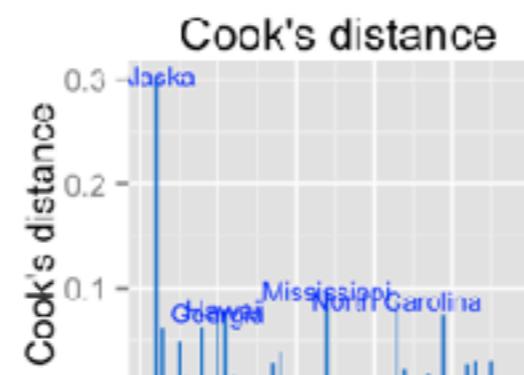
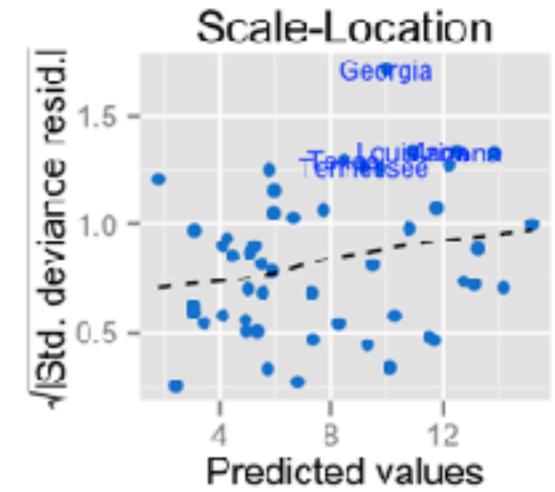
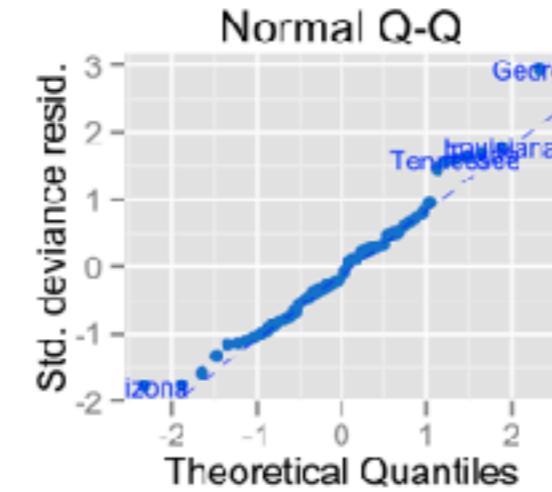
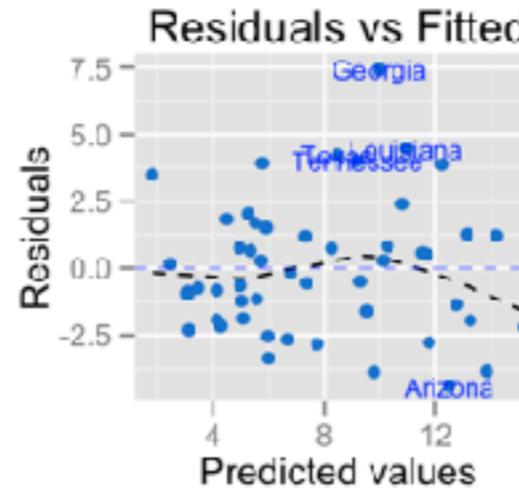
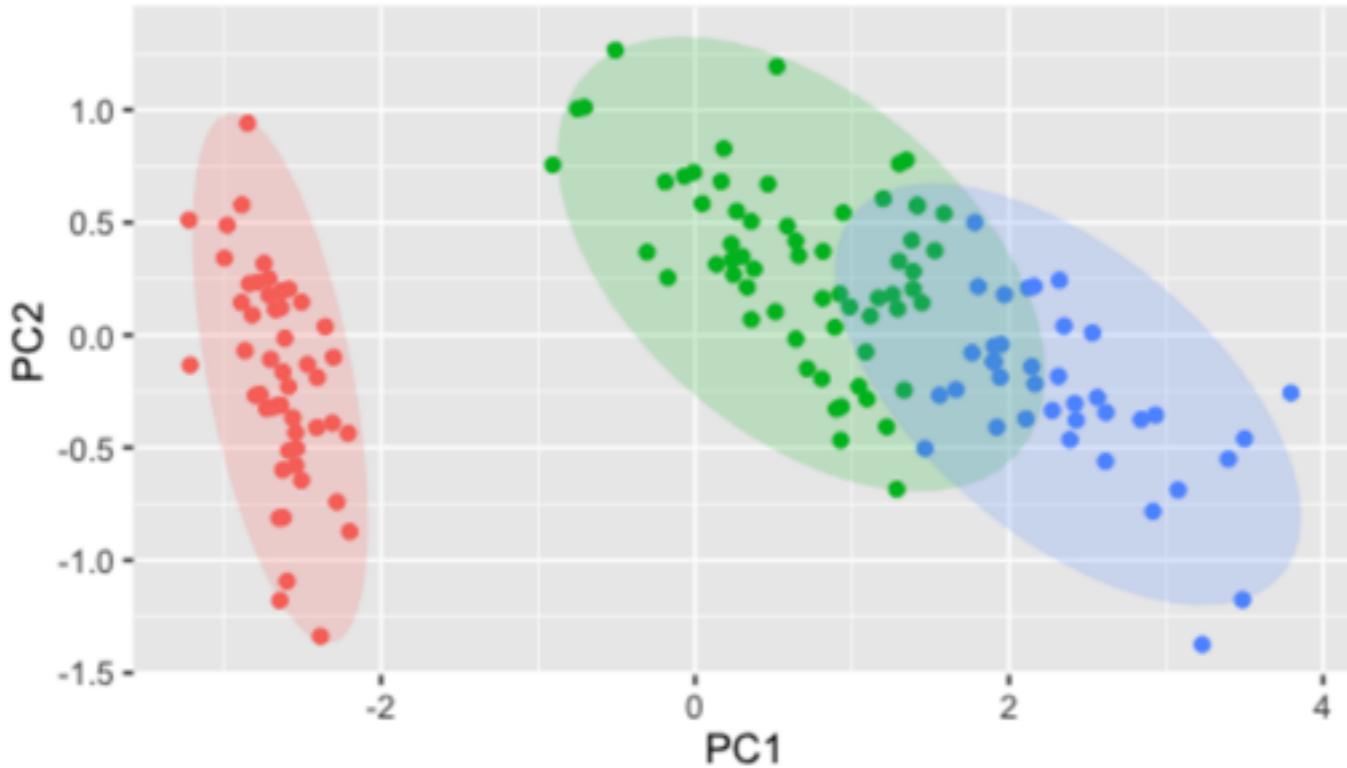
# Package ggfortify

**Table 1:** Supported packages

package	supported types	package	supported types
<b>base</b>	"matrix", "table"	<b>sp</b>	"SpatialPoints", "SpatialPolygons", "Line", "Lines", "Polygon", "Polygons", "SpatialLines", "SpatialLinesDataFrame", "SpatialPointsDataFrame", "SpatialPolygonsDataFrame"
<b>cluster</b>	"clara", "fanny", "pam"	<b>stats</b>	"HoltWinters", "lm", "acf", "ar", "Arima", "stepfun", "stl", "ts", "cmdscale", "decomposed.ts", "density", "factanal", "glm", "kmeans", "princomp", "spec"
<b>changepoint</b>	"cpt"	<b>survival</b>	"survfit", "survfit.cox"
<b>dlm</b>	"dlmFilter", "dlmSmooth"	<b>strucchange</b>	"breakpoints", "breakpointsfull"
<b>fGarch</b>	"fGARCH"	<b>timeSeries</b>	"timeSeries"
<b>forecast</b>	"bats", "forecast", "ets", "nnetar"	<b>tseries</b>	"irts"
<b>fracdiff</b>	"fracdiff"	<b>vars</b>	"varprd"
<b>glmnet</b>	"cv.glmnet", "glmnet"	<b>xts</b>	"xts"
<b>KFAS</b>	"KFS", "signal"	<b>zoo</b>	"zooreg"
<b>lfda</b>	"lfda", "klfda", "self"	<b>MASS</b>	"isoMDS", "sammon"
<b>maps</b>	"map"		

# Package lm ggfortify

PAM

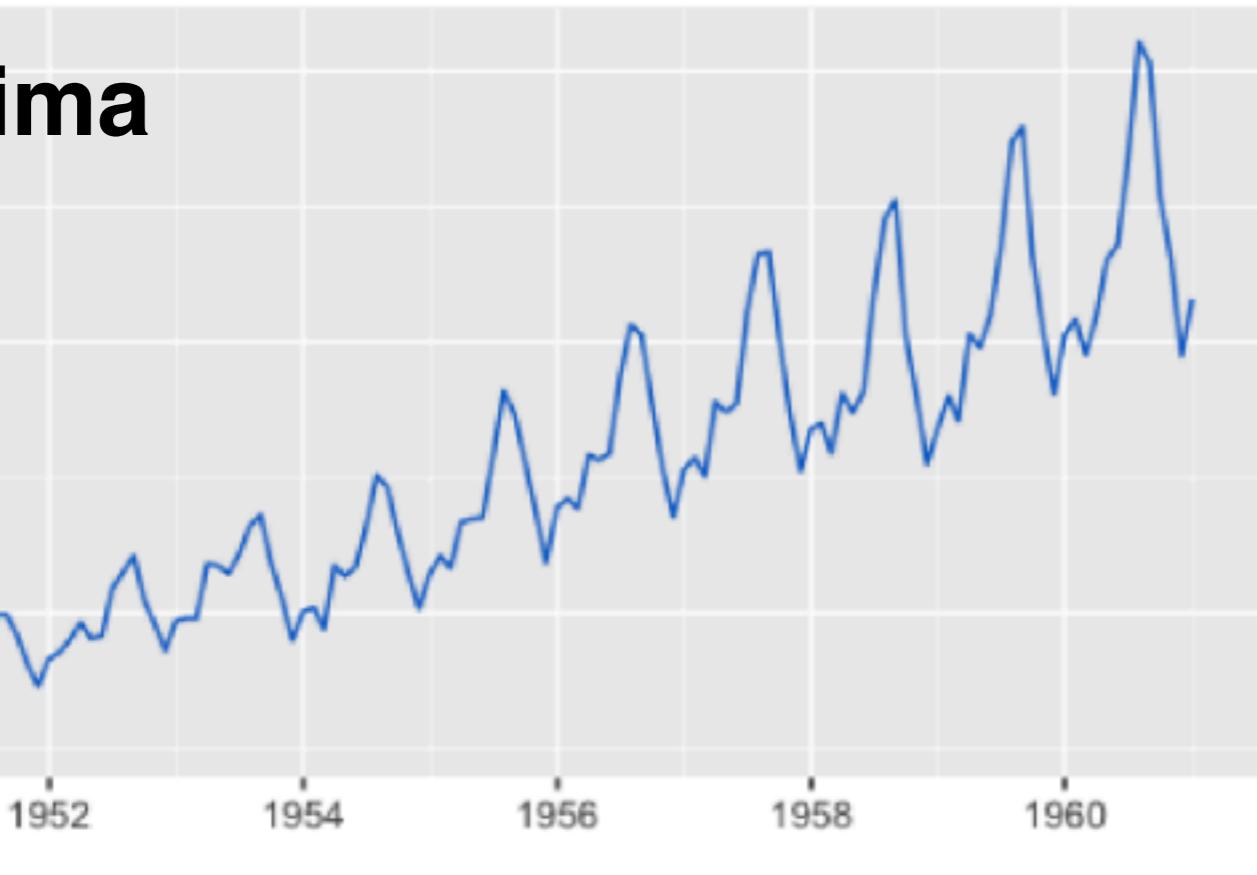


cluster

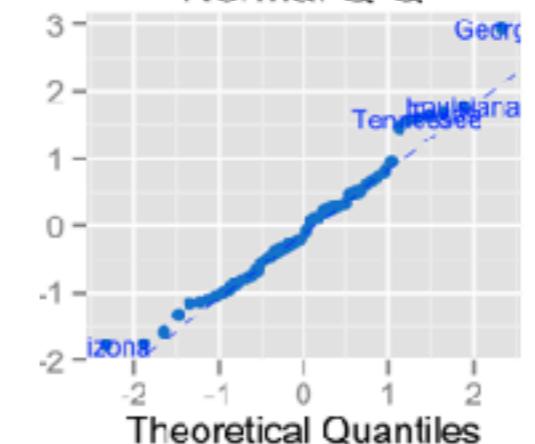
- 1
- 2
- 3

[Yuan Tang, Masaaki Horikoshi, and Wenxuan Li. "ggfortify: Unified Interface to Visualize Statistical Result of Popular R Packages." The R Journal 8.2 \(2016\): 478-489.](#)

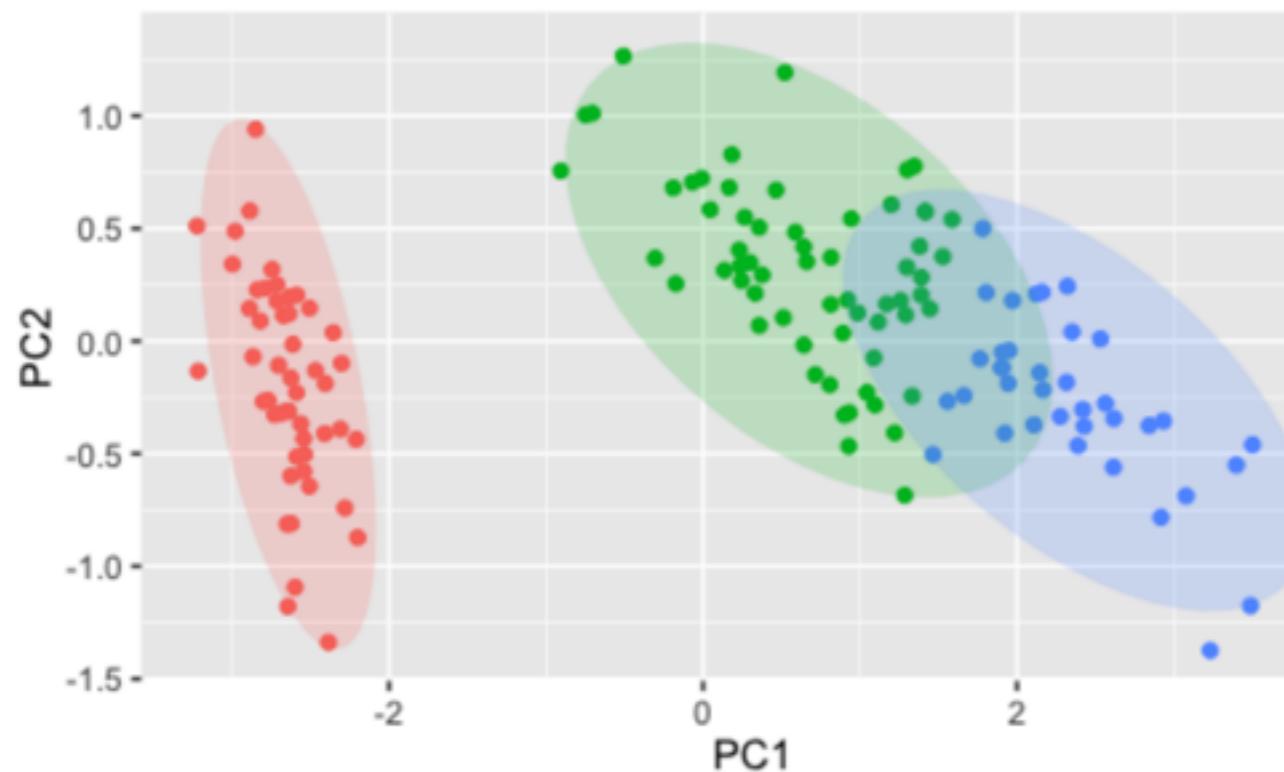
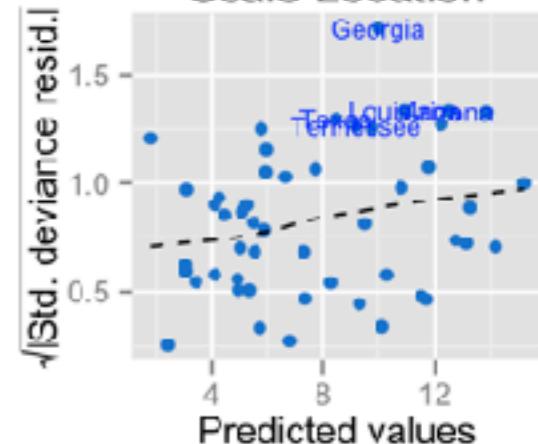
# arima



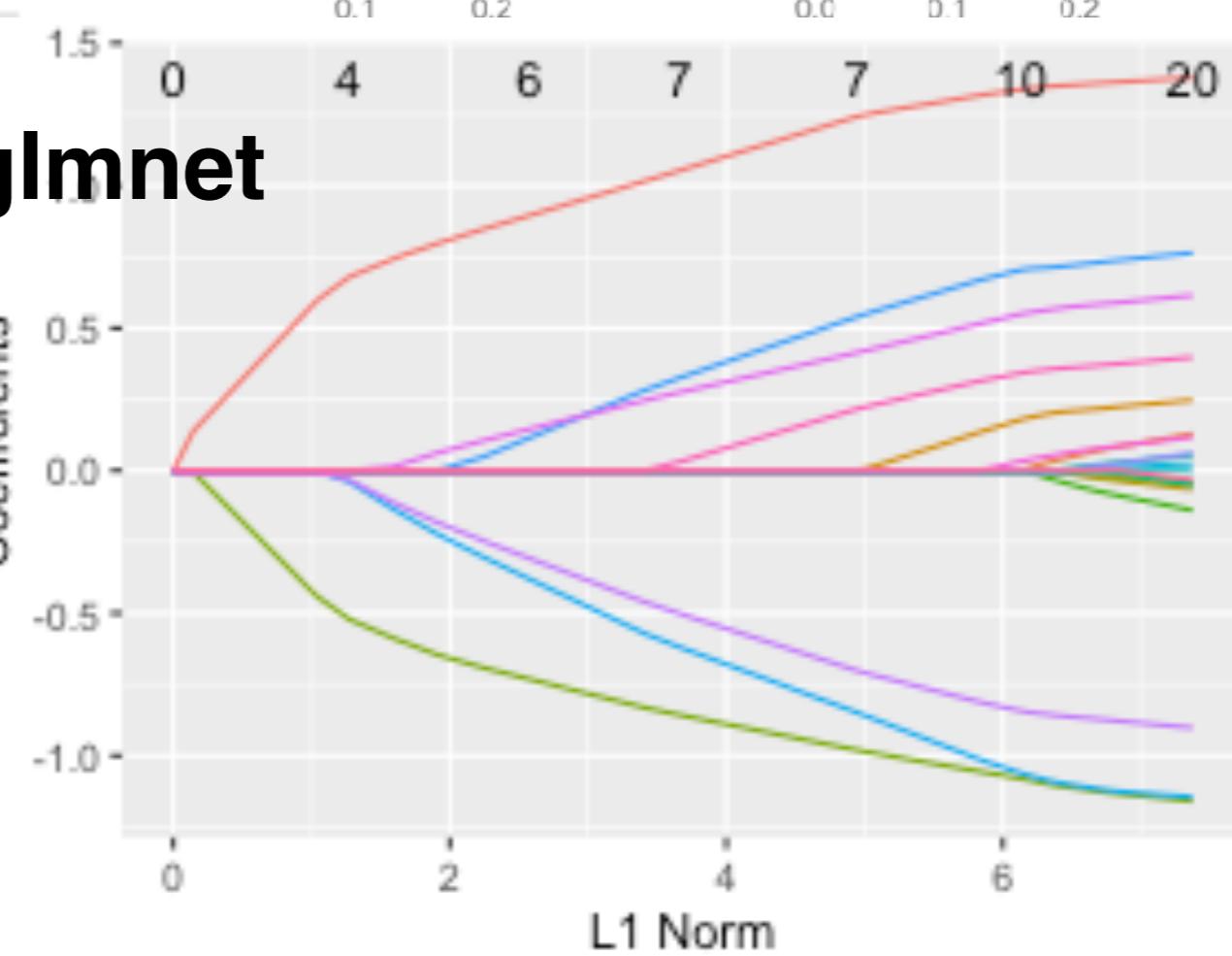
Normal Q-Q



Scale-Location



# glmnet



[Yuan Tang, Masaaki Horikoshi, and Wenzuan  
Statistical Result of Popular R Packages."](#) Th

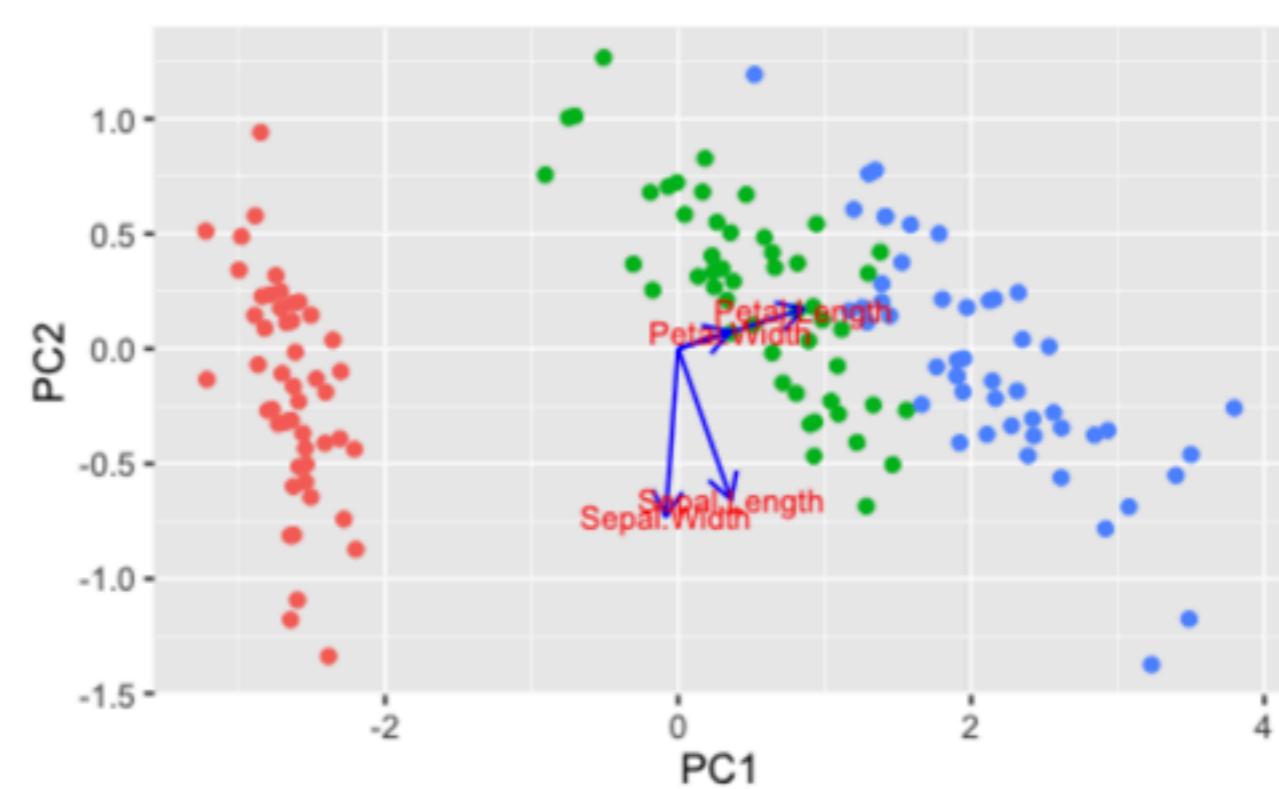
# arima

# aareg



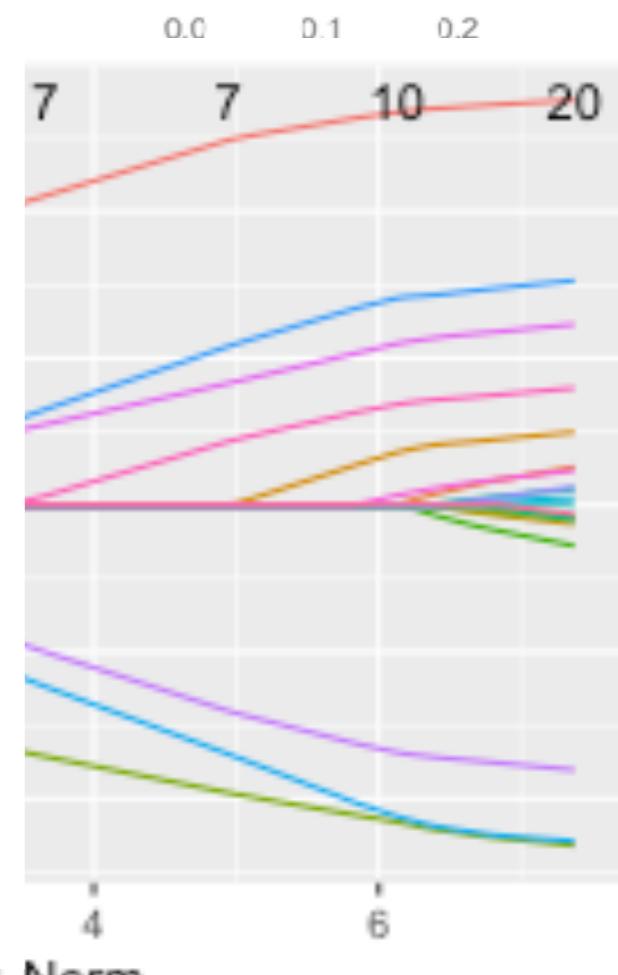
# Normal Q-Q

# Scale-Location



# PCA

Figure 3: PCA with eigen-vectors and labels.



# Package ggfortify

- The main function: `autoplot()`
- Works for large collection for different statistical models
- Plots different things for different models
- Produces `ggplot2` plots

[Yuan Tang, Masaaki Horikoshi, and Wenxuan Li. "ggfortify: Unified Interface to Visualize Statistical Result of Popular R Packages." The R Journal 8.2 \(2016\): 478-489.](#)

# CA with FactoMineR and factoextra

## Basics

**CA** (Correspondence Analysis) is a descriptive technique, that explains dependency between two categorical variables. It is usually applied to contingency tables — decomposes the chi-squared statistic into orthogonal factors, but may be applied to any table with nonnegative values.

It is conceptually similar to principal component analysis. The preprocessed contingency table is decomposed (Generalized Singular Value Decomposition) into orthogonal profiles of rows and columns.

### The Game of Thrones

This example uses the Kaggle dataset about battles in Game of Thrones (see <http://bit.ly/2cJUfQ2>). The **battles** table presents number of battles in which given House (row) was an attacking army with a given attacking strategy (column).

```
head(battles)
#> #>   Battle_Type
#> House      ambush pitched battle razing siege
#> Baratheon    8        2     0     3
#> Frey       1        0     0     2
#> Lannister   1        6     0     2
#> Stark       5        3     0     0
#> Tully       3        0     0     0
```

Use the **FactoMineR::CA()** function for correspondence analysis.

```
library("FactoMineR")
res.ca <- CA(battles, graph = FALSE)
summary(res.ca)
```

The chi square of independence between the two variables is equal to 25.82556 (p-value = 0.183814).

Eigenvalues			
	Dim.1	Dim.2	Dim.3
Variance	0.352	0.201	0.099
% of var.	54.617	38.380	15.824
Cumulative % of var.	54.617	84.976	100.000

Rows				
	Inert1000	Dim.1	ctr	cos2
Baratheon	113.333	0.861	26.281	0.839
Bolton	45.299	-0.568	5.481	0.438
Frey	78.632	0.227	1.099	0.851
Greyjoy	83.333	-0.874	0.307	0.813

Columns				
	Inert1000	Dim.1	ctr	cos2
ambush	217.236	-0.747	59.386	0.989
pitched battle	568.837	0.427	16.766	0.377
razing	99.359	-0.122	0.106	0.884
siege	184.781	0.578	23.342	0.485

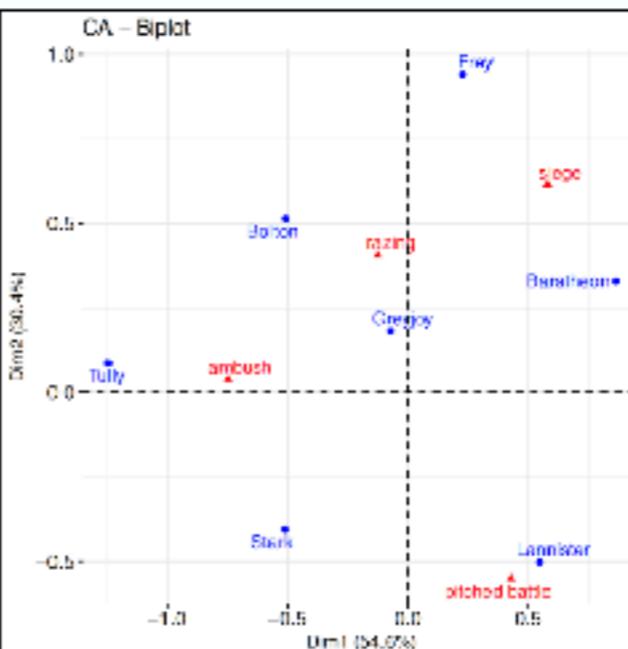
## FactoMineR (for multivariate data analysis) and factoextra (for visualisation of CA results)

### Biplots

Use the **factoextra::fviz\_ca\_biplot()** function to plot the biplot for a given model. Use **fviz\_ca\_row()** or **fviz\_ca\_col()** if you are interested only in profiles of rows/columns.

Consult the help page for numerous parameters of the **fviz\_ca\_biplot()** function, that allows to customise of biplot plots.

```
# Basic biplot, results in the right plot ->
fviz_ca_biplot(res.ca)
# Customised biplot, results in the bottom plot
fviz_ca_biplot(res.ca, repel = TRUE,
               arrow = c(FALSE, TRUE),
               col.row = "contrib",
               col.col = "red4",
               gradient.cols = c("#BBAFBB", "#00AFBB"))
```



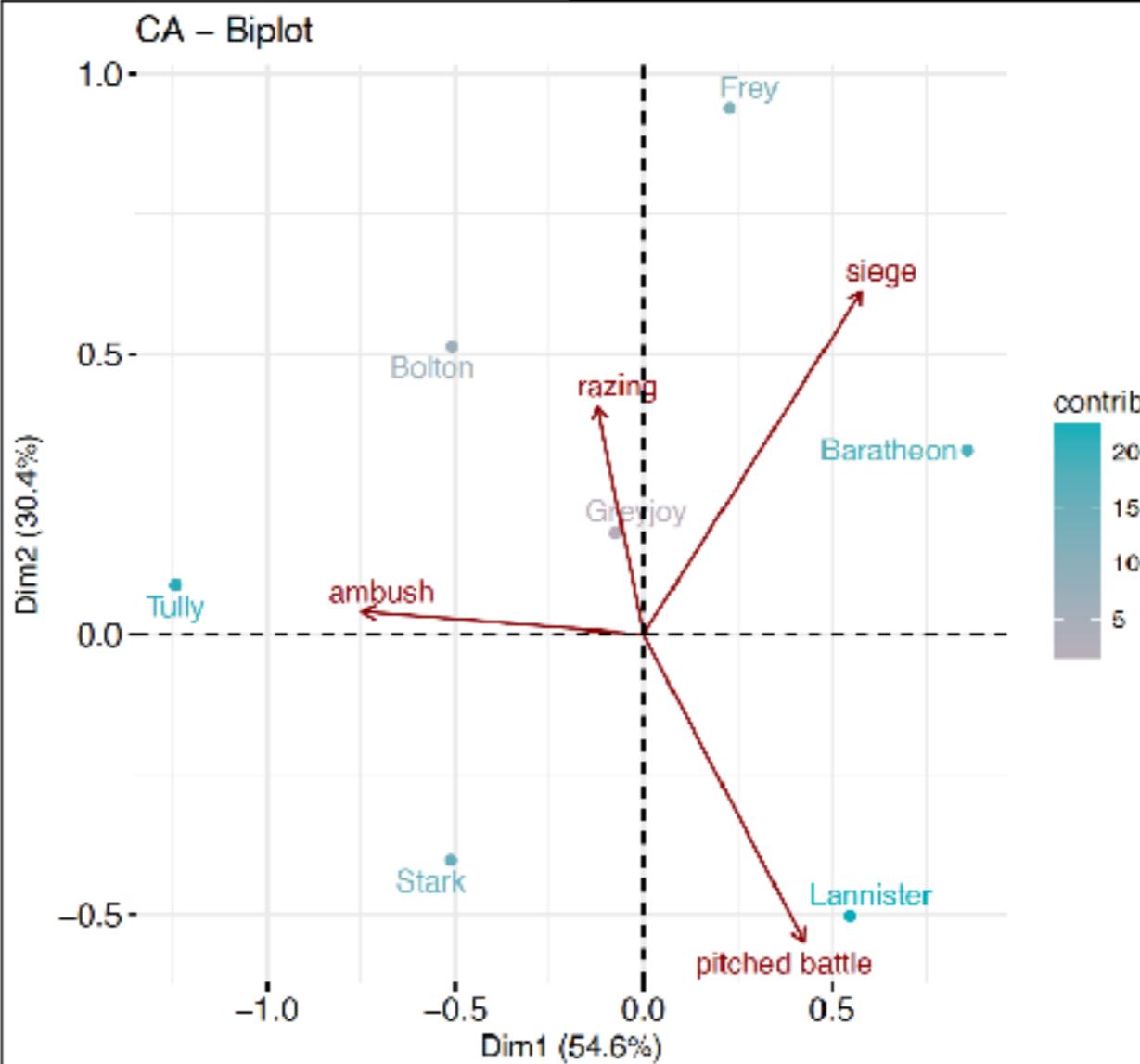
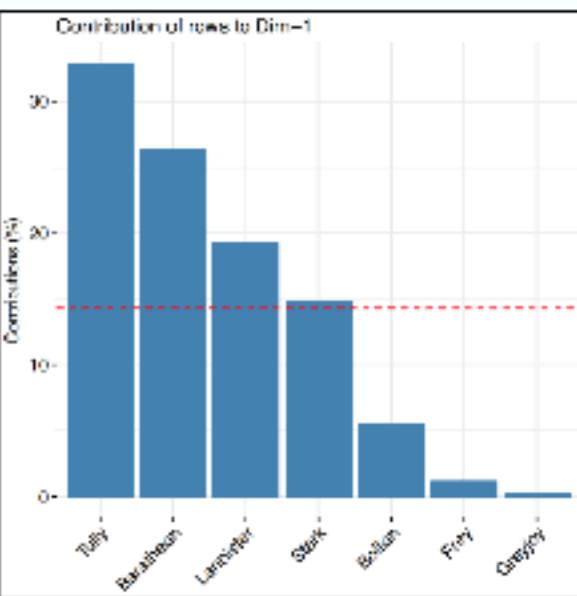
### Row/Column contributions plot

Use the **factoextra::fviz\_contrib()** function to plot contributions of selected dimension (rows or columns) onto a selected axis.

Numerical values for contributions may be extracted with **get\_ca\_row()** or **get\_ca\_col()** functions.

```
# Row contributions to Dimension 1
fviz_contrib(res.ca, choice = "row", axes = 1)
# Column contributions to Dimension 1
fviz_contrib(res.ca, choice = "col", axes = 1)
# Numeric summaries
get_ca_col(res.ca)
Correspondence Analysis - Results for columns
```

Name	Description
1 "\$cnord"	"Coordinates for the columns"
2 "\$cnns2"	"Cns2 for the columns"
3 "\$contrib"	"contributions of the columns"
4 "\$inertia"	"Inertia of the columns"



# PCA with FactoMineR and factoextra

## Basics

**PCA** (Principal Component Analysis) is a dimension-reduction method. It finds principal factors - orthogonal linear combinations of original variables that explain maximum amount of variance.

$$W_{n \times q} = X_{n \times p} R_{p \times q}$$

The  $p$  dimensional input data  $X$  is projected into a  $q$  dimensional subspace by a linear transformation defined by  $R$ . New  $q$  dimensional data  $W$  has orthogonal variables. The transformation may be done through SVD decomposition or eigen value decomposition.

## The Example

This example uses data about Hollywood action movies from 2015. Six quantitative variables with movie ratings scrapped from Rotten Tomato and Metacritic websites.

> head(movies2015)					
	Rotten Tomatoes	Rotten Tomatoes	Metacritic Audience	Audience	
Spectre	64	68	65	67	
Furious 7	81	67	84	68	
Terminator Genisys	25	38	59	63	
San Andreas	58	43	56	55	
Point Break	9	38	37	22	

Use the **FactoMineR::PCA()** function for PCA with supplementary quantitative and categorical variables. Missing values will be replaced by `colMeans`.

```
> library("FactoMineR")
> model <- PCA(movies2015)
> summary(model)
Eigenvalues
```

	Dim.1	Dim.2	Dim.3	Dim.4	Dim.5
Variance	4.474	0.355	0.131	0.040	0.00
% of var.	89.481	7.893	2.027	0.790	0.00
Cumulative % of var.	89.481	96.574	99.202	100.000	100.00

```
Individuals
```

	Dist	Dim.1	ctr	cos2
Spectre	1.077	0.980	2.184	0.842
Furious 7	2.488	2.321	12.045	0.930
Terminator Genisys	1.694	-1.394	4.341	0.877
San Andreas	0.811	-0.784	1.188	0.754
Point Break	3.043	-3.491	26.767	0.982
Run All Night	1.192	0.842	0.584	0.490
No Escape	1.076	-0.506	0.577	0.223

```
Variables
```

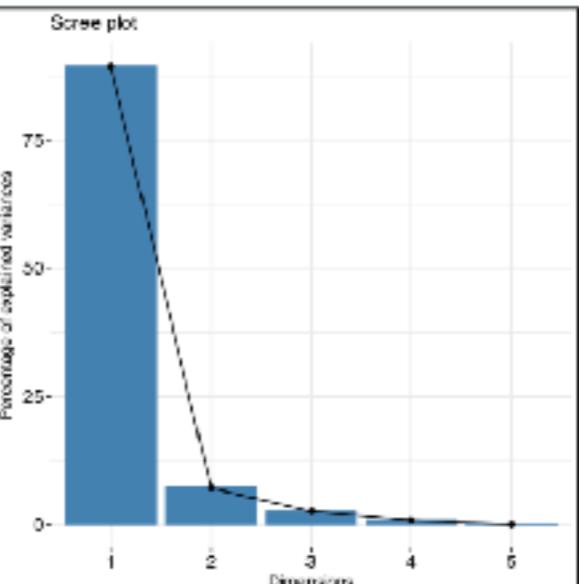
	Dim.1	ctr	cos2	Dim.2
Rotten.Tomatoes	0.988	21.836	0.077	-0.050
Metacritic	0.931	19.389	0.067	-0.338
Average.critics	0.986	21.721	0.072	-0.156
Rotten.Tomatoes.Audience	0.943	19.885	0.080	0.135
Metacritic.Audience	0.876	17.169	0.708	0.447

## FactoMineR (for multivariate data analysis) and factoextra (for visualisation of PCA results)

### Scree plot

Use the **factoextra::get\_eig()** function to extract information about eigenvalues. The **factoextra::fviz\_screenplot()** function will plot the percentage of variance explained by each principal factor.

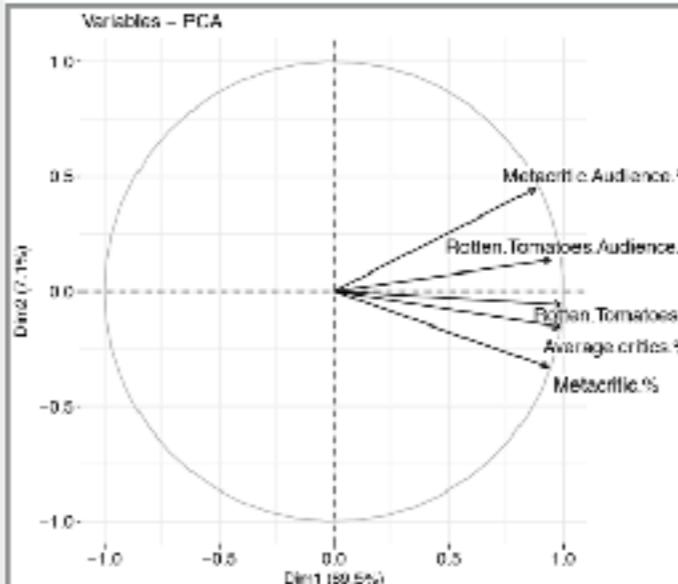
```
> get_eig(model)
#> #> #> eigenvalue variance.percent cum.variance.percent
#> Dim.1 4.474839e+00 8.9488e-01 89.48
#> Dim.2 3.546706e-01 7.8934e-01 96.57
#> Dim.3 1.313722e-01 2.6273e-01 99.20
#> Dim.4 3.991024e-02 7.9036e-01 100.00
#> Dim.5 5.236294e-02 1.0512e-01 100.00
> fviz_screenplot(model)
```



### PCA variables' plot

Use the **factoextra::fviz\_pca\_var()** function to plot contribution of original variables into selected (the **axes** argument) principal components. Show variables through text labels or arrows (the **geom** argument). Result of this function is the **ggplot2** plot.

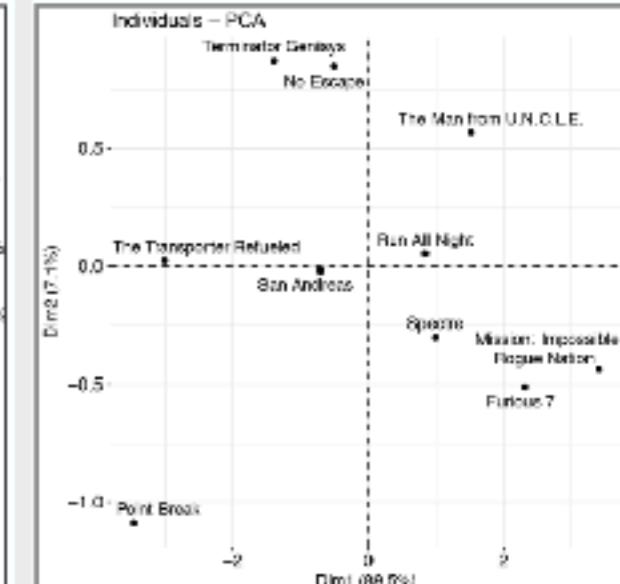
```
> fviz_pca_var(model)
```



### PCA individuals' plot

Use the **factoextra::fviz\_pca\_ind()** function to plot observations with selected (the **axes** argument) principal coordinates. With the **habillage** argument one can select a grouping variable which will be color-coded in the plot. Use **addEllipses** to plot ellipses for each group.

```
> fviz_pca_ind(model)
```



## PCA - Biplot

Use the **factoextra::fviz\_pca\_biplot()** function to combine results for individuals and variables into a single bi-plot.

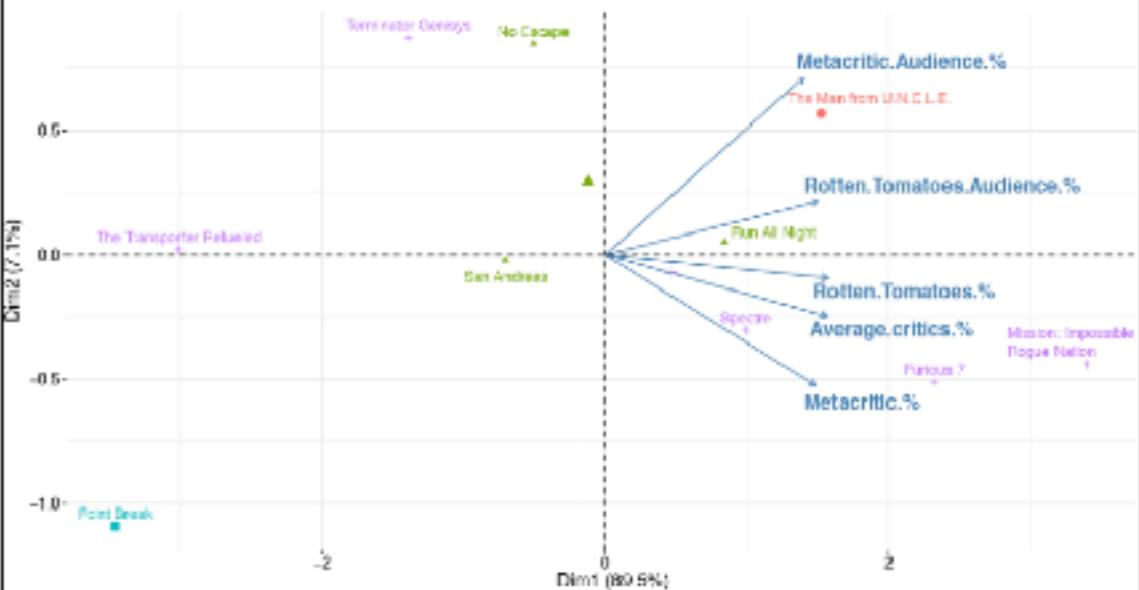
With the **habillage** argument one can select a grouping variable which will be color-coded in the plot. Use **addEllipses** to plot ellipses for each group.

In the presented example, the first principal coordinate is highly correlated with average rating from all sources (audience and critics) while the second principal coordinate discriminate between audience and critics. Thus one can easily identify movies that are preferred by critics and those preferred by audience.

```
> fviz_pca_biplot(model, habillage = films2015$script.type) +
  theme(legend.position = "top")
```

### PCA - Biplot

Groups: ● animation ▲ original screenplay ■ remake/fabool □ sequel

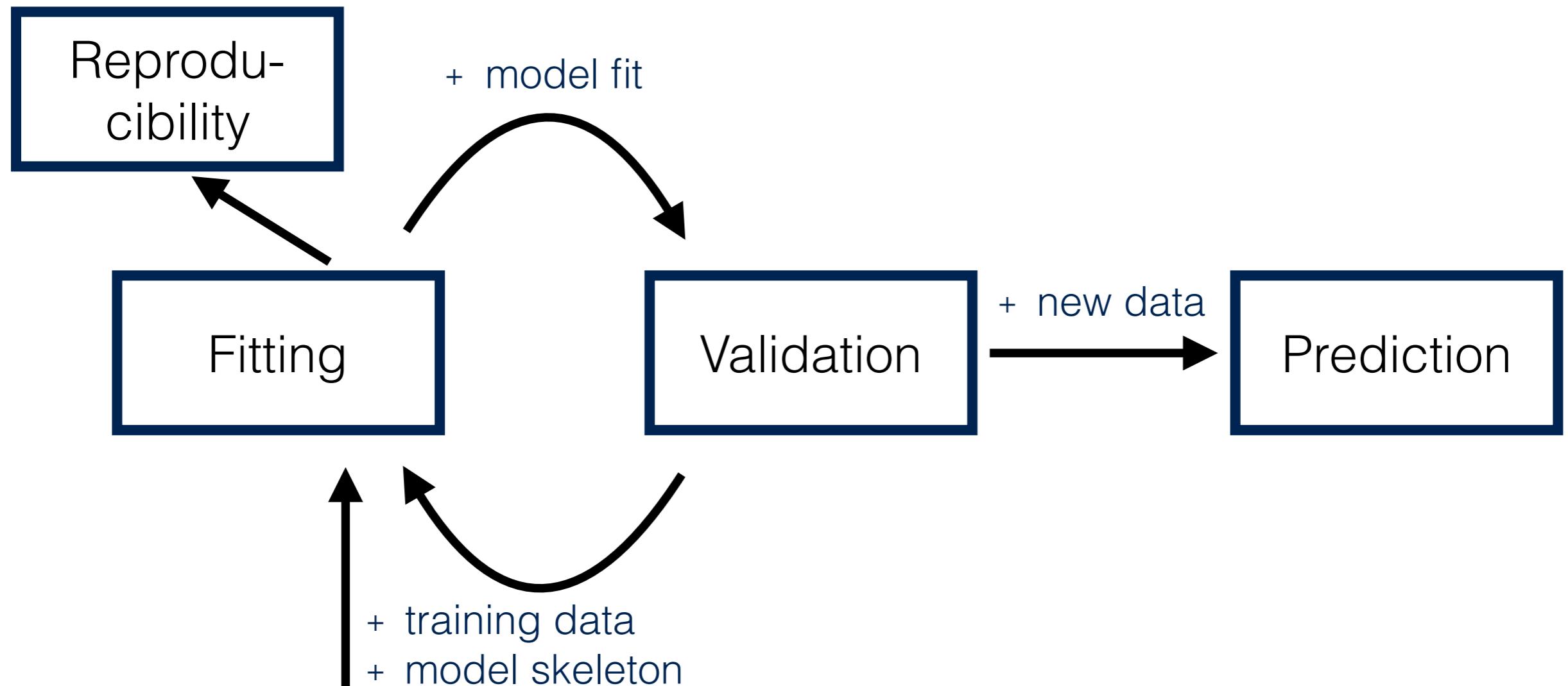


# Package factoextra

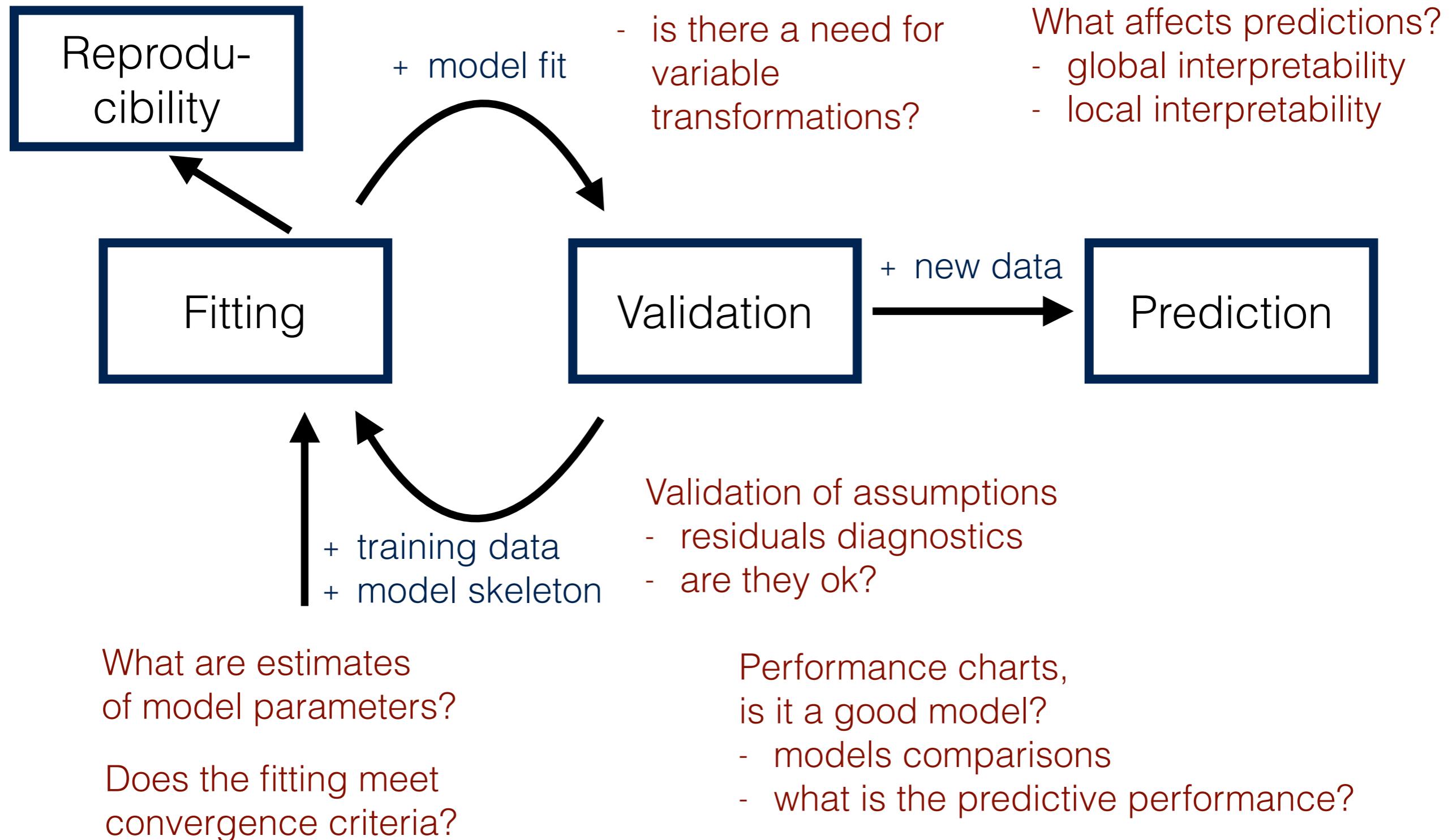
- Set of functions with many arguments
- Works for few selected models
- Plots many (all popular) graphical summaries for given models
- Produces ggplot2 plots

[Yuan Tang, Masaaki Horikoshi, and Wenxuan Li. "ggfortify: Unified Interface to Visualize Statistical Result of Popular R Packages." The R Journal 8.2 \(2016\): 478-489.](#)

# Life-cycle of a typical prognostic model

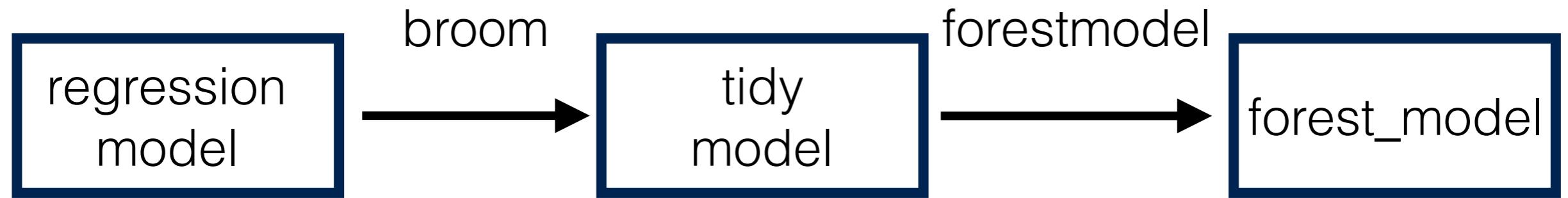


# Life-cycle of a typical prognostic model

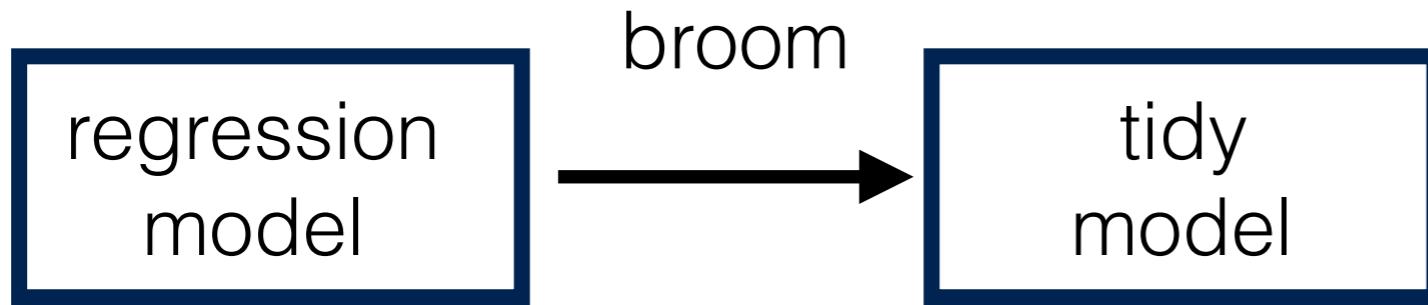


# Fitting

# Package forestmodel



# Package forestmodel



Class	tidy	glance	spec	x	
aareg	x	x	stanfit	x	
acf	x		stanreg	x	x
anova	x		summary.glht	x	
aov	x		summary.lm	x	x
aovlist	x		summaryDefault	x	x
Arima	x	x	survexp	x	x
betareg	x	x	survfit	x	x
biglm	x	x	survreg	x	x

# Package forestmodel



Variable	N	Hazard ratio	p
Age	180	■	1.01 (0.99, 1.03) 0.467
Sex	Male	■	Reference
	Female	■	0.60 (0.41, 0.88) 0.008
ECOG	0	■	Reference
	1	■	1.32 (0.85, 2.04) 0.215
	2	■	2.19 (1.33, 3.61) 0.002
	3	■	6.00 (0.79, 45.70) 0.083

# Package forestmodel

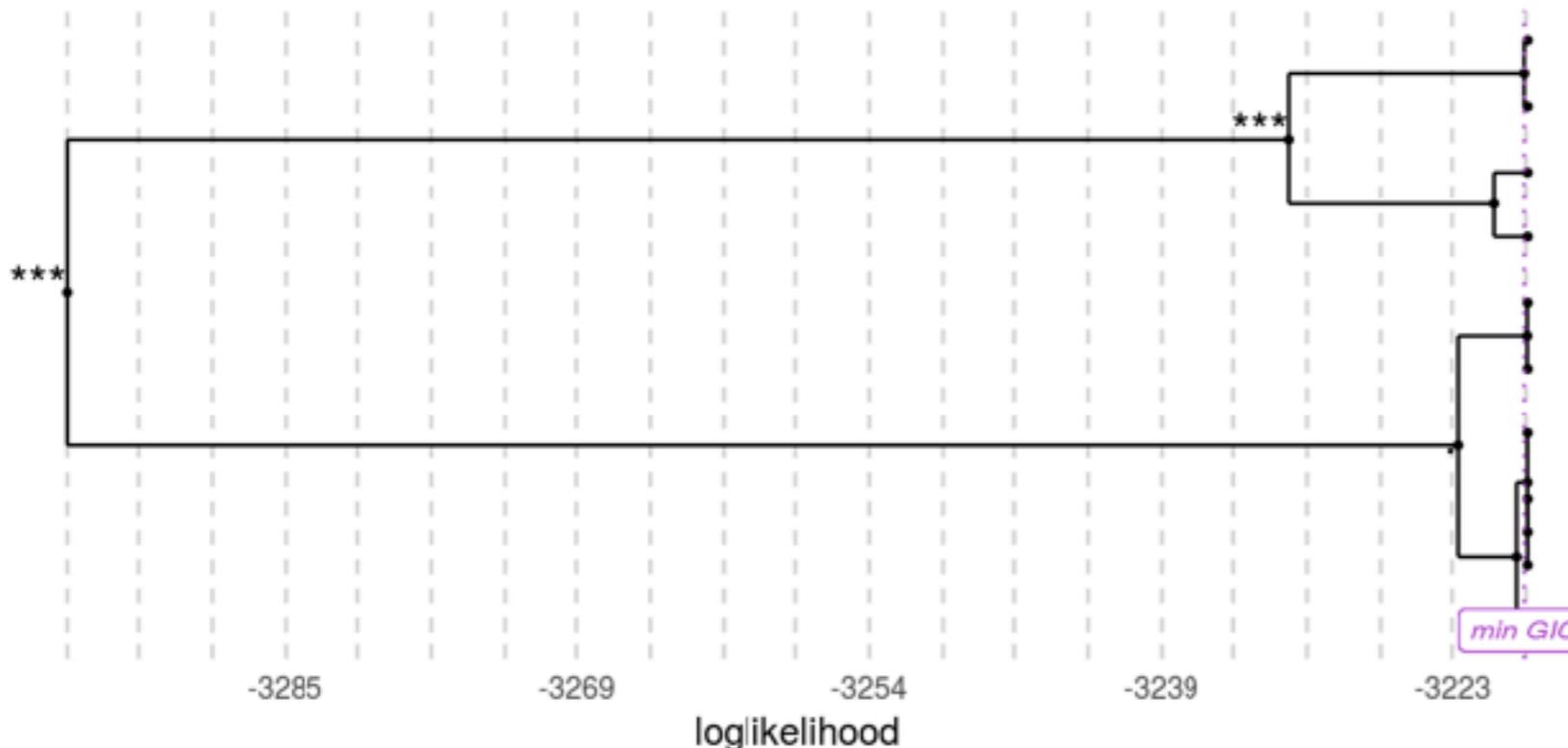
- The main function: `forest_model()`
- Works for many regression models (thanks to the broom package)
- Plot a single summary - forest models, rich in information about model parameters
- Produces ggplot2 plots

# Package factorMerger

## Visualisation for post-hoc testing

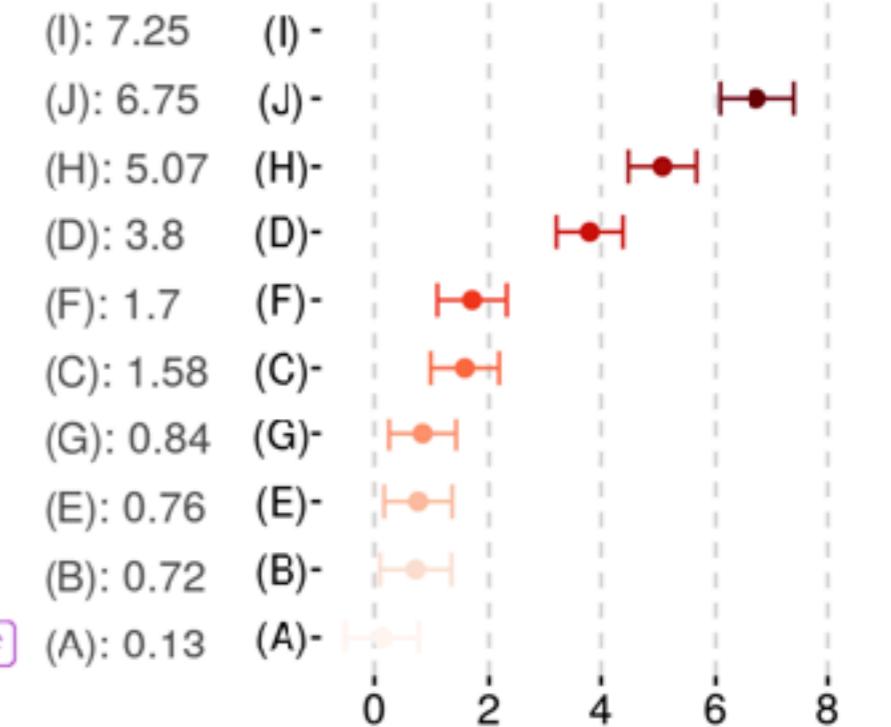
### Merging path plot

Optimal GIC partition: (A)(B)(E)(G):(C)(F):(D):(H):(J):(I)



### Summary statistics

Means and standard deviation

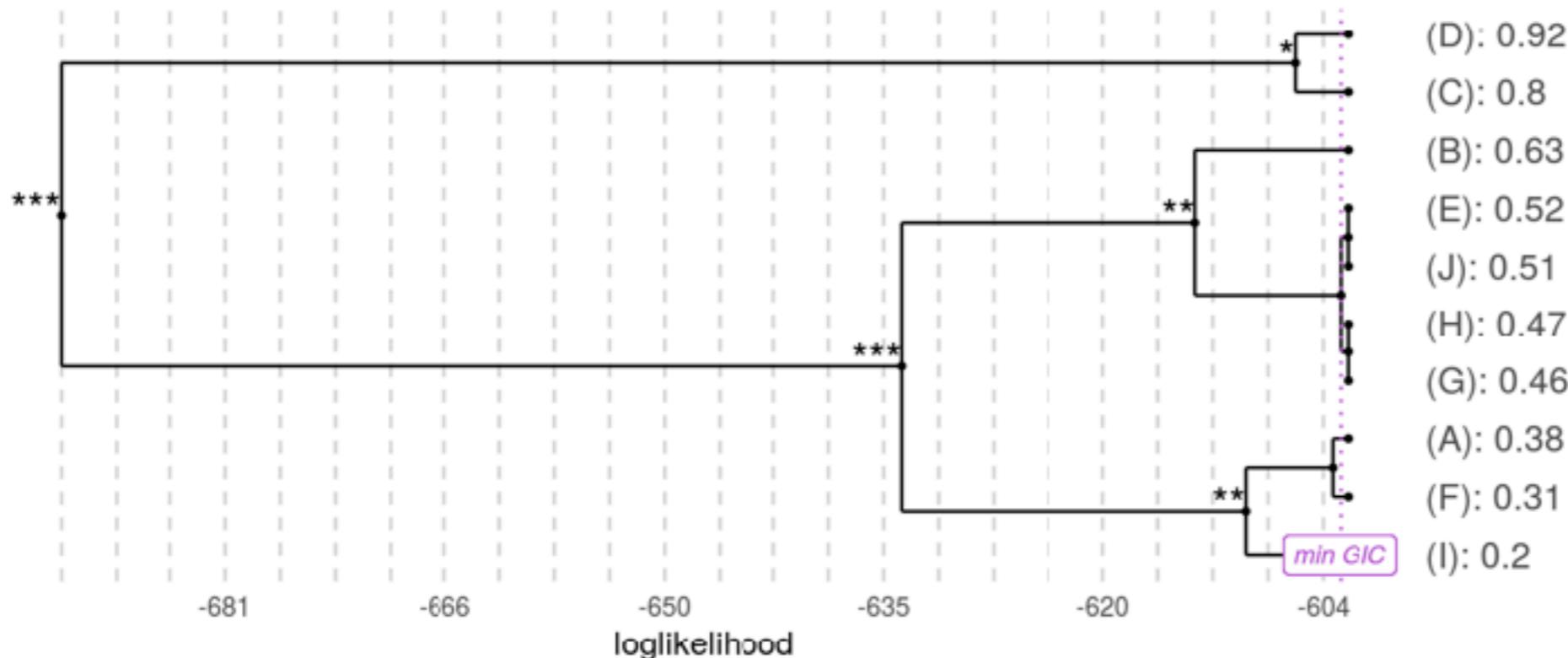


# Package factorMerger

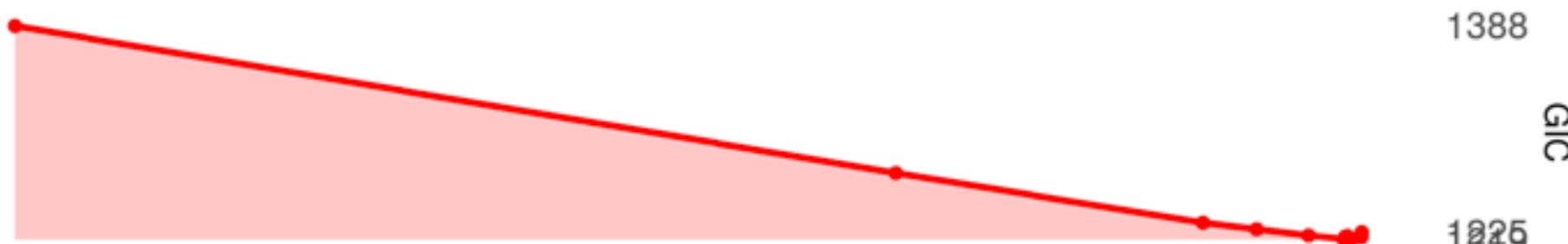
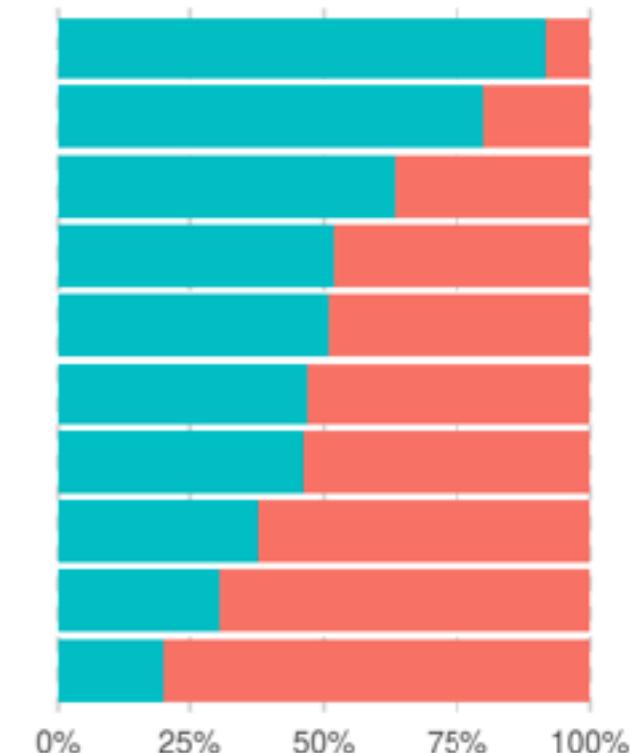
## Visualisation for post-hoc testing

### Merging path plot

Optimal GIC partition: (I):(F)(A):(G)(H)(J)(E):(B):(C):(D)



### Success ratio

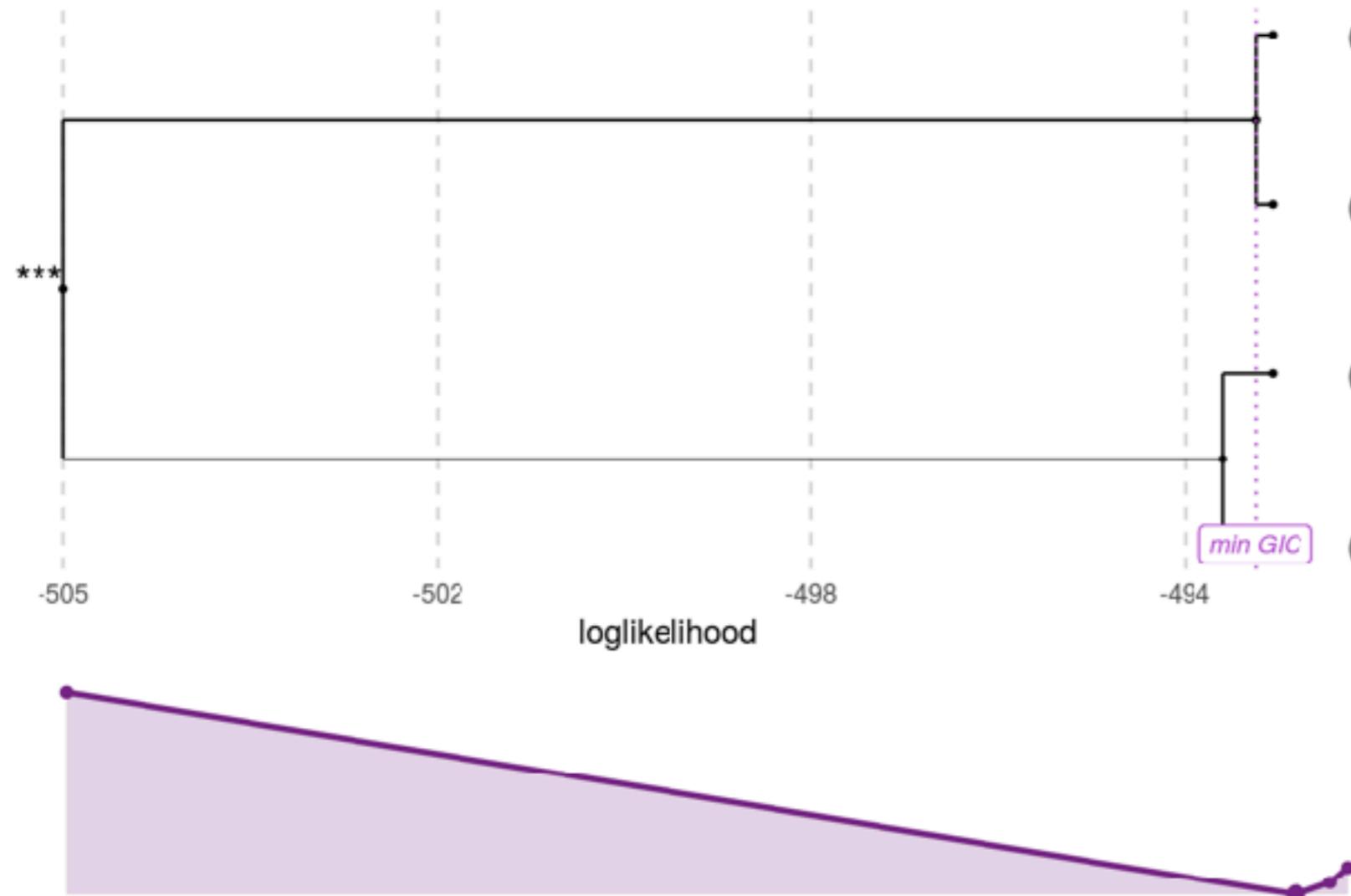


# Package factorMerger

## Visualisation for post-hoc testing

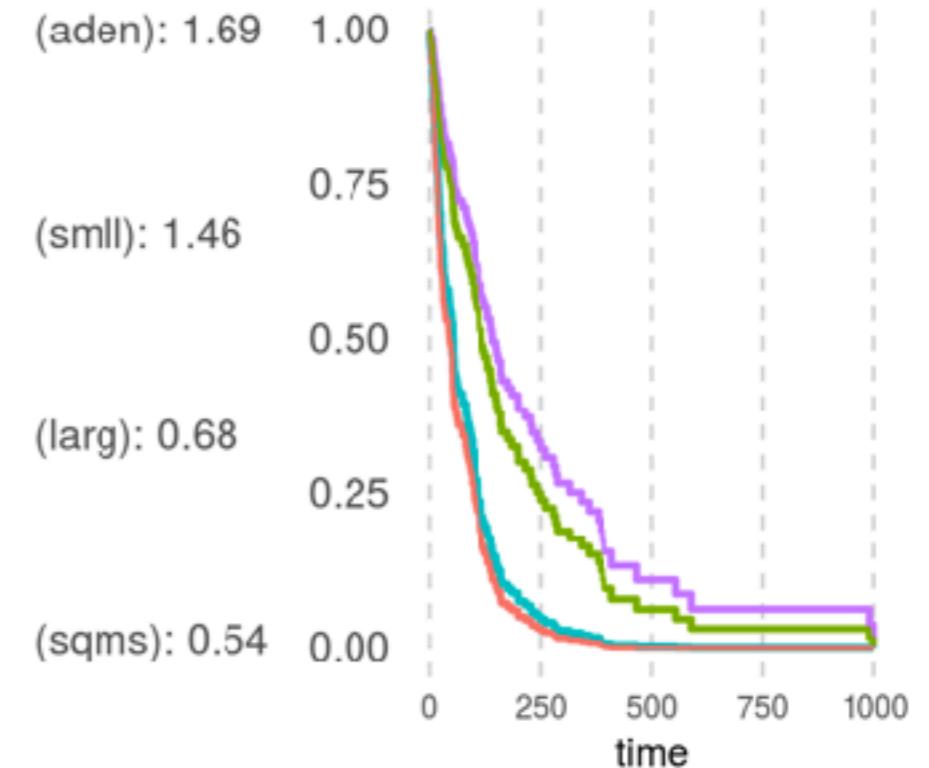
Merging path plot

Optimal GIC partition: (sqms)(larg):(smll)(aden)



Survival plot

Adjusted survival curves for

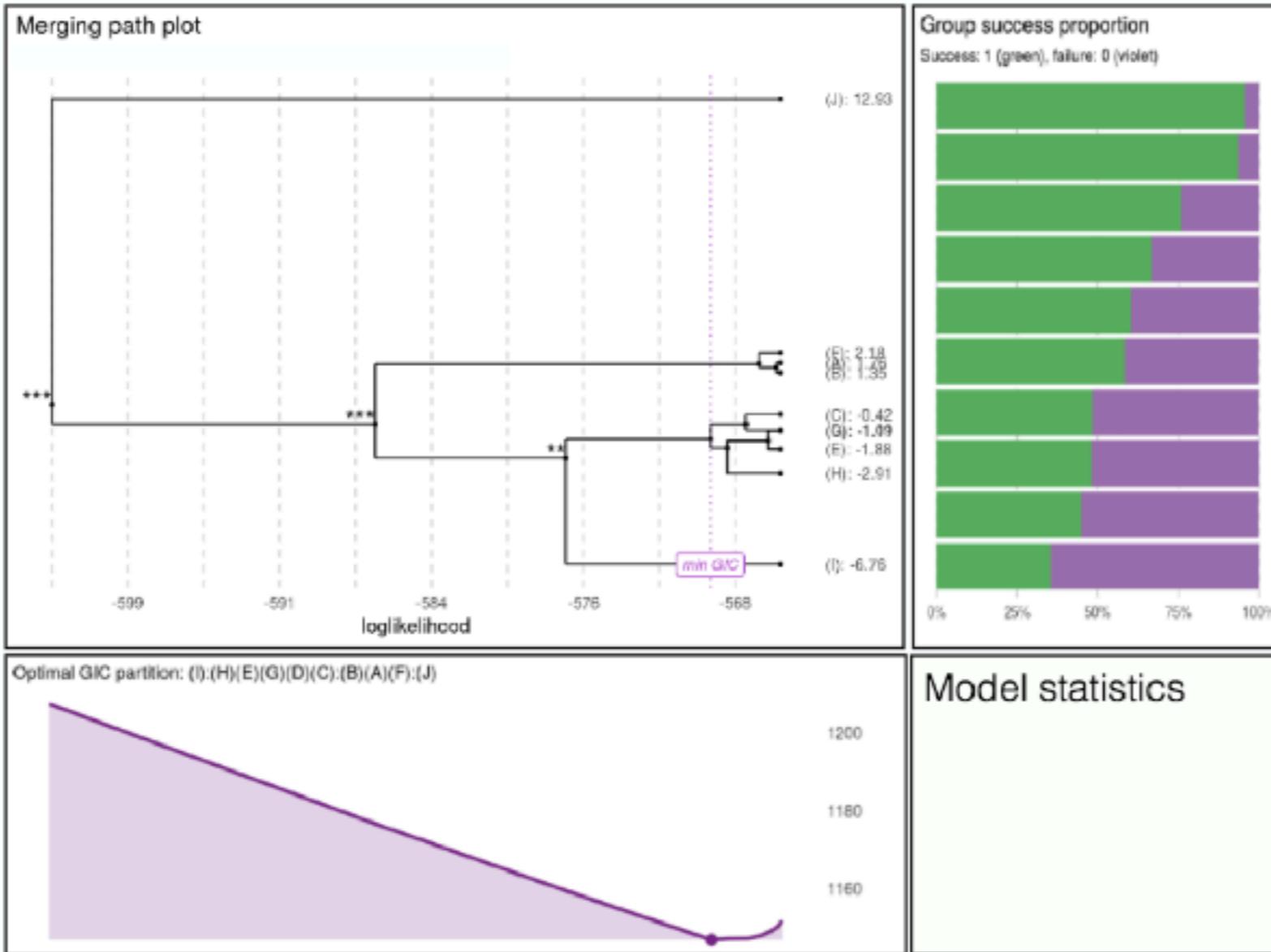


## Level fusing plot

The top-left plot shows level fusing paths (merging paths). With arguments **family=**, **show=**, **fuse=**, **spacing=**, one can select how to merge factors and what shall be presented on OX/OY axes.

Argument	Summary
panel = "all"	All panels
panel = "left"	Only left two panels
panel = "top"	Only top two panels
panel = "merging"	Merging path plot

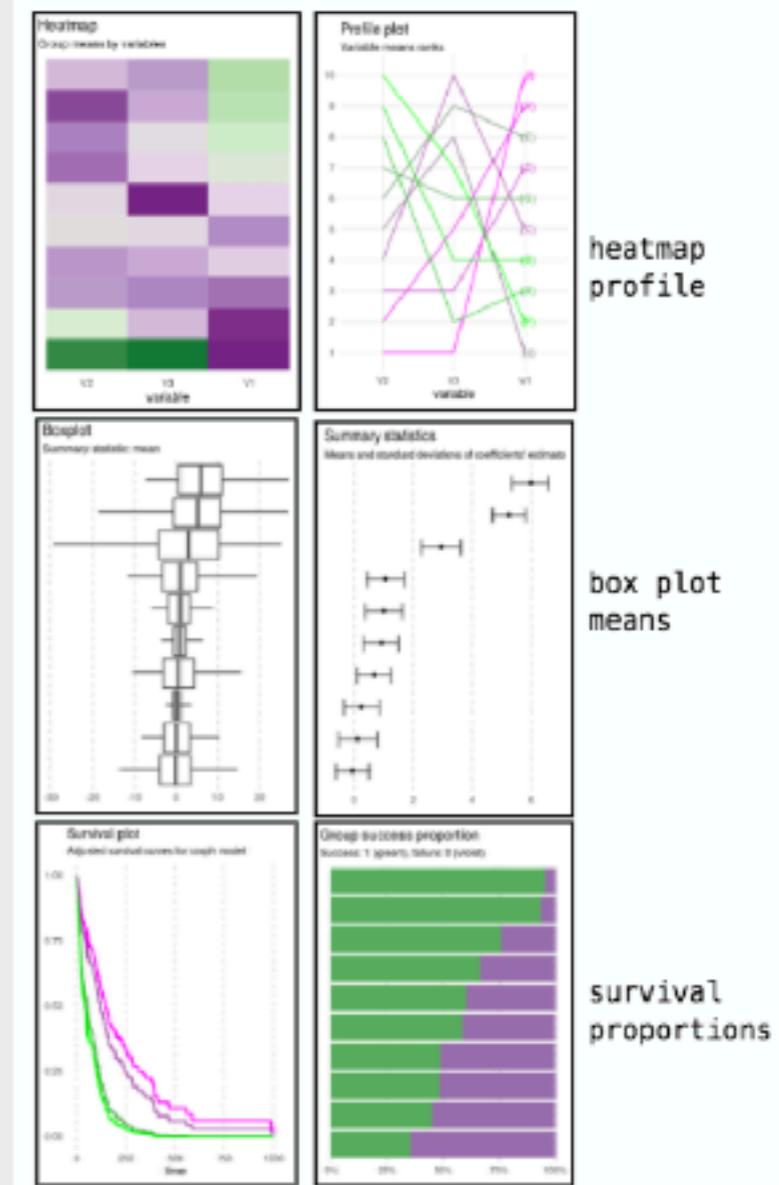
Argument	Summary
show = "likelihood"	Plot likelihood on OX axis
show = "p-value"	Plot p-values on OX axis
fuse = "allall"	Compare all pairs of groups
fuse = "nearby"	Compare nearby groups
fuse = "cluster"	DMR4glm algorithm
spacing = "equidistant"	Levels equidistant on OY scale
spacing = "effects"	Levels according to their effects
family = "gaussian"	For one-dimensional Gaussian
family = "mgaussian"	For multi dimensional Gaussian
family = "binomial"	For binomial regression
family = "survival"	For Cox regression



## Group summaries

The top-right panel shows group characteristics. Use the parameter **summary=** to select the most suitable presentation.

Argument	Summary
summary = "heatmap"	For m gaussian
summary = "profile"	For m gaussian
summary = "boxplot"	For gaussian
summary = "means"	For gaussian
summary = "survival"	For Cox regression
summary = "proportions"	For binomial regression



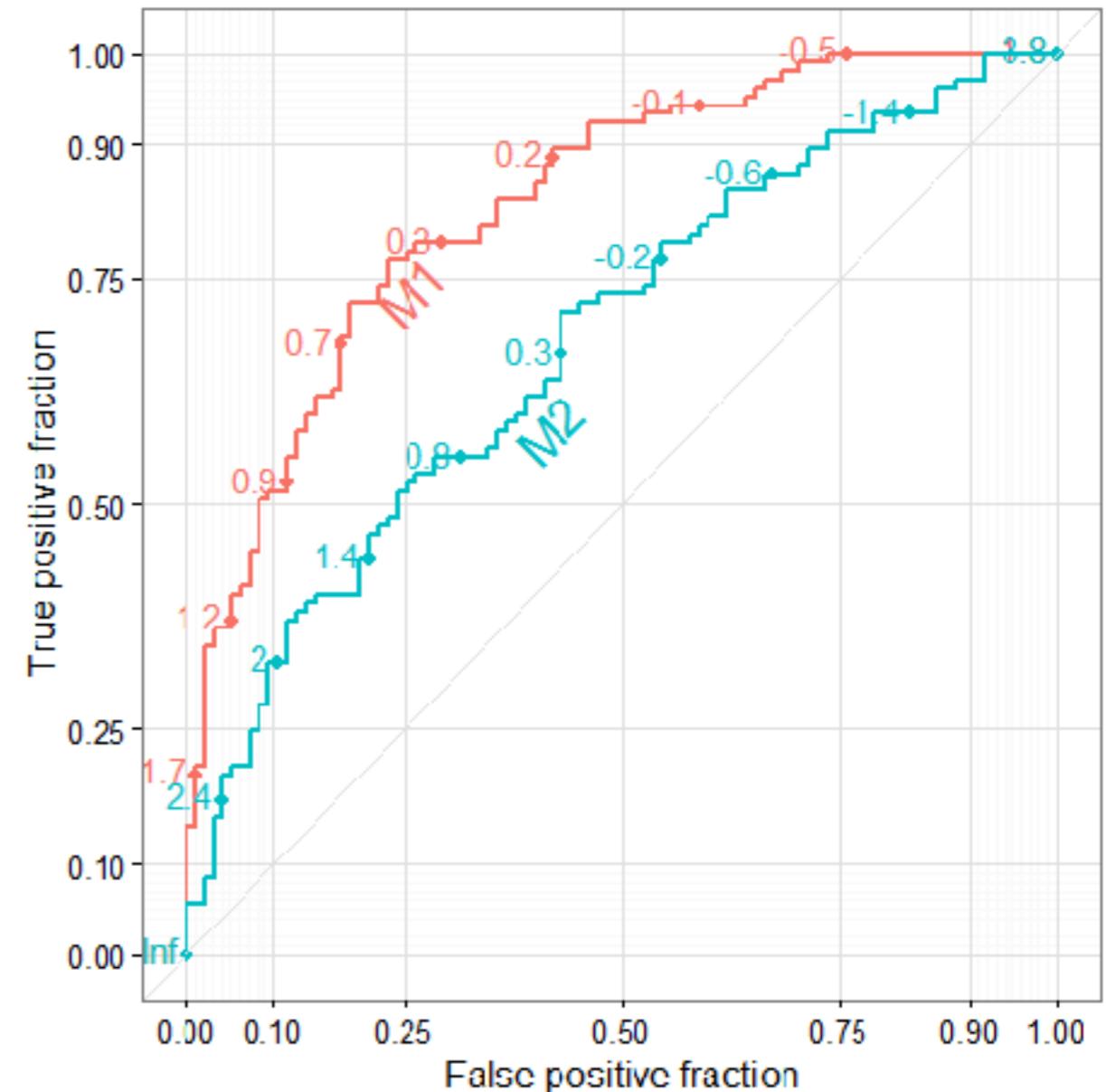
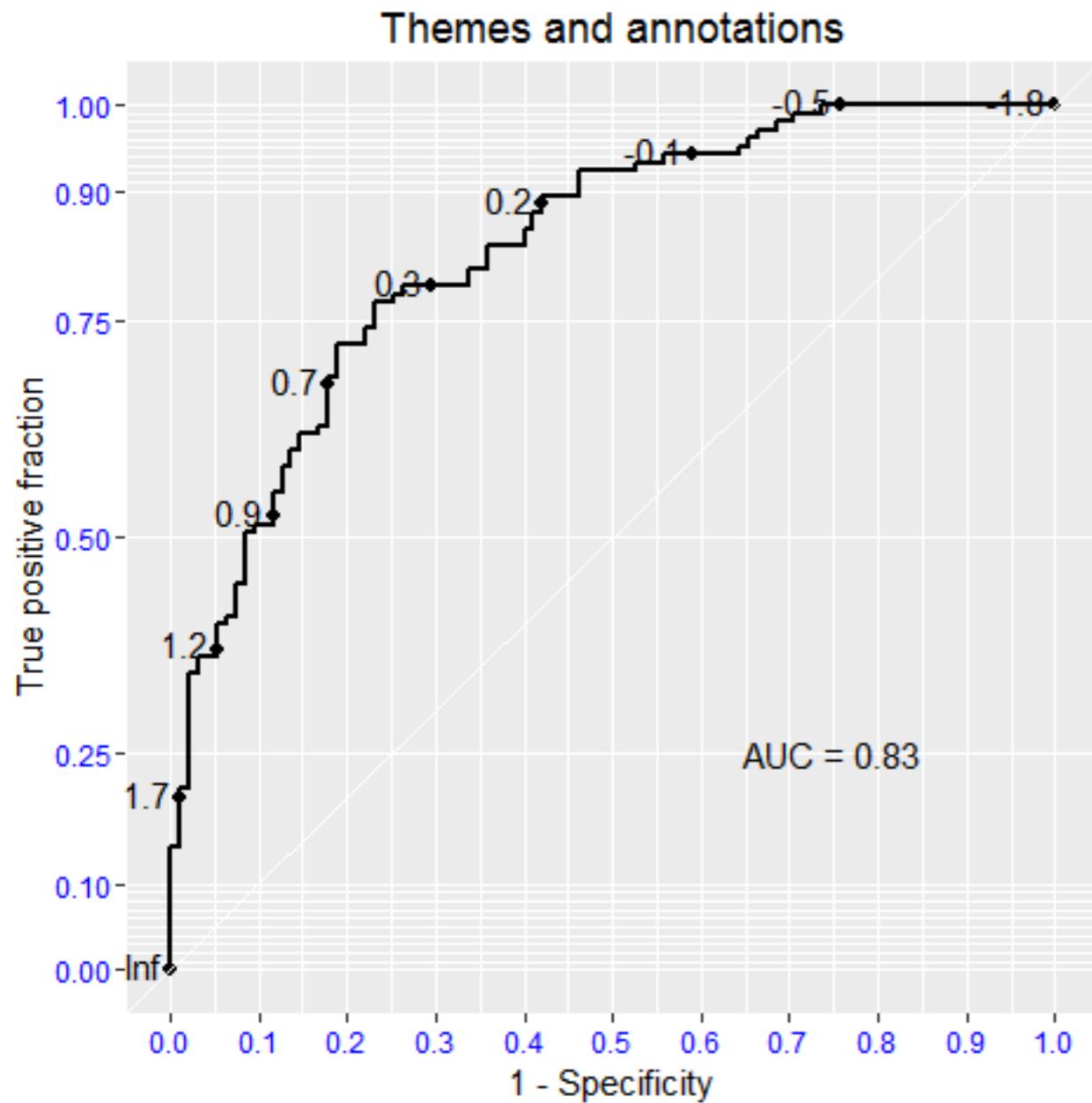
# Package factorMerger

- The main function: mergeFactors()
- Works for likelihood-based regression models
- Plot a single summary - merging paths plots, rich in information about group's similarities
- Produces ggplot2 plots

Agnieszka Sitko, Przemysław Biecek (2017)  
<https://github.com/geneticsMiNIng/FactorMerger>

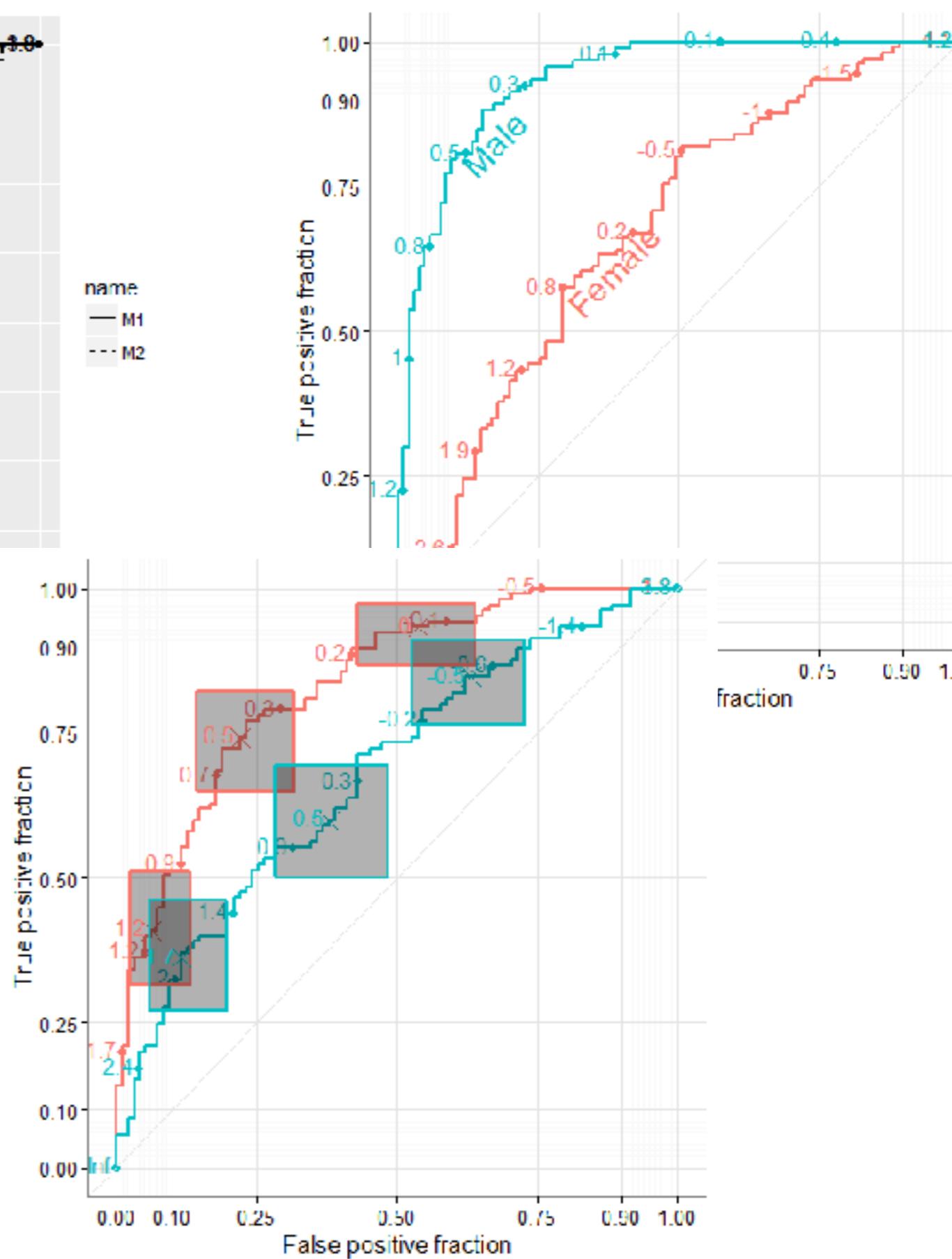
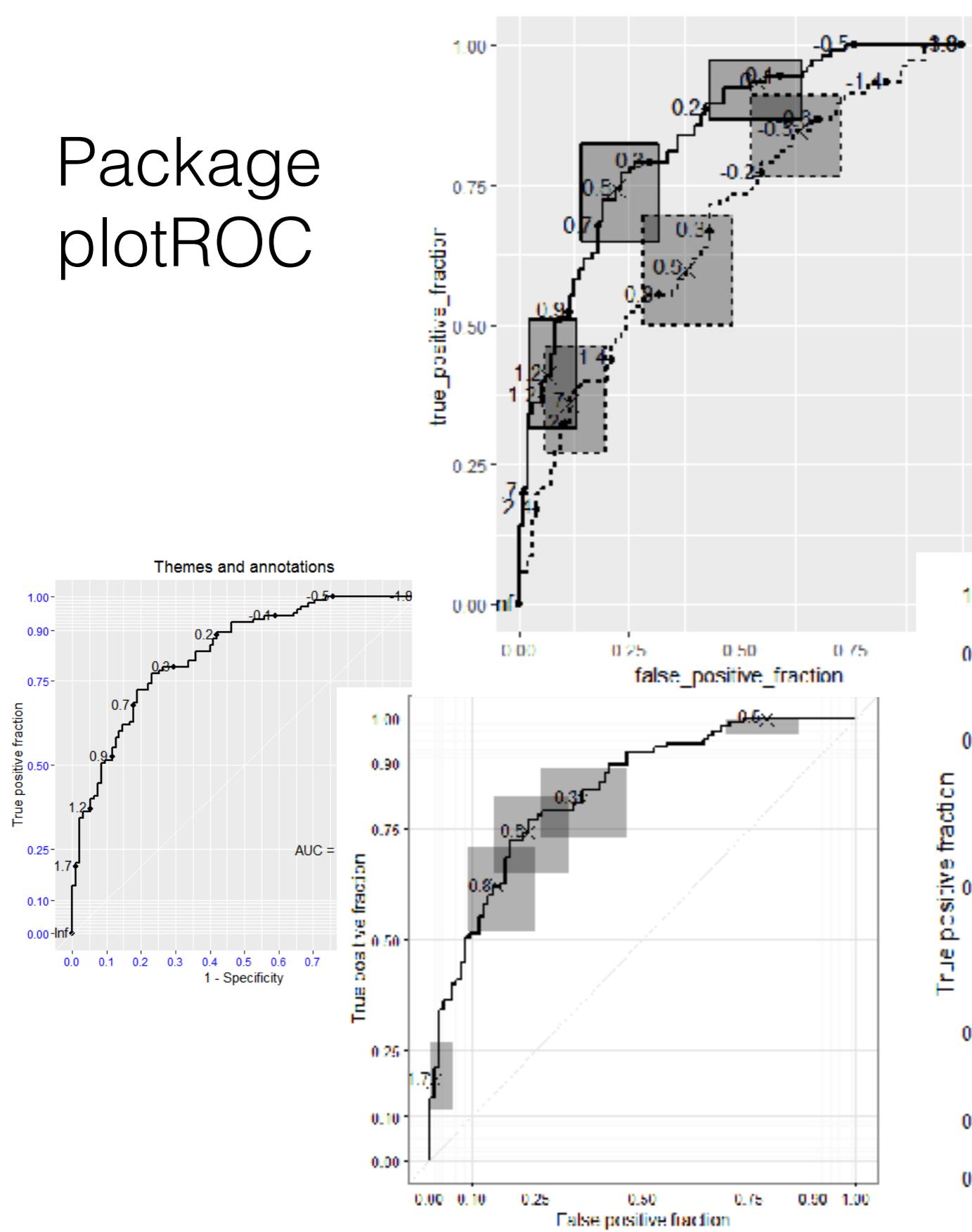
# Validation

# Package plotROC



Michael Sachs (2016) <http://sachsma.github.io/plotROC>

# Package plotROC



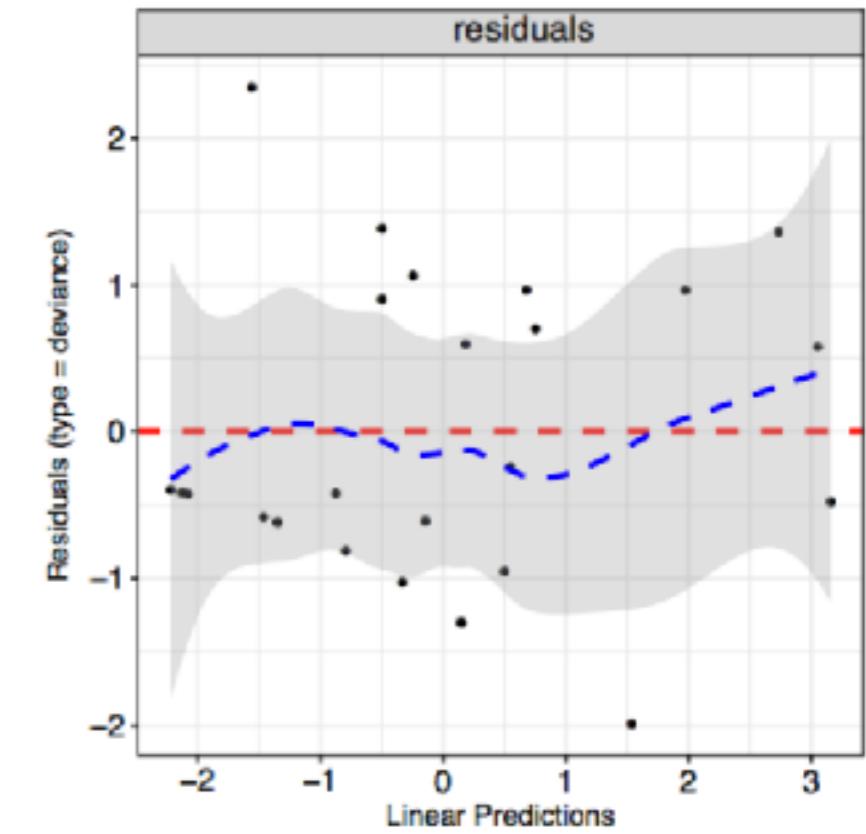
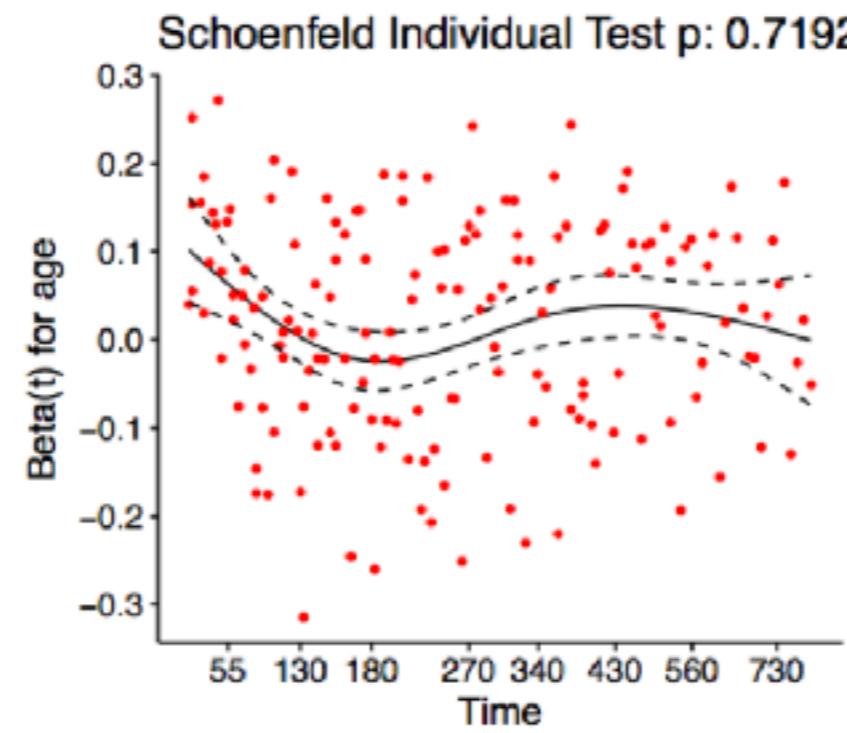
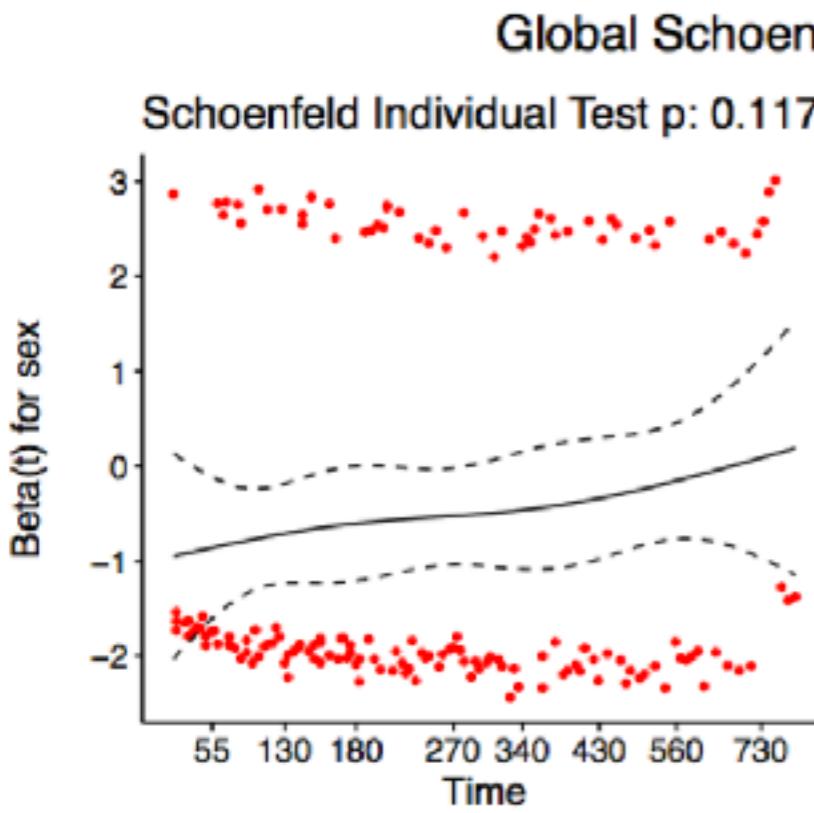
Michael Sachs (2016) <http://sachs-mc.github.io/plotROC>

# Package plotROC

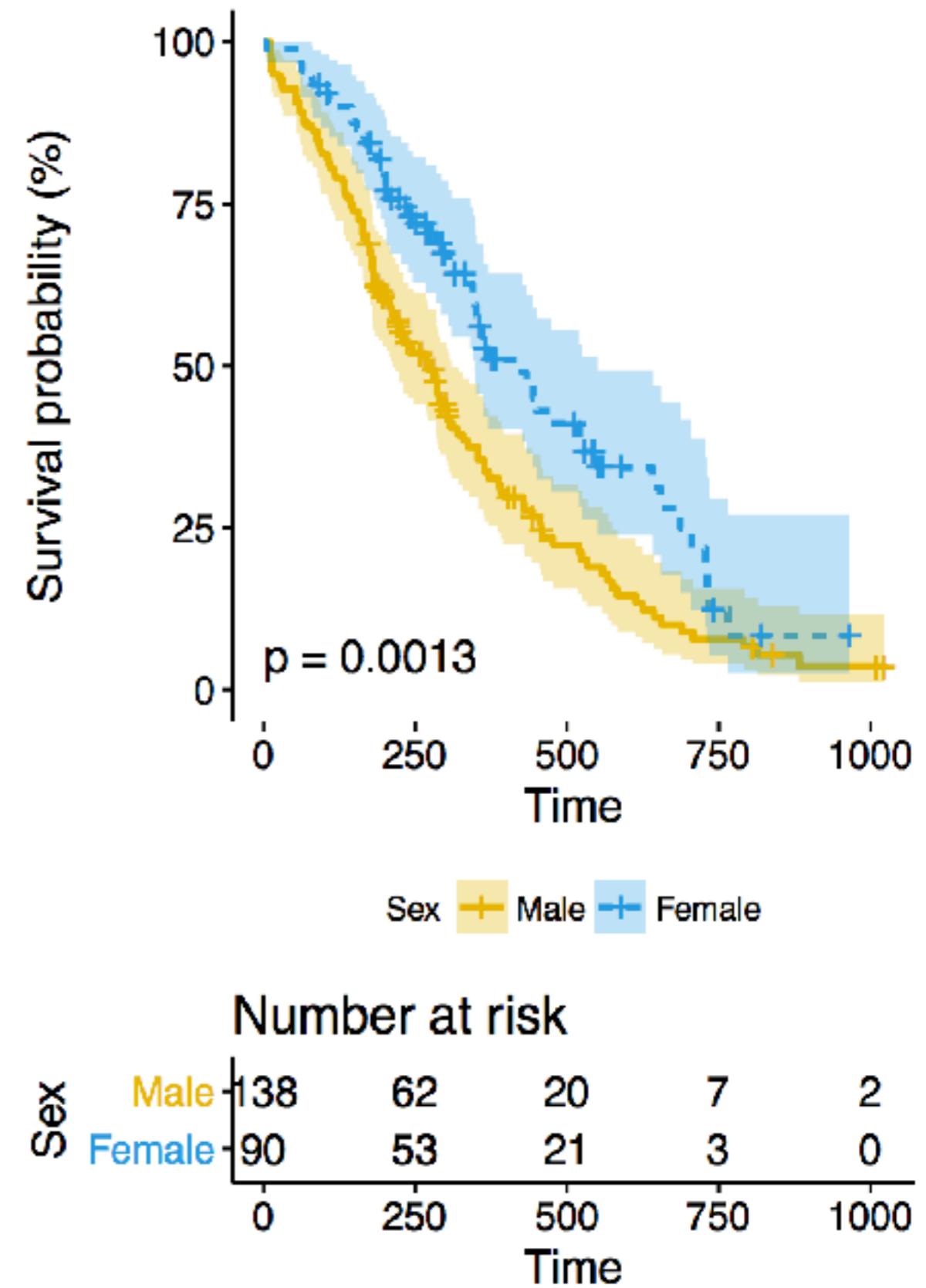
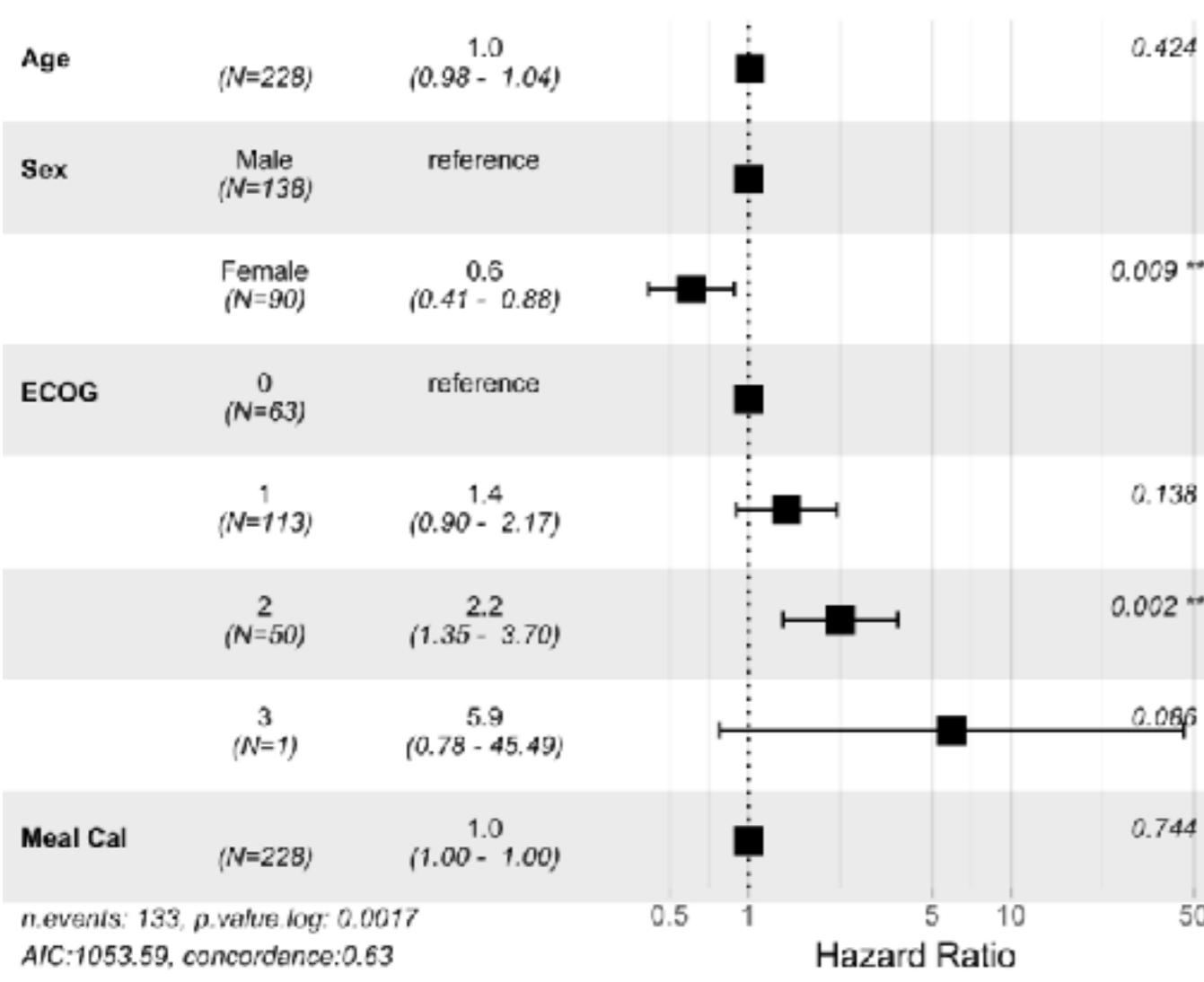
- The main function: `geom_roc()`
- Works for multiple models
- For each model plots a ROC curve with confidence intervals (with `geom_rocci()`)
- Produces ggplot2 plots

# Package survminer

Diagnostic plots for various residuals: "martingale", "deviance", "score", "schoenfeld", "dfbeta", "dfbetas" and "scaledsch"



# Package survminer



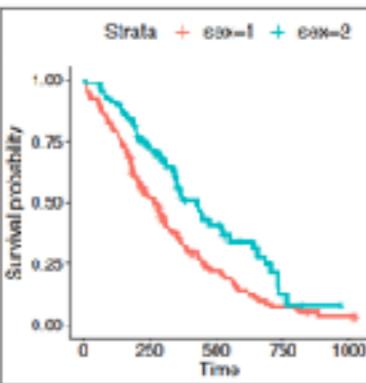
# Creating Survival Plots

## Informative and Elegant with **survminer**

### Survival Curves

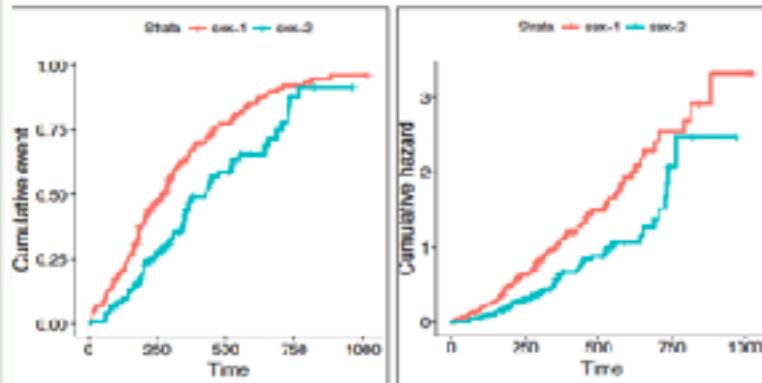
The **ggsurvplot()** function creates **ggplot2** plots from **survfit** objects.

```
library("survival")
fit <- survfit(Surv(time, status) ~ sex, data = lung)
class(fit)
## [1] "survfit"
library("survminer")
ggsurvplot(fit, data = lung)
```



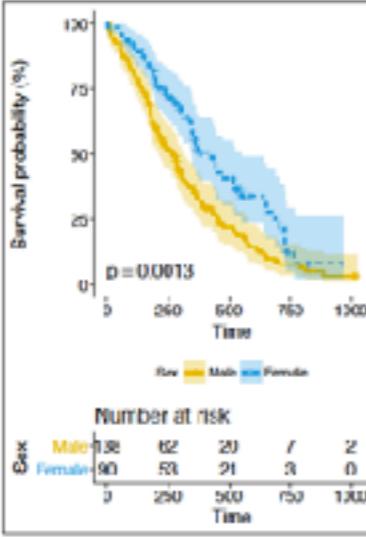
Use the **fun** argument to set the transformation of the survival curve. E.g. "**event**" for cumulative events, "**cumhaz**" for the cumulative hazard function or "**pct**" for survival probability in percentage.

```
ggsurvplot(fit, data = lung, fun = "event")
ggsurvplot(fit, data = lung, fun = "cumhaz")
```



With lots of graphical parameters you have full control over look and feel of the survival plots; position and content of the legend; additional annotations like p-value, title, subtitle.

```
ggsurvplot(fit, data = lung,
conf.int = TRUE,
pval = TRUE,
fun = "pct",
risk.table = TRUE,
size = 1,
linetype = "strata",
palette = c("#E78880",
           "#2E9DFC"),
legend = "bottom",
legend.title = "Sex",
legend.labs = c("Male",
              "Female"))
```



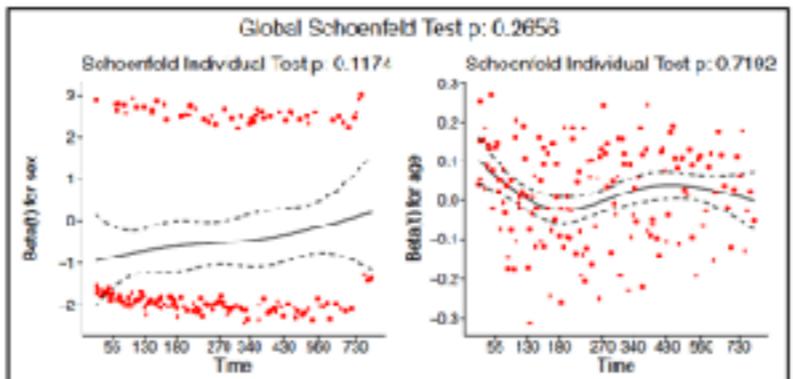
### Diagnostics of Cox Model

The function **cox.zph()** from **survival** package may be used to test the proportional hazards assumption for a Cox regression model fit. The graphical verification of this assumption may be performed with the function **ggscoxph()** from the **survminer** package. For each covariate it produces plots with scaled Schoenfeld residuals against the time.

```
library("survival")
fit <- coxph(Surv(time, status) ~ sex + age, data = lung)
ftest <- cox.zph(fit)
ftest
```

	chisq	p
## sex	0.1236	2.452 0.117
## age	-0.0275	0.129 0.719
## GLOBAL	NA	2.651 0.266

```
library("survminer")
ggscoxph(ftest)
```



The function **ggcodiagnostics()** plots different types of residuals as a function of time, linear predictor or observation id. The type of residual is selected with **type** argument. Possible values are "martingale", "deviance", "score", "schoenfeld", "dfbeta", "dfbetas", and "scaledsch".

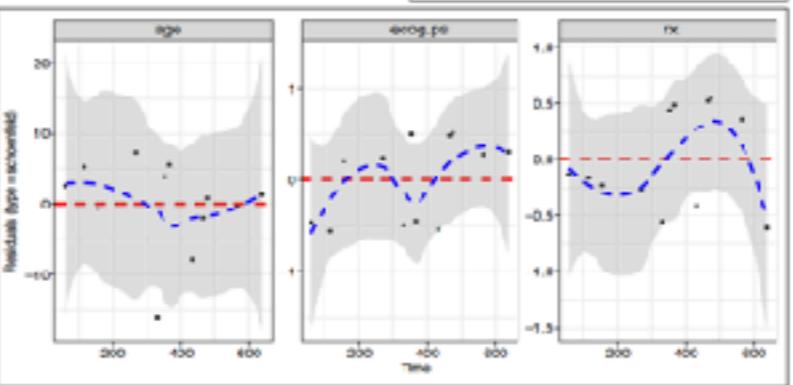
The **ox.scale** argument defines what shall be plotted on the OX axis. Possible values are "linear.predictions", "observation.id", "time".

Logical arguments **hline** and **sline** may be used to add horizontal line or smooth line to the plot.

```
library("survival")
library("survminer")
fit <- coxph(Surv(time, status) ~ sex + age, data = lung)
```

```
ggcodiagnostics(fit,
type = "deviance",
ox.scale = "linear.predictions")
```

```
ggcodiagnostics(fit,
type = "schoenfeld",
ox.scale = "time")
```



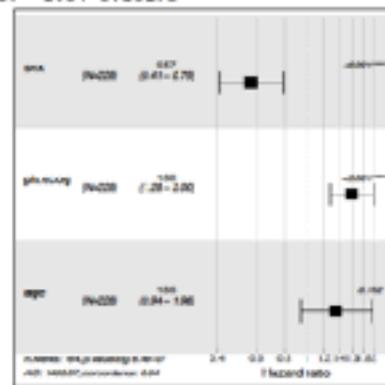
### Summary of Cox Model

The function **ggsforest()** from the **survminer** package creates a forest plot for a Cox regression model fit. Hazard ratio estimates along with confidence intervals and p values are plotted for each variable.

```
library("survival")
library("survminer")
lung$age <- ifelse(lung$age > 70, ">70", "<= 70")
fit <- coxph(Surv(time, status) ~ sex + ph.ecog + age, data = lung)
fit
```

```
## Call:
## coxph(formula = Surv(time, status) ~ sex+ph.ecog+age, data=lung)
##
##          coef exp(coef) se(coef)    z      p
## sex     -0.567   0.567   0.158  -3.37 0.00075
## ph.ecog  0.470   1.660   0.113   4.16 3.1e-05
## age>70  0.387   1.359   0.157   1.64 0.1015
##
## Likelihood ratio test=31.6  on
## n= 227, number of events= 164
```

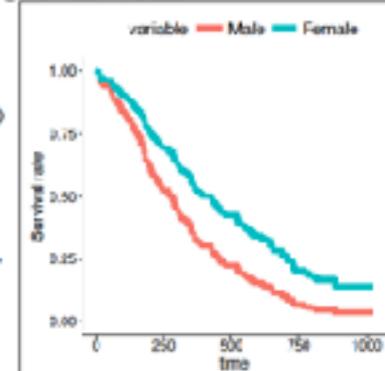
```
ggsforest(fit)
```



The function **ggcoxadjustedcurves()** from the **survminer** package plots Adjusted Survival Curves for Cox Proportional Hazards Model. Adjusted Survival Curves show how a selected factor influences survival estimated from a Cox model.

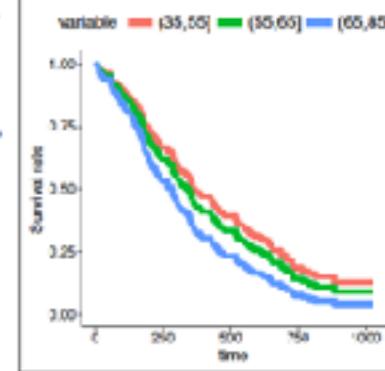
Note that these curves differ from Kaplan Meier estimates since they present expected survival based on given Cox model.

```
library("survival")
library("survminer")
lung$sex <- ifelse(lung$sex == 1,
                   "Male", "Female")
fit <- coxph(Surv(time, status) ~ sex + ph.ecog + age,
            data = lung)
ggcoxadjustedcurves(fit, data=lung,
variable=lung$sex)
```



Note that it is not necessary to include the grouping factor in the Cox model. Survival curves are estimated from Cox model for each group defined by the factor independently.

```
lung$age3 <- cut(lung$age,
                  c(35,55,65,85))
ggcoxadjustedcurves(fit, data=lung,
variable=lung$age3)
```



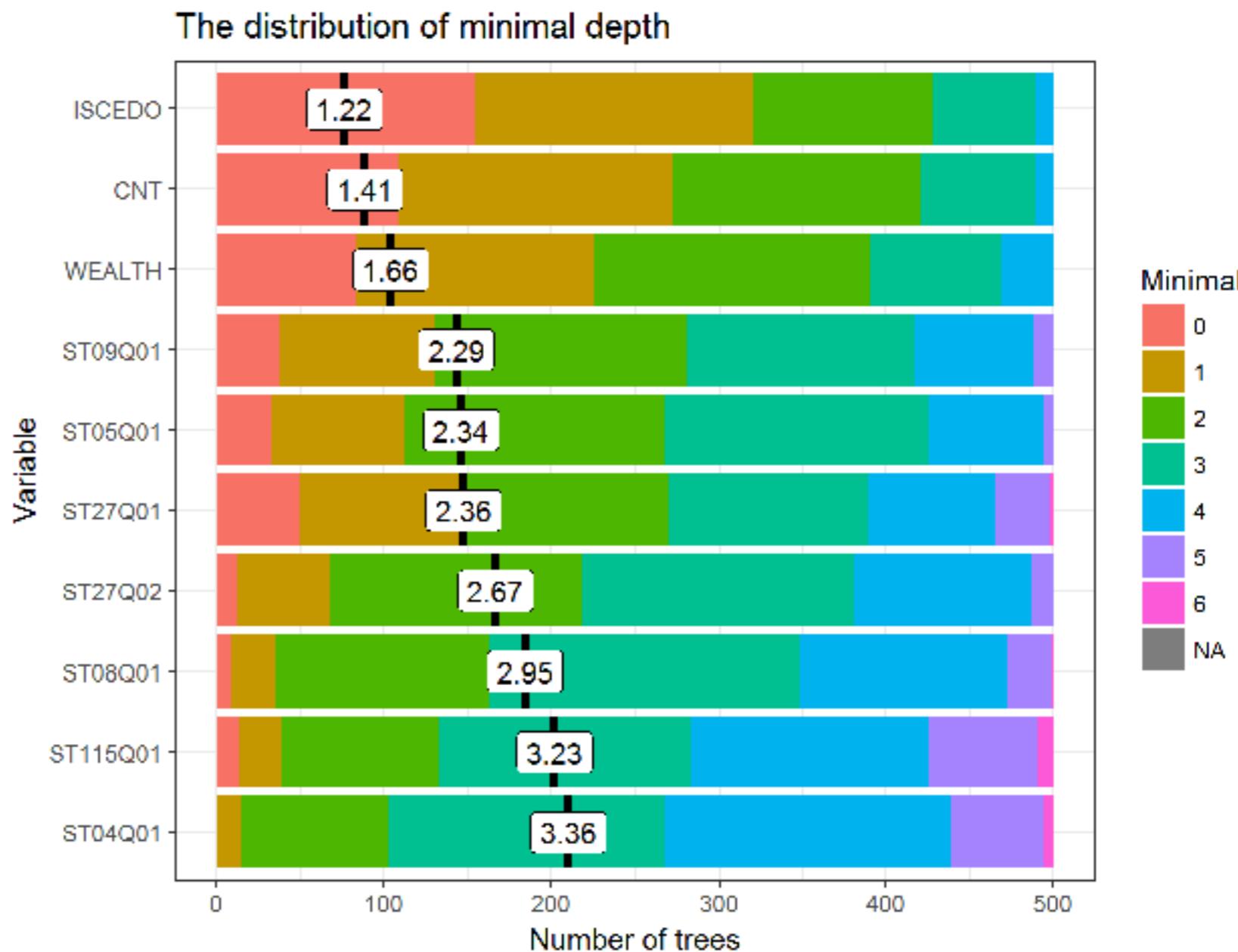
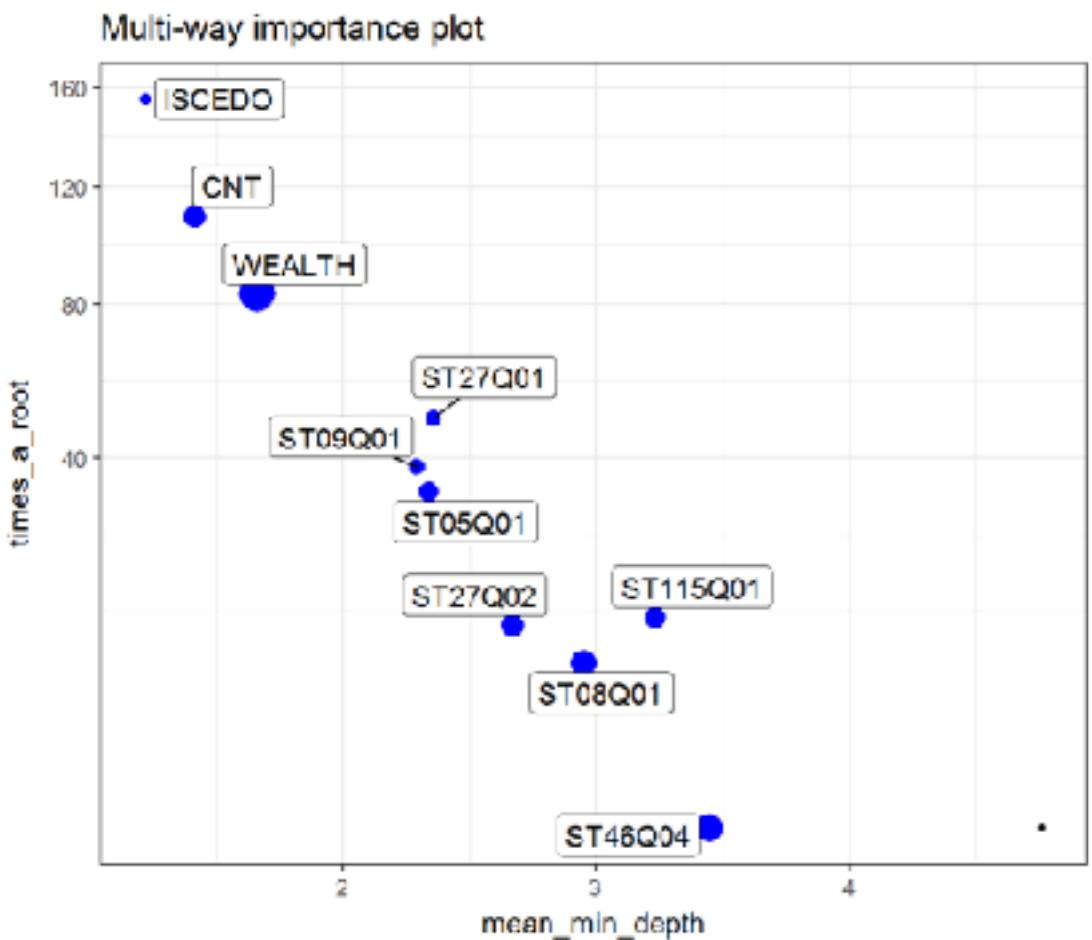
# Package survminer

- Set of functions with many arguments
- Works for selected survival models (e.g. CoxPH, survfit, cmprisk)
- Plots many (all popular) graphical summaries for given models
- Produces ggplot2 plots

# Prediction

# Package randomForestExplainer

## What is in a random forest?



Ola Paluszynska, Przemyslaw Biecek (2017)

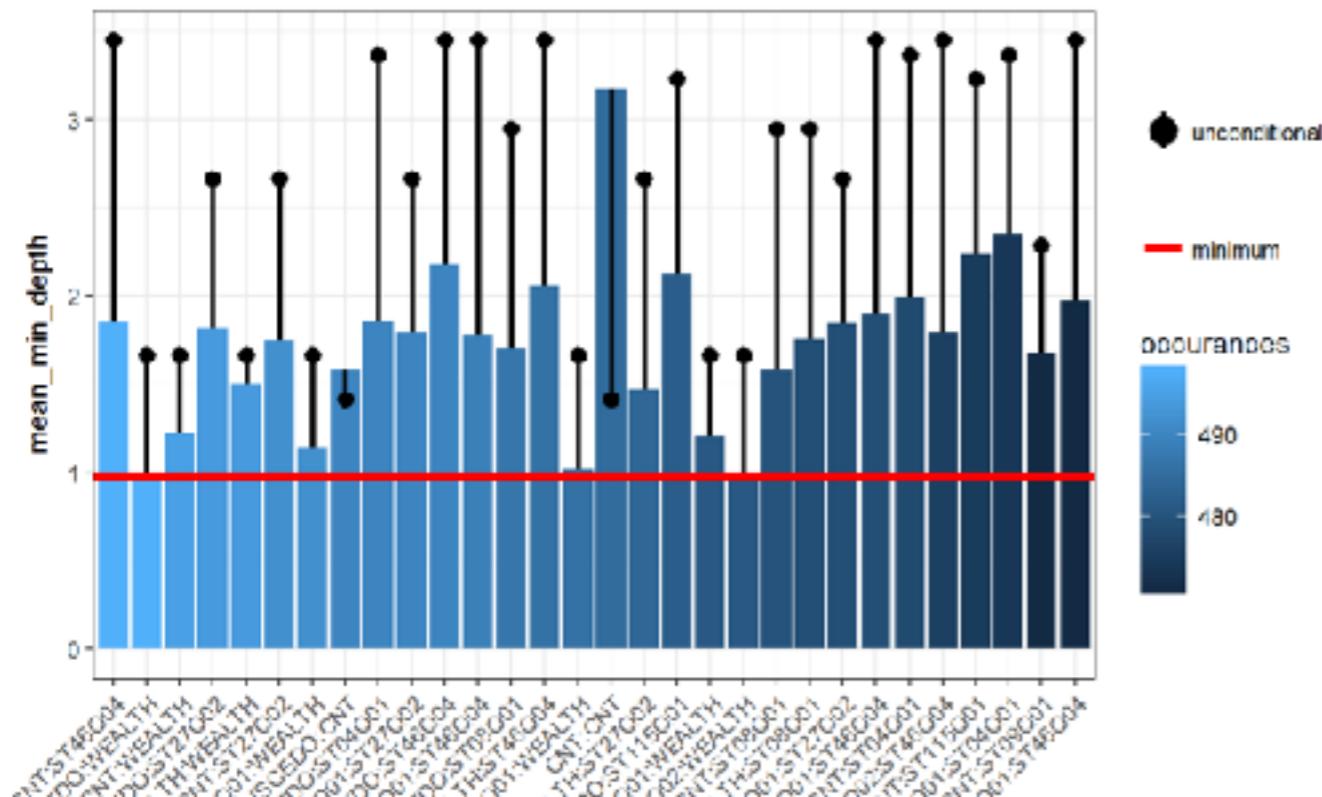
<https://github.com/geneticsMiNIng/BlackBoxOpener>

John Ehrlinger (2015) ggRandomForests: Random Forests for Regression

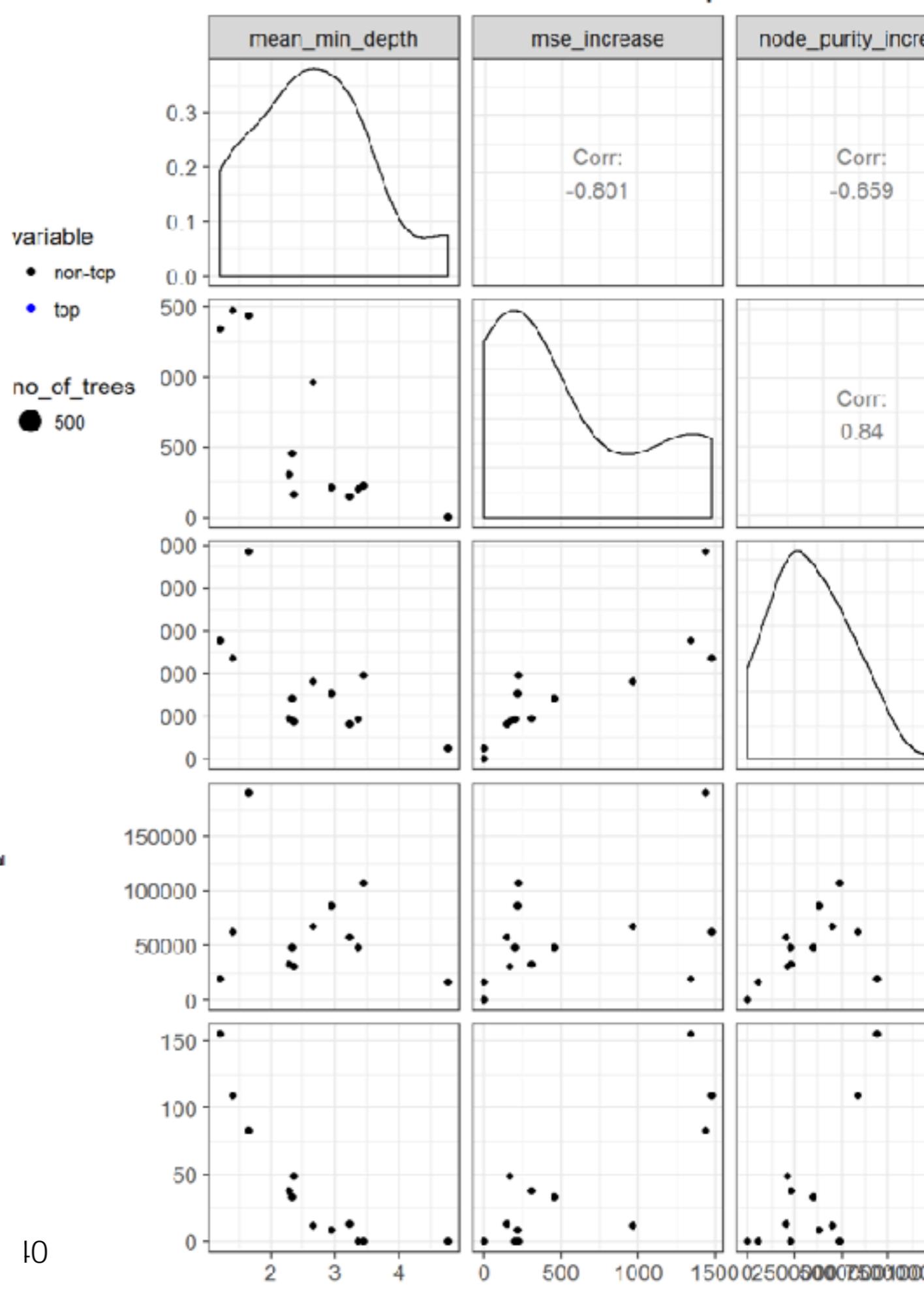
## Multi-way importance plot



Mean minimal depth for 30 most frequent interactions



## Relations between measures of importance



# Package randomForestExplainer

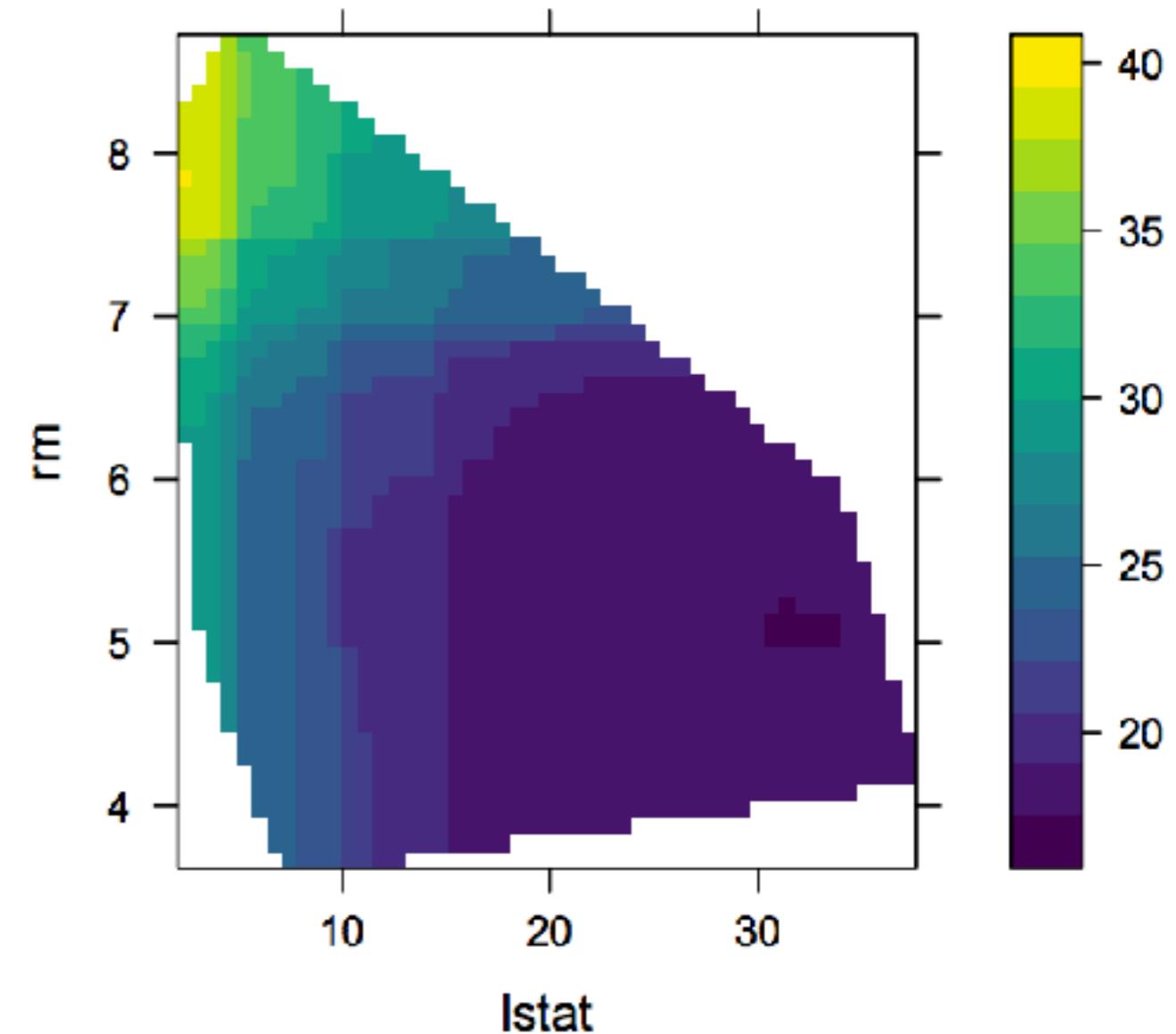
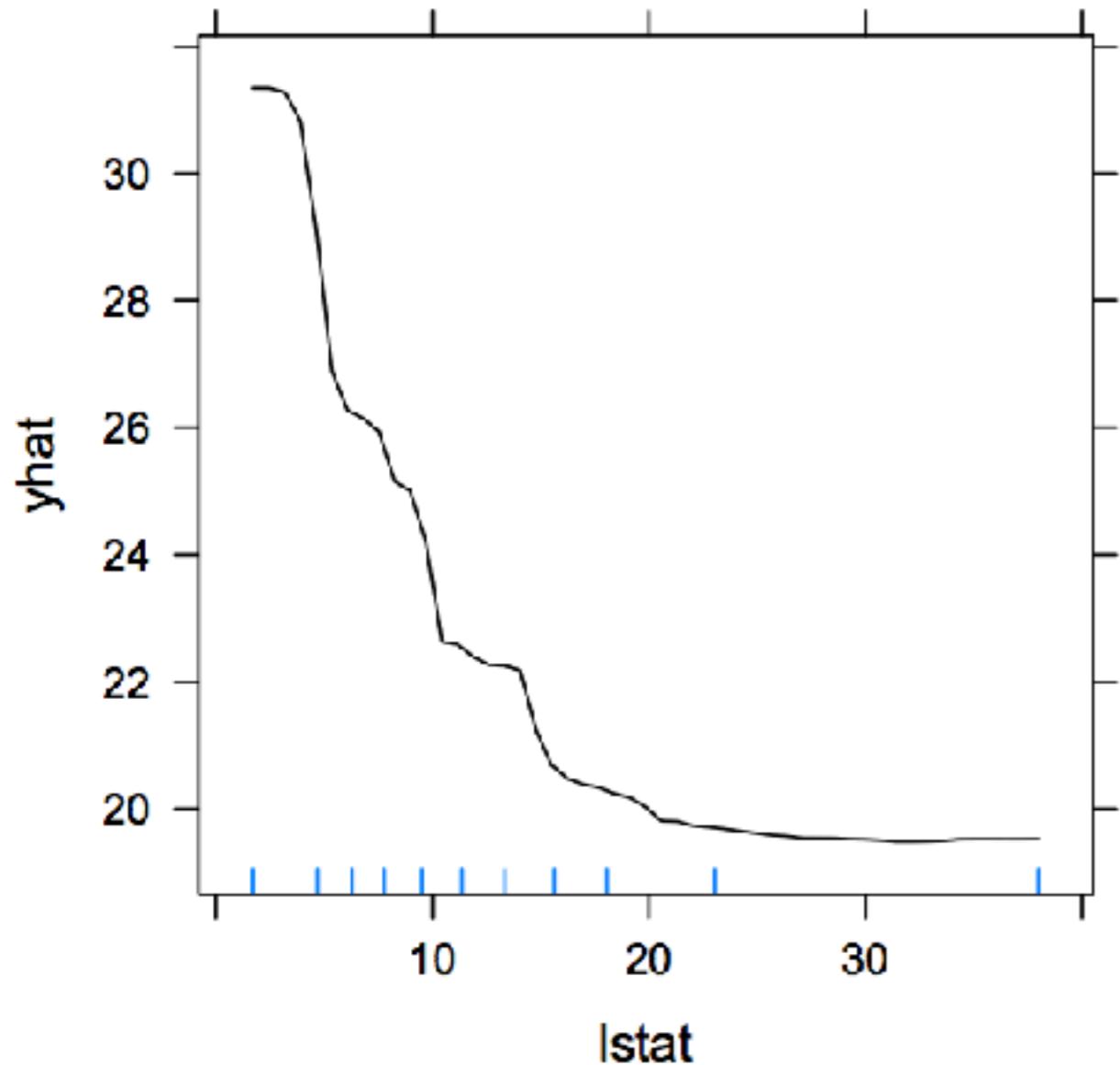
- The main function: `summarize_forest()`
- Works for `randomForest` objects
- Calculates and plots various measures of variable importance
- Produces `ggplot2` plots

Ola Paluszynska, Przemyslaw Biecek (2017)

<https://github.com/geneticsMiNIng/BlackBoxOpener>

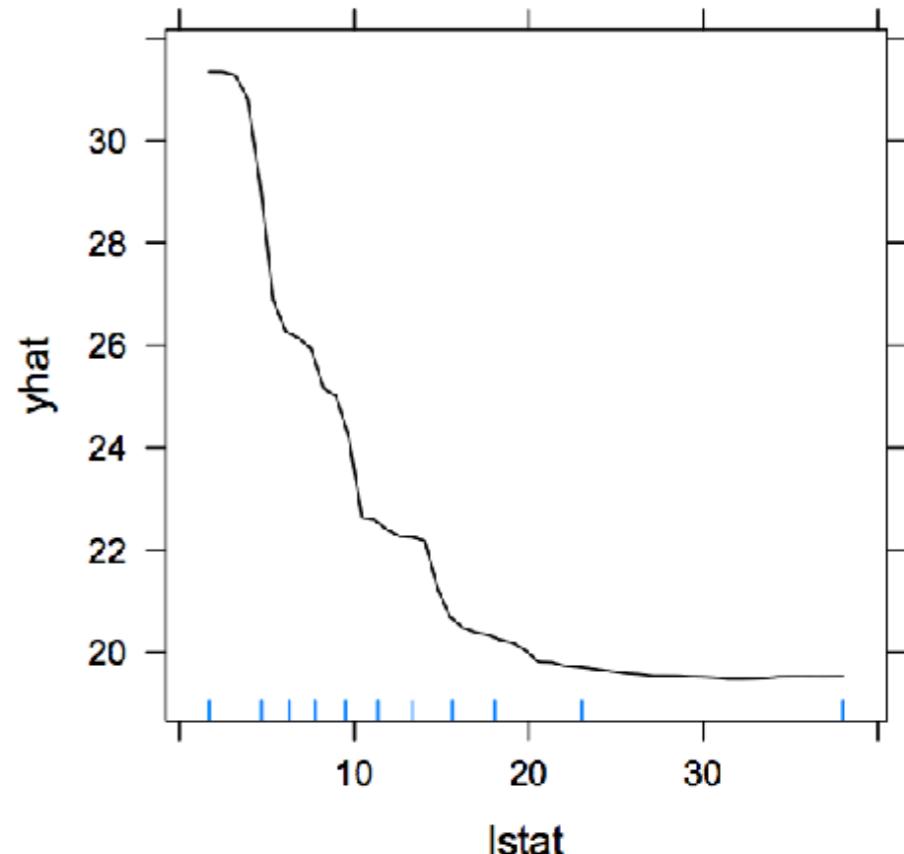
John Ehrlinger (2015) `ggRandomForests`: Random Forests for Regression

# Package pdp - Partial Dependence Plots



# Package pdp - Partial Dependence Plots

Type of model	R package	Object class
Decision tree	<b>C50</b> (Kuhn et al., 2015)	"C5.0"
	<b>party</b>	"BinaryTree"
	<b>partykit</b>	"party"
	<b>rpart</b> (Therneau et al., 2015)	"rpart"
Bagged decision trees	<b>adabag</b> (Alfaro et al., 2013)	"bagging"
	<b>ipred</b> (Peters and Hothorn, 2015)	"classbagg", "regbagg"
		"boosting"
Boosted decision trees	<b>adabag</b> (Alfaro et al., 2013)	"gbm"
	<b>gbm</b>	"xgb.Booster"
	<b>xgboost</b>	"cubist"
Cubist	<b>Cubist</b> (Kuhn et al., 2014)	"lda", "qda"
Discriminant analysis	<b>MASS</b> (Venables and Ripley, 2002)	"glm", "lm"
Generalized linear model	<b>stats</b>	"lm"
Linear model	<b>stats</b>	"nls"
Nonlinear least squares	<b>stats</b>	"earth"
Multivariate adaptive regression splines (MARS)	<b>earth</b> (Milborrow, 2016)	"mars"
	<b>mda</b> (Leisch et al., 2016)	"ppr"
Projection pursuit regression	<b>stats</b>	
Random forest	<b>randomForest</b>	"randomForest"
	<b>party</b>	"RandomForest"
	<b>partykit</b>	"cforest"
	<b>ranger</b> (Wright, 2016)	"ranger"
Support vector machine	<b>e1071</b> (Meyer et al., 2015)	"svm"
	<b>kernlab</b> (Karatzoglou)	



pdp: An R Package for Constructing Partial Dependence Plots

Brandon M. Greenwell (2017)

<https://journal.r-project.org/archive/2017/RJ-2017-016/index.html>

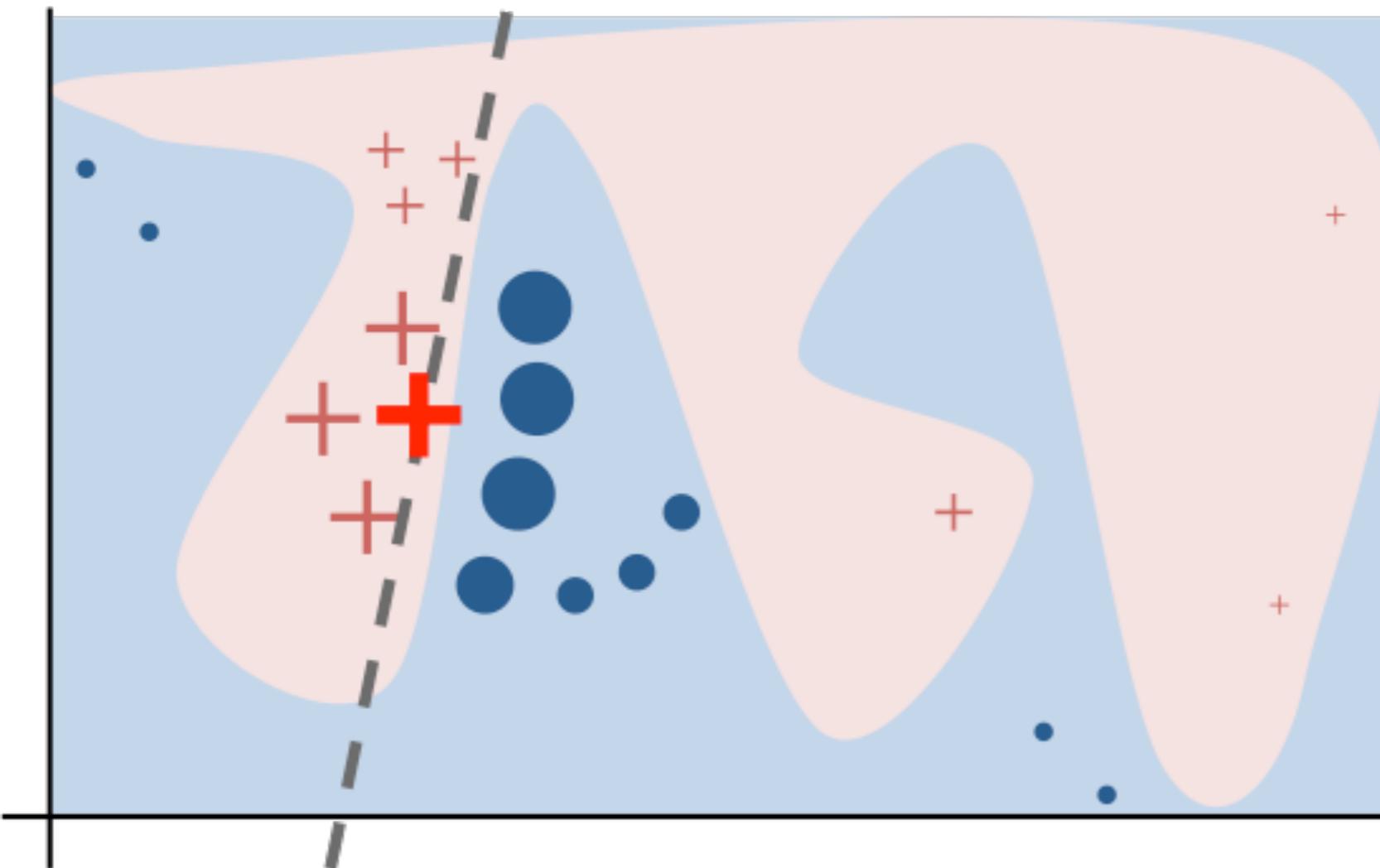
**Table 1:** Models specifically supported by the **pdp** package may still need to supply additional arguments in the call.

# Package pdp - Partial Dependence Plots

- The main function: `partial()`
- Works for various models
- Calculates and plots partial dependence functions
- Produces `ggplot2` plots and lattice plots

# LIME: Local Interpretable Model-agnostic Explanations

1. Generate a fake dataset around  $x$ .
2. Use black-box estimator to get target values  $y$ .
3. Train a new white-box estimator for  $(y, x)$ .
4. Check prediction quality of a white-box classifier.
5. Use white-box estimator as an explanation of black-box model.



"Why Should I Trust You?" Explaining the Predictions of Any Classifier.

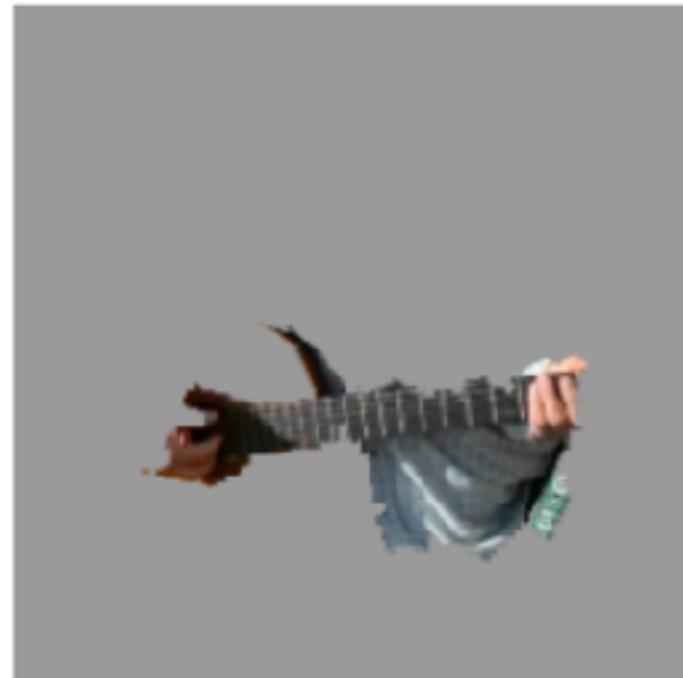
Marco Tulio Ribeiro, Sameer Singh, Carlos Guestrin (2016). <https://arxiv.org/pdf/1602.04938.pdf>

Port to R: Thomas Lin Pedersen (2017) <https://github.com/thomasp85/lime>

# LIME: Local Interpretable Model-agnostic Explanations

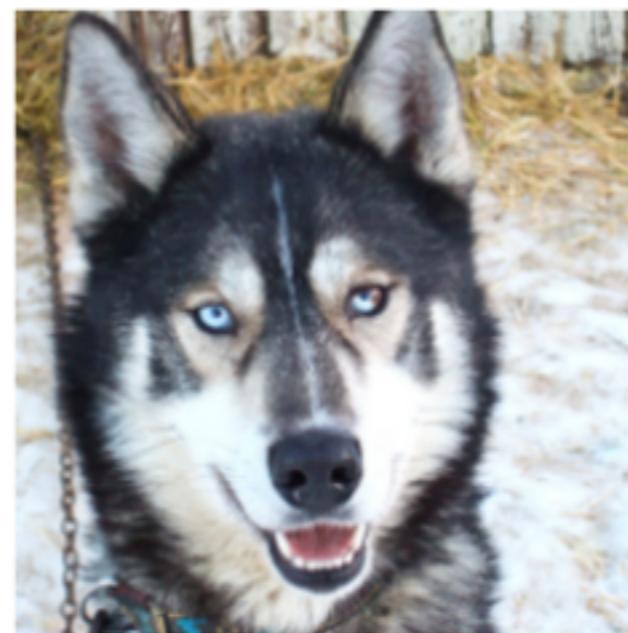


(a) Original Image

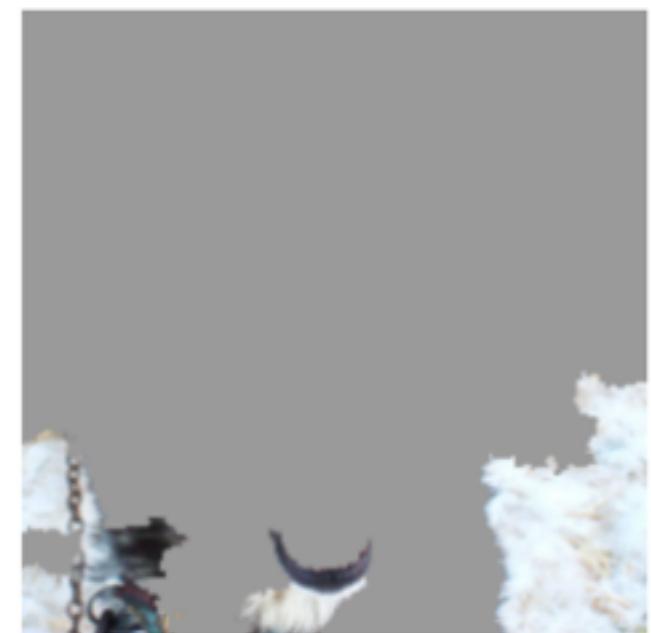


(b) Explaining *Electric guitar*

What went wrong  
and what went OK



(a) Husky classified as wolf



(b) Explanation

An R port <https://github.com/thomasp85/lime>

"Why Should I Trust You?" Explaining the Predictions of Any Classifier.

Marco Tulio Ribeiro, Sameer Singh, Carlos Guestrin (2016). <https://arxiv.org/pdf/1602.04938.pdf>

# Package lime - Local Interpretable Model-agnostic Explanations

- R port of a python package <https://github.com/marcotcr/lime>
- Originated from text-analysis / image-analysis
- Model agnostic
- Fits different „white box“ models

"Why Should I Trust You?" Explaining the Predictions of Any Classifier.  
Marco Tulio Ribeiro, Sameer Singh, Carlos Guestrin (2016). <https://arxiv.org/pdf/1602.04938.pdf>

# Reproducibility

# archivist - reproducible and recordable research

```
library("archivist")
model <- lm(Sepal.Length ~ Sepal.Width, data=iris)
saveToLocalRepo(model)

models <- asearch("pbiecek/graphGallery", patterns = "class:lm")
modelsBIC <- sapply(models, BIC)
sort(modelsBIC)

## 990861c7c27812ee959f10e5f76fe2c3 2a6e492cb6982f230e48cf46023e2e4f
##                               39.05577                         67.52735
## 0a82efeb8250a47718cea9d7f64e5ae7 378237103bb60c58600fe69bed6c7f11
##                               189.73593                         189.73593
## 7f11e03539d48d35f7e7fe7780527ba7 c1b1ef7bcddefb181f79176015bc3931
##                               189.73593                         189.73593
```

# archivist - reproducible and recordable research

[ write local ]	[ read local ]	[ read remote ]	[ shortcuts ]
createLocalRepo(R)			
setLocalRepo(R)	setLocalRepo(R)	setRemoteRepo(R)	
saveToLocalRepo(A, R)			asave(A, R)
	loadFromLocalRepo(H, R)	loadFromRemoteRepo(H, R)	aread(RH)
	searchInLocalRepo(P, R)	searchInRemoteRepo(P, R)	asearch(RH)
rmFromLocalRepo(A, R)			
			asession(RH)
			ahistory(RH)
	summaryLocalRepo(R)	summaryRemoteRepo(R)	
	showLocalRepo(R)	showRemoteRepo(R)	
deleteLocalRepo(R)			

A - artifact, any R object, like `data.frame`, `ggplot`, `Im`

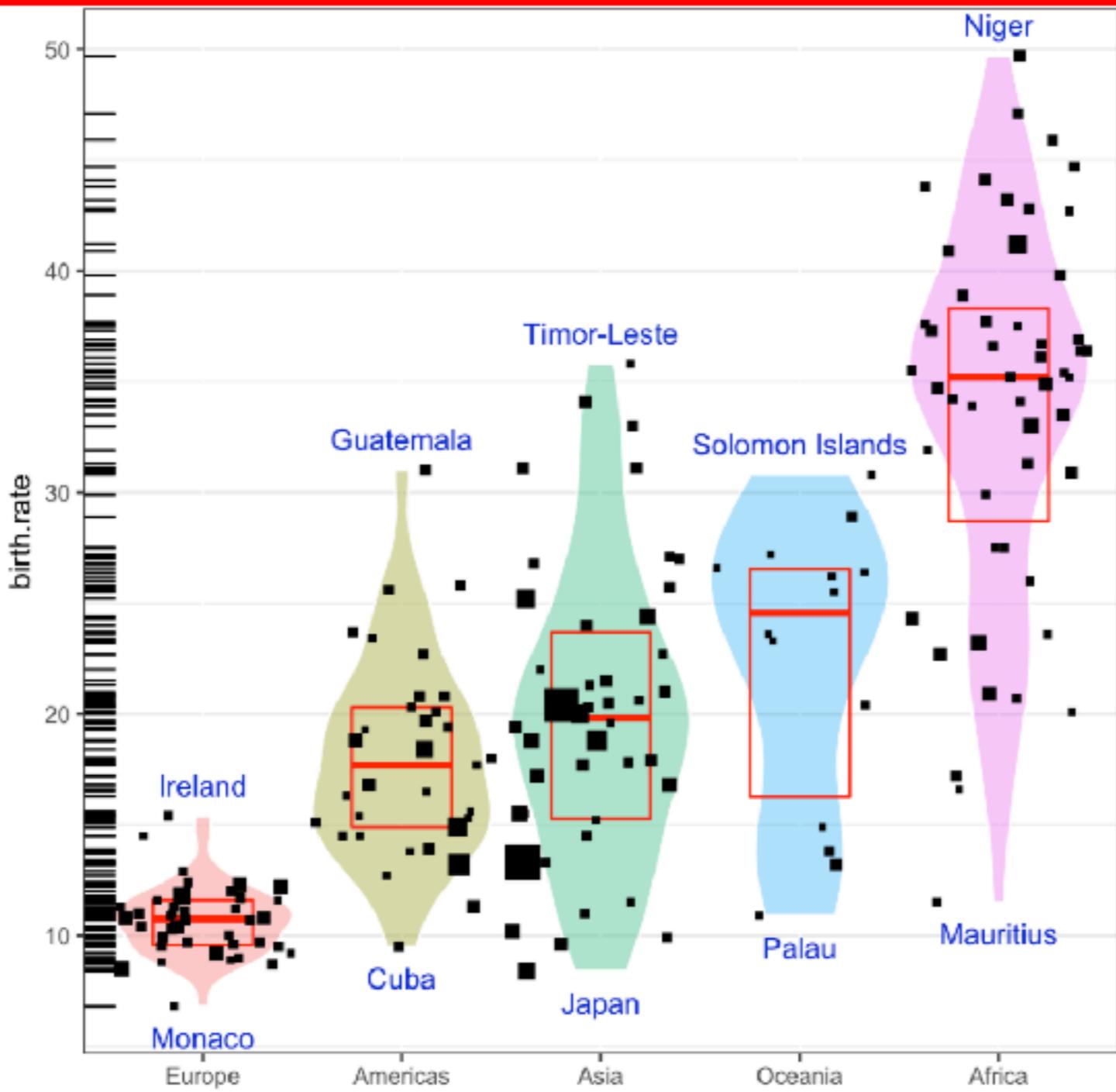
H - md5hash, cryptographical hash of arbitrary R object

P - pattern, used to find artifacts with suitable tags

R - repository, a local repository is a folder, a remote repo is based on git or hg. Repository contains rda dumps, miniatures and data base with object's tags.

```
ggplot(countries, aes(x=continent, y=birth.rate, label=country)) +  
  geom_violin(scale="width", aes(fill=continent), color="white", alpha=0.4) +  
  stat_summary(fun.data = "q3", geom = "crossbar",  
    colour = "red", width = 0.5) +  
  geom_jitter(aes(size=(population)^0.9), position=position_jitter(width = .45, height = 0),  
    shape=15) +  
  geom_rug(sides = "1") +  
  geom_text(data=countriesMin, vjust=2, color="blue3") +  
  geom_text(data=countriesMax, vjust=-1, color="blue3") +  
  theme_bw() + xlab("") + theme(legend.position="none", panel.grid.major.x = element_line(color="white"))
```

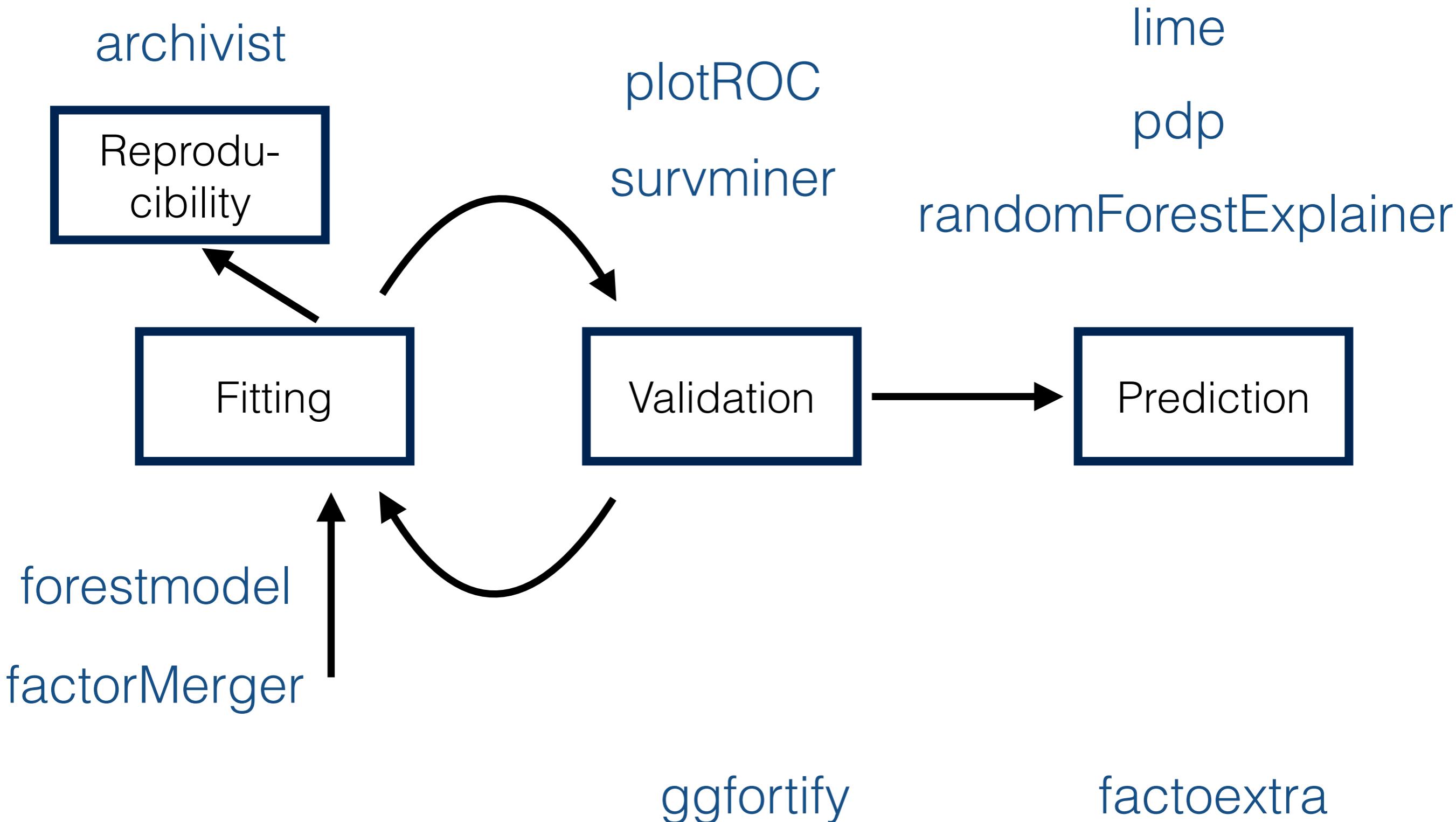
Load: archivist::aread('pbiecek/Eseje/arepo/ba7f58faf7373420e3ddce039558140')



# archivist - reproducible and recordable research

- Set of functions for storing, searching for, and recovering binary copies of R objects
- Repository of R objects that can be hosted as a local foley or as a git/mercurial folder
- Generate a md5 checksum to verify object's identity

# Life-cycle of a typical prognostic model



## Further reading

ELI5 is a Python library which allows to visualize and debug various Machine Learning models

<http://eli5.readthedocs.io/en/latest/index.html>

Ideas on interpreting machine learning.

Patrick Hall, Wen Phan, SriSatish Ambati (2017)

<https://www.oreilly.com/ideas/ideas-on-interpreting-machine-learning>

ggRandomForests: Random Forests for Regression

John Ehrlinger (2015)

<https://arxiv.org/pdf/1501.07196.pdf>

rms: Regression Modeling Strategies.

Frank E Harrell (2009-2017) CRAN

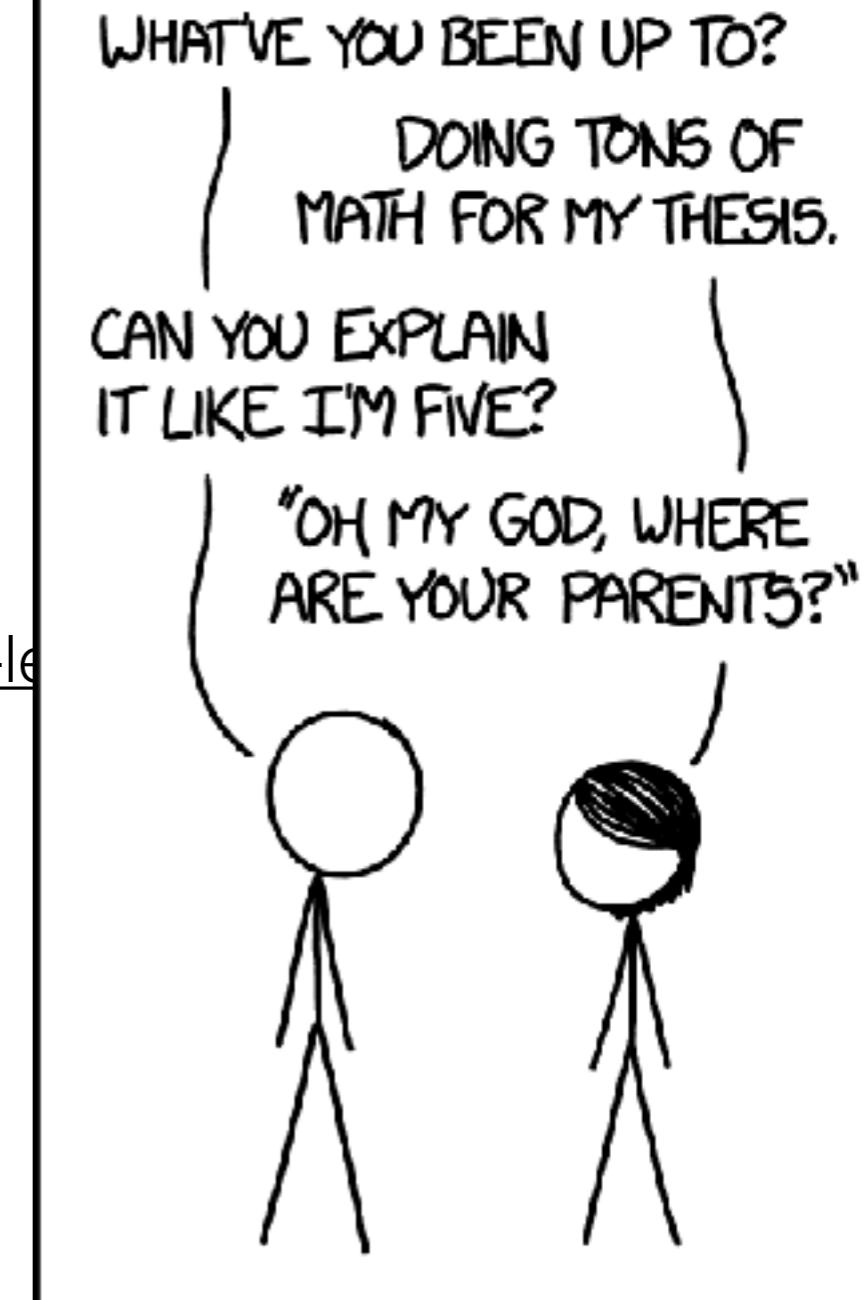
<https://cran.r-project.org/web/packages/rms/index.html>

Visualizing statistical models: Removing the blindfold.

Hadley Wickham, Dianne Cook, Heike Hofmann (2015)

Statistical Analysis and Data Mining

<http://had.co.nz/stat645/model-vis.pdf>



[https://imgs.xkcd.com/comics/like\\_im\\_five.png](https://imgs.xkcd.com/comics/like_im_five.png)

**Thank you for your attention!**