

# imlplots

*Julia Fried, Tobias Riebe, Christian Scholbeck*

*2018-04-11*

## imlplots: interpretable machine learning plots

**imlplots** is an R package that provides an interactive Shiny dashboard for three kinds of Interpretable Machine Learning (IML) plots

- Partial Dependence Plots (PDP)
- Individual Conditional Expectation (ICE) plots
- Accumulated Local Effect (ALE) plots

## Installation

The package can be installed directly from github with devtools

```
# install.packages("devtools")
devtools::install_github('juliafried/imlplots')
library(imlplots)
```

## Quickstart

You can fit classification and regression problems from the **mlr** package and analyse possible interaction effects in the Shiny dashboard.

For quickstart we take the popular Boston Housing data, where we want to predict the median housing price in Boston.

```
head(boston)
```

```
##      crim zn  indus chas   nox    rm  age    dis rad tax ptratio  black
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12
##   lstat medv
## 1   4.98 24.0
## 2   9.14 21.6
## 3   4.03 34.7
## 4   2.94 33.4
## 5   5.33 36.2
## 6   5.21 28.7
```

For using **imlplots** Shiny dashboard, three input arguments need to be specified

- **data** - the input data
- **task** - the learning task

- `models` - one or several trained models

We create a regression task with `medv` as target variable. The task structure is determined by `mlr` package.

```
boston.task = makeRegrTask(data = boston, target = "medv")
```

The `imlplots` dashboard allows the comparison of multiple learning algorithms, therefore we fit two different models - first a random forest and second a SVM.

```
rf.mod = train("regr.randomForest", boston.task)
glm.mod = train("regr.glm", boston.task)
```

The input for the Shiny app is a list of learners.

```
mod.list = list(rf.mod, glm.mod)
```

Now the Shiny app can be used.

```
imlplots(data = boston, task = boston.task, models = mod.list)
```

## Code for Copy & Paste

```
library(imlplots)

boston.task = makeRegrTask(data = boston, target = "medv")

rf.mod = train("regr.randomForest", boston.task)
glm.mod = train("regr.glm", boston.task)

mod.list = list(rf.mod, glm.mod)

imlplots(data = boston, task = boston.task, models = mod.list)
```

## Further Examples

### IML Plots for Regression Tasks

To show how you can use the `imlplots` Shiny app for regression tasks we use fire data, where the burned area of forests due to fires should be analyzed.

```
head(fire)
```

```
##   month day FFMC  DMC    DC  ISI temp RH wind rain area
## 1     3   5 86.2 26.2  94.3  5.1  8.2 51  6.7  0.0   0
## 2    10   2 90.6 35.4 669.1  6.7 18.0 33  0.9  0.0   0
## 3    10   6 90.6 43.7 686.9  6.7 14.6 33  1.3  0.0   0
## 4     3   5 91.7 33.3  77.5  9.0  8.3 97  4.0  0.2   0
## 5     3   7 89.3 51.3 102.2  9.6 11.4 99  1.8  0.0   0
## 6     8   7 92.3 85.3 488.0 14.7 22.2 29  5.4  0.0   0
```

The target variable is `area`, which is between 0.00 and 1090.84 ha.

```
summary(fire$area)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    0.00    0.52   12.85    6.57 1090.84
```

We create a regression task with target variable `area`.

```
fire.task = makeRegrTask(data = fire, target = "area")
```

Next we train some `mlr` models and create a list of models. Note: The order in your model list will determine the model order in the Shiny dashboard.

```
fire.rf = train("regr.randomForest", fire.task)
fire.glm = train("regr.glm", fire.task)
```

```
fire.models = list(fire.rf, fire.glm)
```

Now we can open the `imlplots` Shiny app.

```
imlplots(data = fire, task = fire.task, models = fire.models)
```

The Shiny dashboard contains four tabs

- Data
- Settings
- Plots
- Learner Summary

The **Data** tab shows your used input data. This data is taken to generate IML plots. If you want to check how changes in the data effect your plot, you can simply filter in the **Data** tab.

LMU Data Science Innovationslabor Reload application

Interactive plots for interpretable machine learning

Data Settings Plots Learner Summary

Data used for plotting ICE curves

Select observations to sample from

Plot all sampled observations

Show 10 entries

Search:

	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
	All	All	All	All	All	All	All	All	All	All	All
1	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0	0
2	oct	tue	90.6	35.4	669.1	6.7	18	33	0.9	0	0
3	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0	0
4	mar	fri	91.7	33.3	77.5	9	8.3	97	4	0.2	0
5	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0	0
6	aug	sun	92.3	85.3	488	14.7	22.2	29	5.4	0	0
7	aug	mon	92.3	88.9	495.6	8.5	24.1	27	3.1	0	0
8	aug	mon	91.5	145.4	608.2	10.7	8	86	2.2	0	0
9	sep	tue	91	129.5	692.6	7	13.1	63	5.4	0	0
10	sep	sat	92.5	88	698.6	7.1	22.8	40	4	0	0

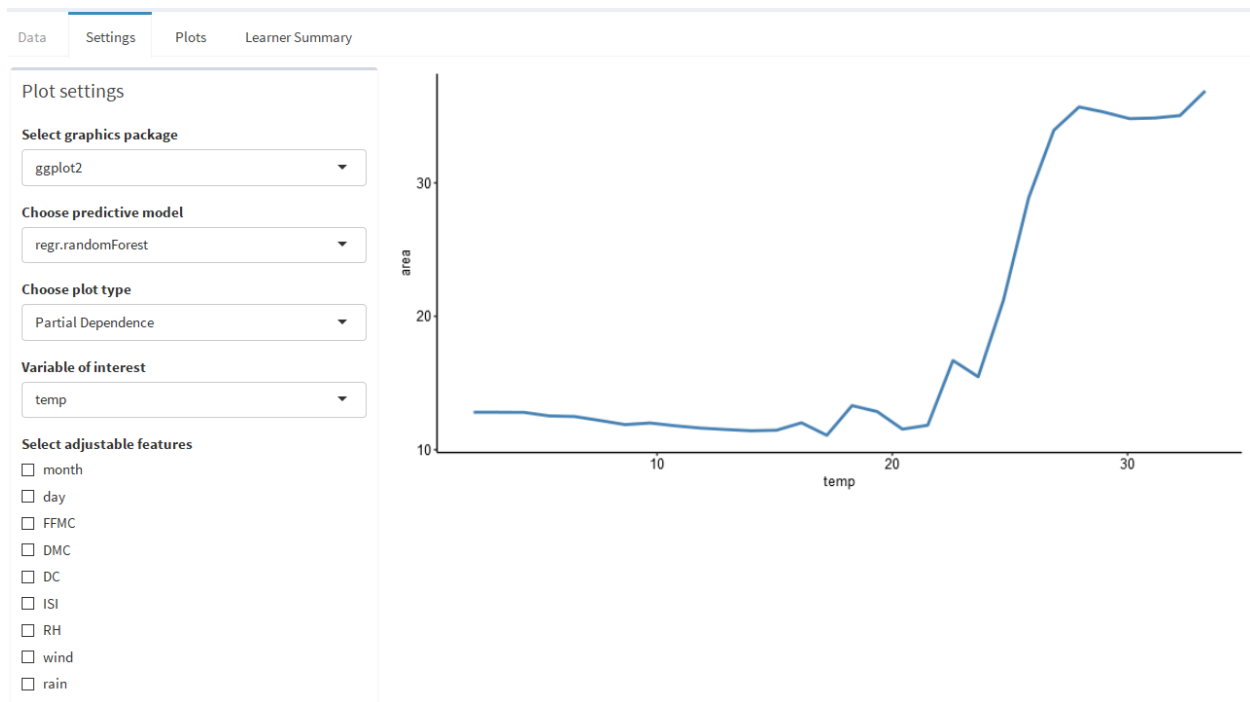
Showing 1 to 10 of 517 entries

Previous 1 2 3 4 5 ... 52 Next

For filtering two options are given

1. Plot all sampled observations: In this setting you can filter via the filters beneath the column titles and all rows will be used for plotting.
2. Plot individual observations: In this setting after using the filters, you have to manually select specific rows.

The next tab **Settings** contains all possible plot settings and the selected IML plot.



There are various settings

1. **Select graphics package:** You can select the graphics package - we offer **ggplot2** and **plotly**. Use **ggplot2** if your computer is not the fastest one.
2. **Choose predictive model:** Choose one of your fitted models. The order in the dropdown is the order of your list.
3. **Choose plot type:** We offer PDP, ICE and ALE plots. If you select ICE plot, you will get a new selection field. Possible are **centered** and **regular** ICE plots.
4. **Variable of interest:** This dropdown will determine the x-axis of your plot.

On the right side of the dashboard page, the selected plot is shown.

To check out effects, you can turn on **Select adjustable features**. This option allows you to set one of the variables to a specific value.

Select adjustable features

☐ month
☒ day
☐ FFMCI
☐ DMC
☐ DC
☐ ISI
☐ RH
☐ wind
☐ rain

Number of knots for each line

1

30

517

Number of individual observations (lines) to sample from data

1

30

517

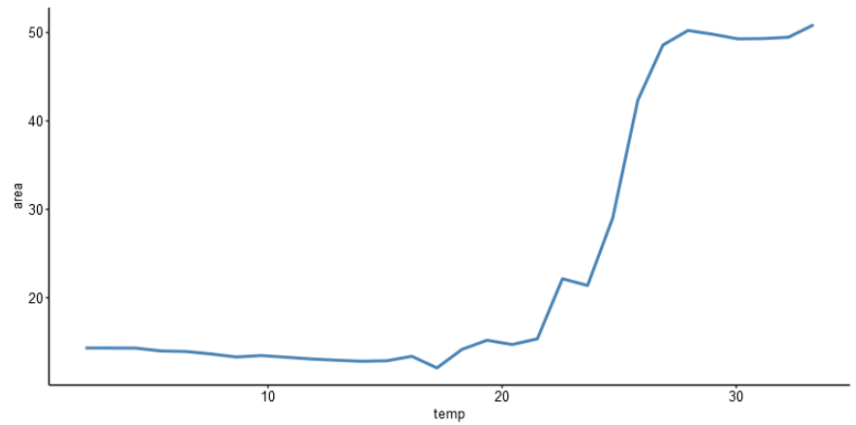
Adjust feature values

day

1

4

7



It is also possible to change the number of knots and lines (individual observations) with the shown sliders. The ICE plot contains all sampled, individual observations in blue. The red line is from PDP.

Plot settings

Select graphics package

ggplot2

Choose predictive model

regr.randomForest

Choose plot type

Individual Conditional Expectation

Ice plot mode

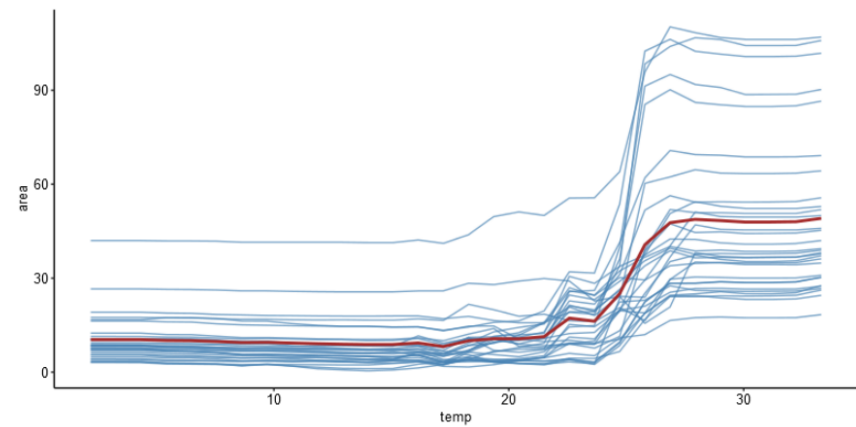
Regular

Variable of interest

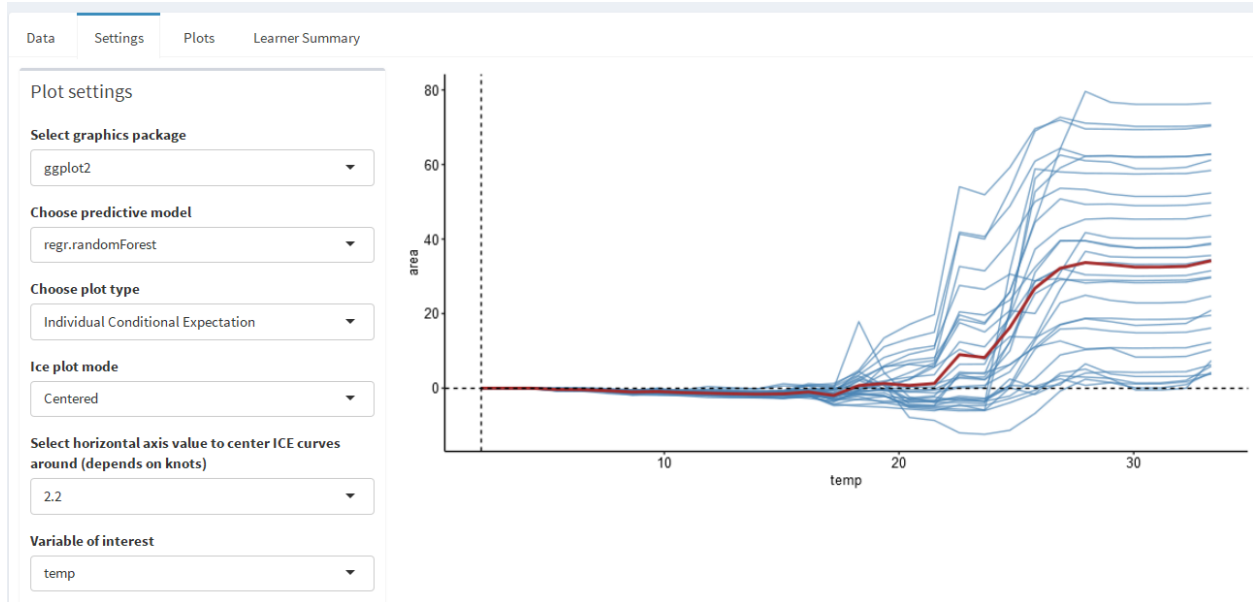
temp

Select adjustable features

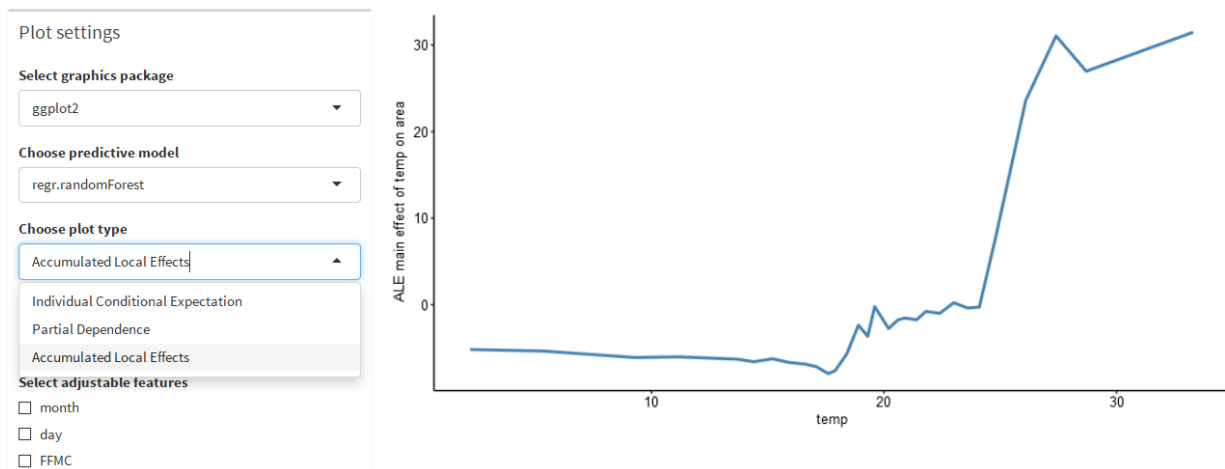
☐ month



As described above, you can select between **Regular** and **Centered** ICE plots.



The ALE plot can be selected, too. Please keep in mind, that the ALE plot has a different y-axis than the PDP and ICE plot.



For ALE plots you can switch between two ALE Plot Modes. The **Main Effects** mode allows you to select one variable of interest and shows its interaction effect. The **Second Order Effects** setting allows to select another ALE **interaction variable** and therefore shows the effect for this extra variable too. If you select **plotly** as graphics package, the second order effects ALE plot will be a 3D plot.

**Select graphics package**  
 plotly (resource intensive) ▼

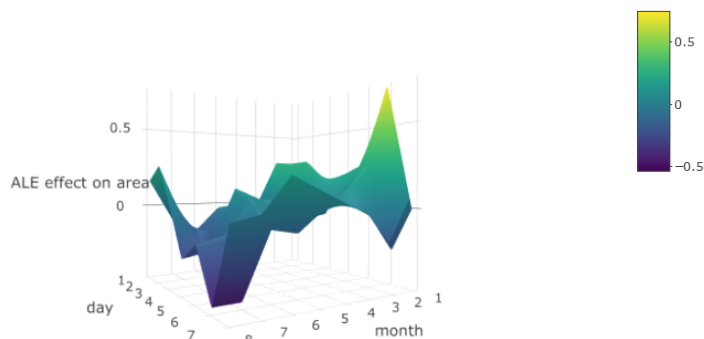
**Choose predictive model**  
 regr.randomForest ▼

**Choose plot type**  
 Accumulated Local Effects ▼

**ALE Plot Mode**  
 Second Order Effects ▼

**Variable of interest**  
 month ▼

**ALE interaction variable**  
 day ▼



The third tab **Plots** shows the IML plot in full screen via the sub-tab **Zoomed plot**. The sub-tab **Scatterplot** shows the filtered and unfiltered scatterplot between the **variable of interest** and the **target** variable of the model.

In the **Data** tab we filtered for a high value of burned area and selected three individual observations.

Data Settings Plots Learner Summary

Data used for plotting ICE curves

Select observations to sample from  
 Plot individual observations ▼

Show 10 entries

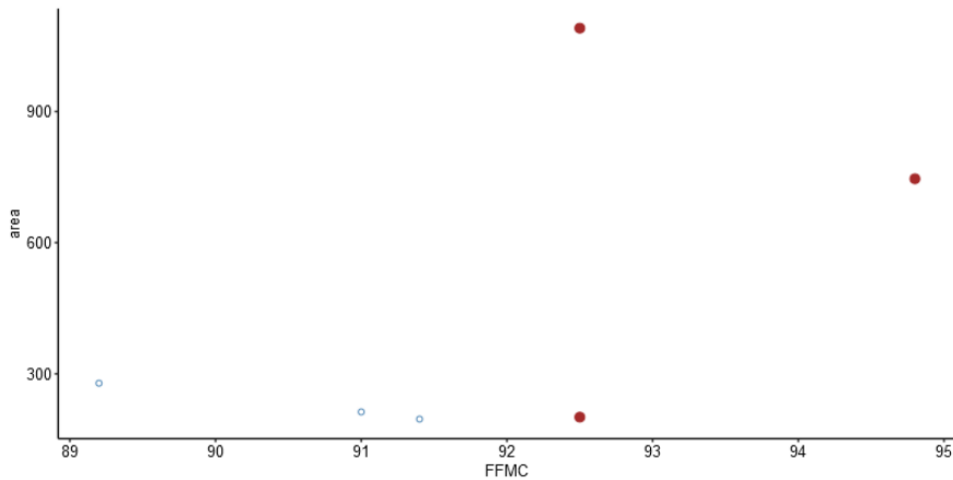
	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
	All	All	All	All	All	All	All	All	All	All	193.31 ... 1090.8
236	8	7	91.4	142.4	601.4	10.6	19.6	41	5.8	0	196.48
237	9	6	92.5	121.1	674.4	8.6	18.2	46	1.8	0	200.94
238	9	2	91	129.5	692.6	7	18.8	40	2.2	0	212.88
239	9	6	92.5	121.1	674.4	8.6	25.1	27	4	0	1090.84
416	8	4	94.8	222.4	698.6	13.9	27.5	27	4.9	0	746.28
480	7	1	89.2	103.9	431.6	6.4	22.6	57	4.9	0	278.53

Showing 1 to 6 of 6 entries (filtered from 517 total entries)

Previous 1 Next

The filtered data scatterplot shows the selected high area values and also the three individual observations (in red).

## Filtered data



The unfiltered data scatterplot shows all data points and also the three individual observations (in red).

The fourth tab **Learner summary** shows the currently selected learner summary. If you want to see another summary, you have to select another model in the **Settings** tab.

Data
Settings
Plots
Learner Summary

```

[1] ""
[2] "Call:"
[3] " randomForest(x = data[["data"]], y = data[["target"]], keep.inbag = if (is.null(keep.inbag)) TRUE else keep.inbag) "
[4] "          Type of random forest: regression"
[5] "          Number of trees: 500"
[6] "No. of variables tried at each split: 3"
[7] ""
[8] "          Mean of squared residuals: 4187.245"
[9] "          % Var explained: -3.54"

```

## Code for Copy & Paste

```

library(imlplots)

fire.task = makeRegrTask(data = fire, target = "area")

fire.rf = train("regr.randomForest", fire.task)
fire.lm = train("regr.lm", fire.task)
fire.glm = train("regr.glm", fire.task)

fire.models = list(fire.rf, fire.lm, fire.glm)

implots(data = fire, task = fire.task, models = fire.models)

```



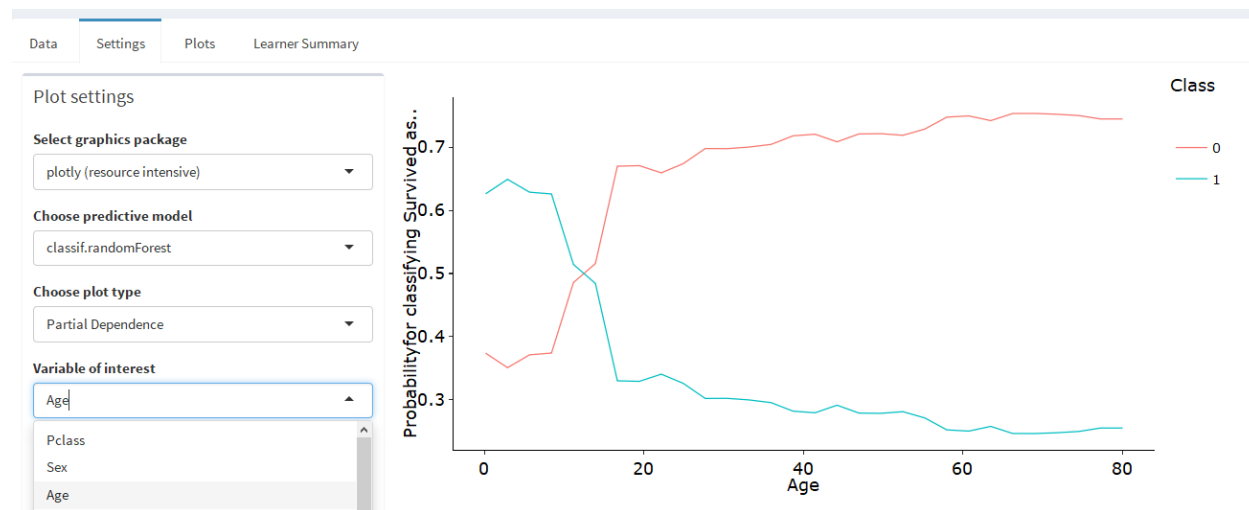
## IML Plots for Classification Tasks

For the classification example only the differences to the regression example will be explained. We use the titanic data set from kaggle for this example.

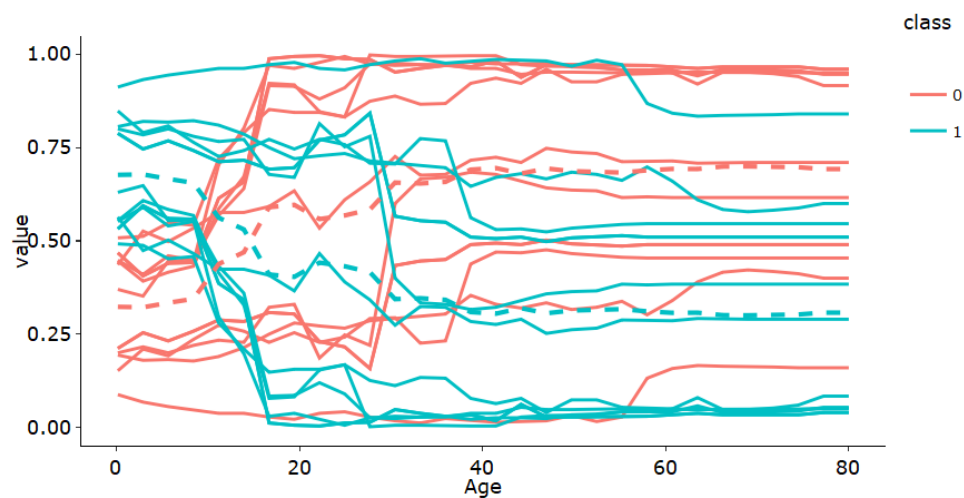
Again we create a task. This time we only fit one random forest. Afterwards we open the app.

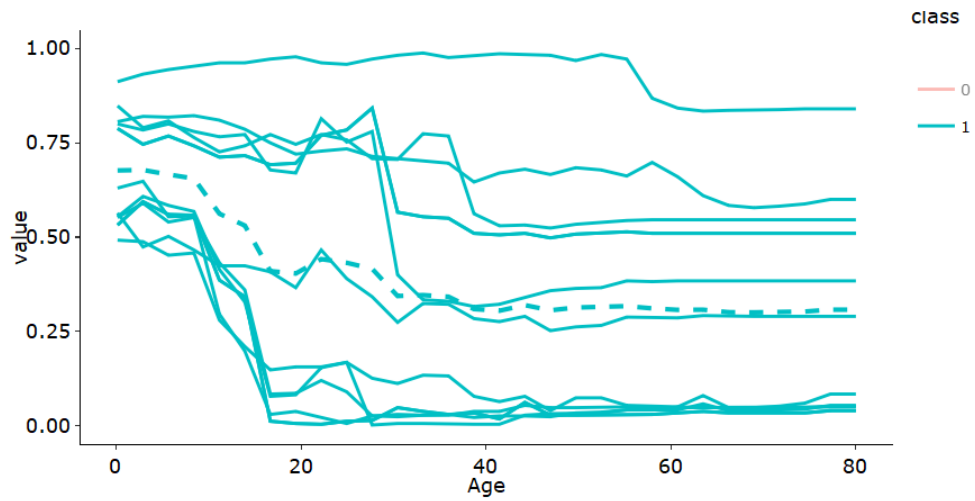
```
titanic.task = makeClassifTask(data = titanic, target = "Survived")
titanic.rf = train("classif.randomForest", titanic.task)
imlplots(data = titanic, task = titanic.task, titanic.rf)
```

This time it is useful to select plotly in the Select graphics package dropdown.



This allows you to deselect single classes to increase the visibility of individual lines, which is very useful for ICE plot.





Code for Copy & Paste

```
library(imlplots)

titanic.task = makeClassifTask(data = titanic, target = "Survived")

titanic.rf = train("classif.randomForest", titanic.task)

implots(data = titanic, task = titanic.task, titanic.rf)
```

## References

- References