DATA SOCIETY:

Interactive Plotting Libraries - 1

One should look for what is and not what he thinks should be. (Albert Einstein)

Interactive plotting libraries: topic introduction

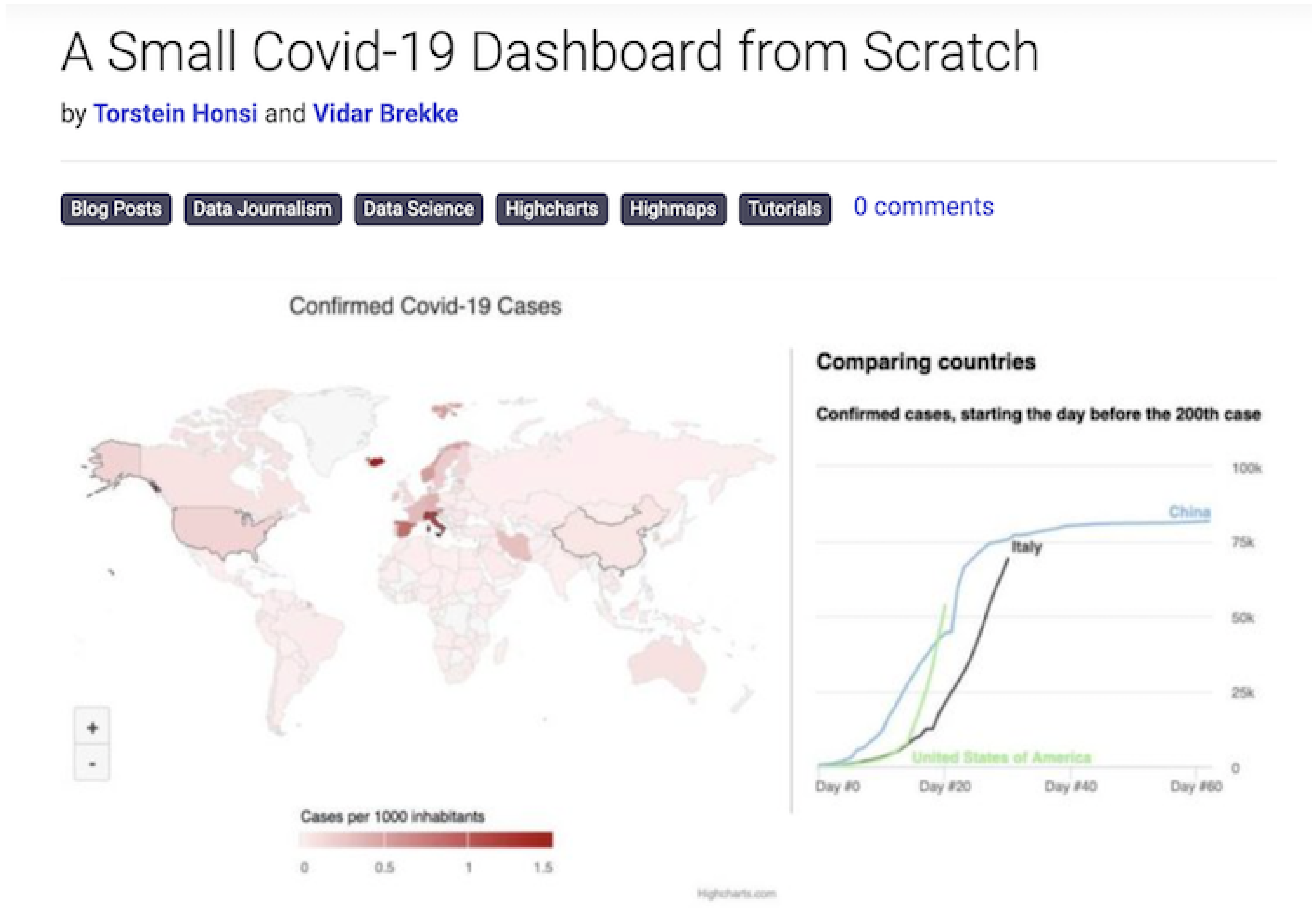
In this part of the course, we will cover the following concepts:

- Discover different functions to build interactive visualizations
- Visualize data with `highcharter`

Warm-up

- During the COVID-19 pandemic, the demand for creating quick visualizations to put data into perspective rose quickly

- Before we begin, ***read this article about data visualization*** and check out an interactive visualization dashboard



A Small Covid-19 Dashboard from Scratch

by Torstein Honsi and Vidar Brekke

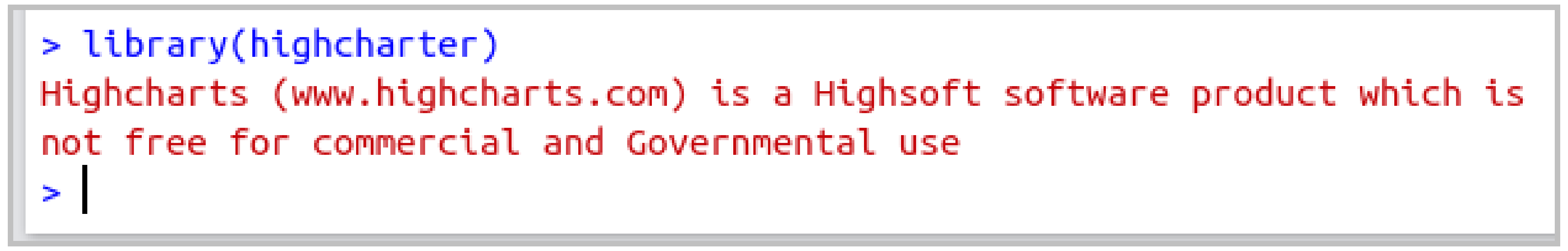
Blog Posts | Data Journalism | Data Science | Highcharts | Highmaps | Tutorials | 0 comments

Module completion checklist

Objective	Complete
Install the highcharter package and discuss its application to build interactive visualizations	
Create a scatterplot using highcharter with tidy data	

Interactive visualizations with `highcharter`

- Highcharter is an R wrapper that allows R users to tap into one of the most comprehensive data visualization JavaScript-based libraries
- Though free for individual research and non-profit purposes, there are some restrictions
- You may need a license to integrate it into a software or organization-wide products
- For more information, ***refer to Highcharter's website***

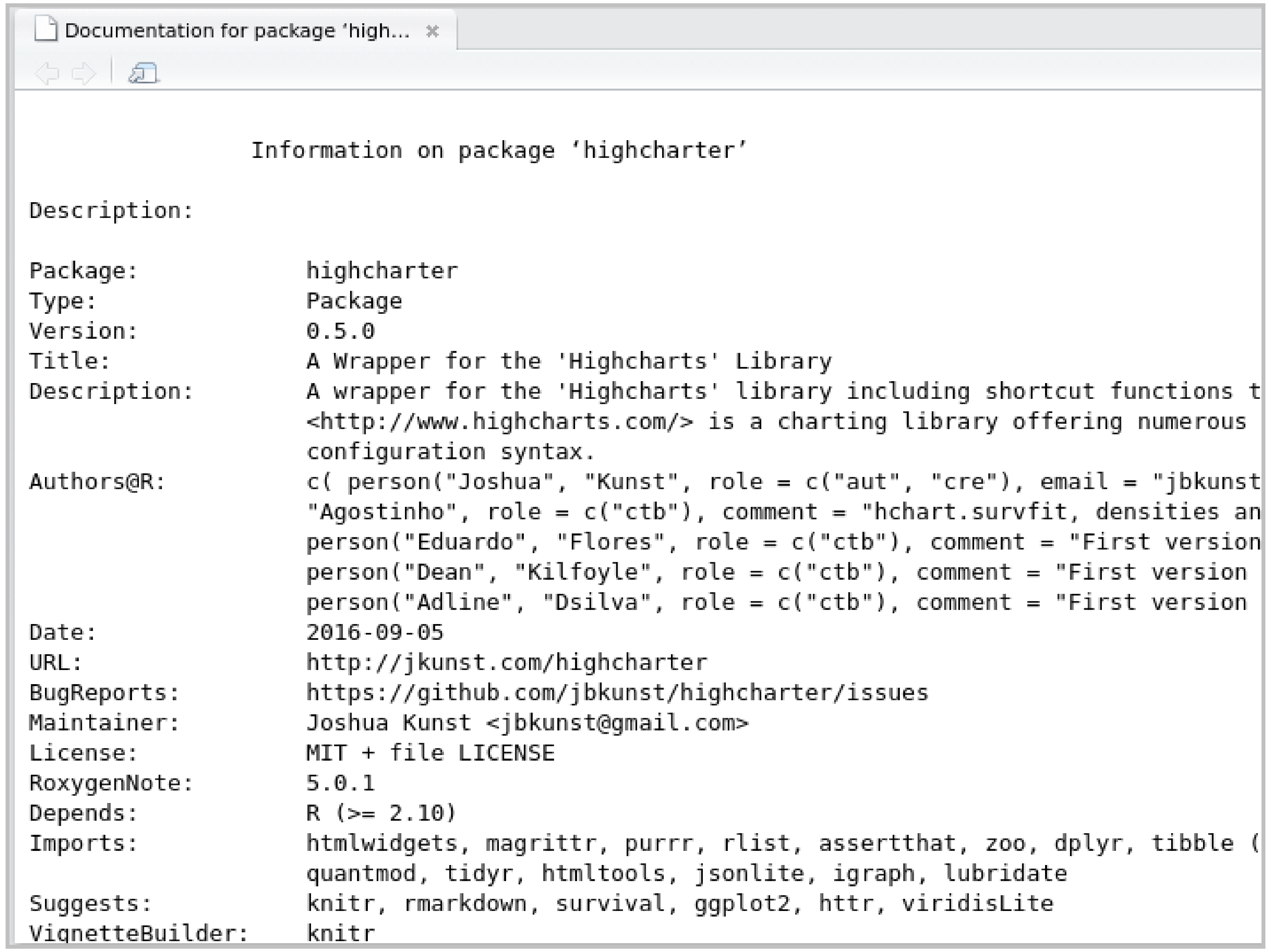


```
> library(highcharter)
Highcharts (www.highcharts.com) is a Highsoft software product which is
not free for commercial and Governmental use
> |
```

Installing `highcharter`

- Let's install the package and check its documentation

```r
# Install `highcharter` package.
install.packages("highcharter")

# Load the library.
library(highcharter)

# View documentation.
library(help = "highcharter")
```

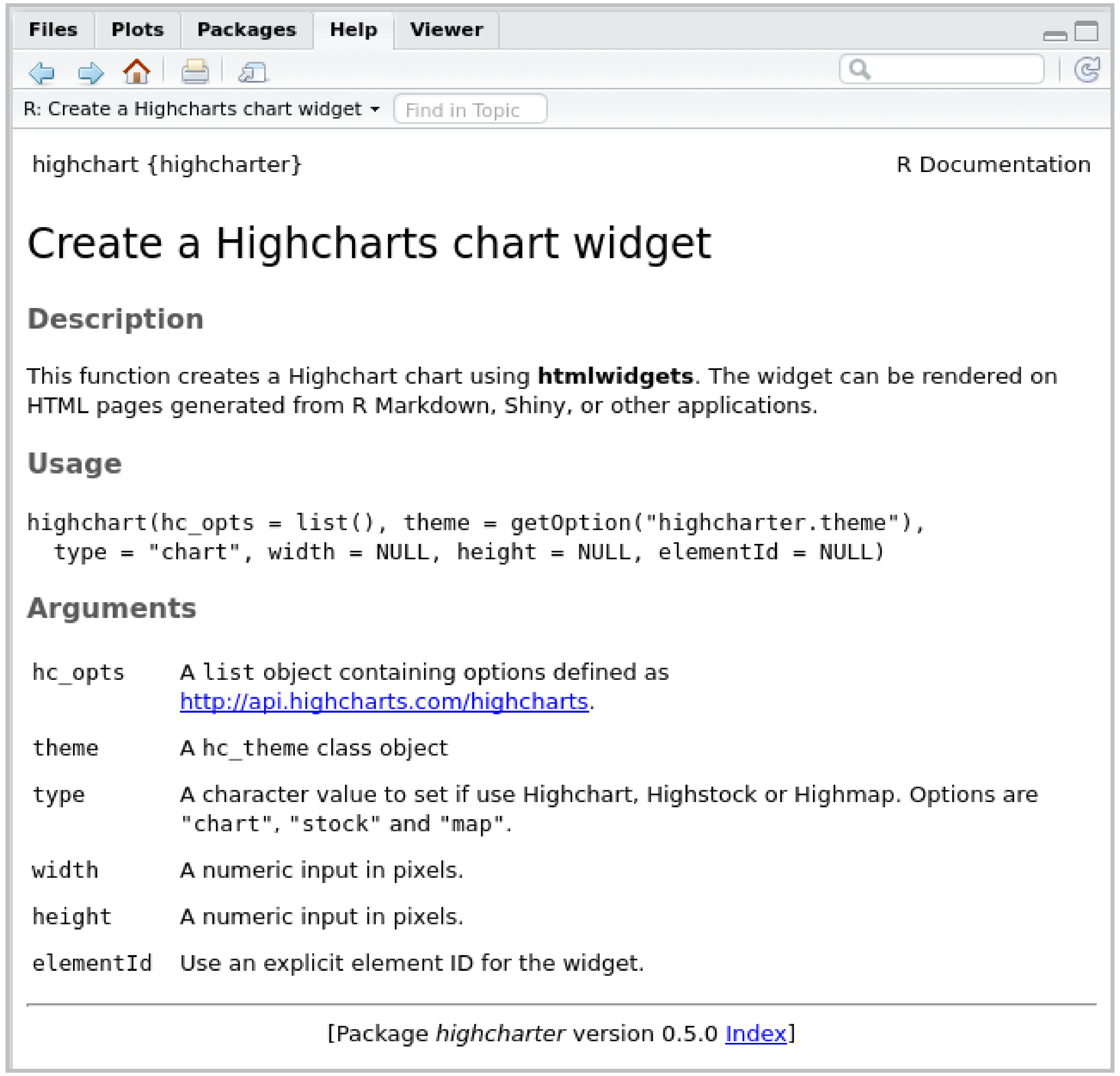


```
                 Information on package 'highcharter'

Description:

Package:            highcharter
Type:               Package
Version:            0.5.0
Title:              A Wrapper for the 'Highcharts' Library
Description:        A wrapper for the 'Highcharts' library including shortcut functions t
                    <http://www.highcharts.com/> is a charting library offering numerous
                    configuration syntax.
Authors@R:          c( person("Joshua", "Kunst", role = c("aut", "cre"), email = "jbkunst
                    "Agostinho", role = c("ctb"), comment = "hchart.survfit, densities an
                    person("Eduardo", "Flores", role = c("ctb"), comment = "First version
                    person("Dean", "Kilfoyle", role = c("ctb"), comment = "First version
                    person("Adline", "Dsilva", role = c("ctb"), comment = "First version
Date:               2016-09-05
URL:                http://jkunst.com/highcharter
BugReports:         https://github.com/jbkunst/highcharter/issues
Maintainer:         Joshua Kunst <jbkunst@gmail.com>
License:            MIT + file LICENSE
RoxygenNote:        5.0.1
Depends:            R (>= 2.10)
Imports:            htmlwidgets, magrittr, purrr, rlist, assertthat, zoo, dplyr, tibble (
                    quantmod, tidyr, htmltools, jsonlite, igraph, lubridate
Suggests:           knitr, rmarkdown, survival, ggplot2, httr, viridisLite
VignetteBuilder:    knitr
```

Using `highchart()` function

```
?highchart
```

- To create a plot, we need to call the main plotting function `highchart()`
- The function doesn't need any required arguments
- The graphic parameters and plotting options can be specified within the layers



highchart {highcharter} R Documentation

Create a Highcharts chart widget

Description

This function creates a Highchart chart using **htmlwidgets**. The widget can be rendered on HTML pages generated from R Markdown, Shiny, or other applications.

Usage

```
highchart(hc_opts = list(), theme = getOption("highcharter.theme"),
  type = "chart", width = NULL, height = NULL, elementId = NULL)
```

Arguments

`hc_opts`	A list object containing options defined as http://api.highcharts.com/highcharts.
`theme`	A hc_theme class object
`type`	A character value to set if use Highchart, Highstock or Highmap. Options are "chart", "stock" and "map".
`width`	A numeric input in pixels.
`height`	A numeric input in pixels.
`elementId`	Use an explicit element ID for the widget.

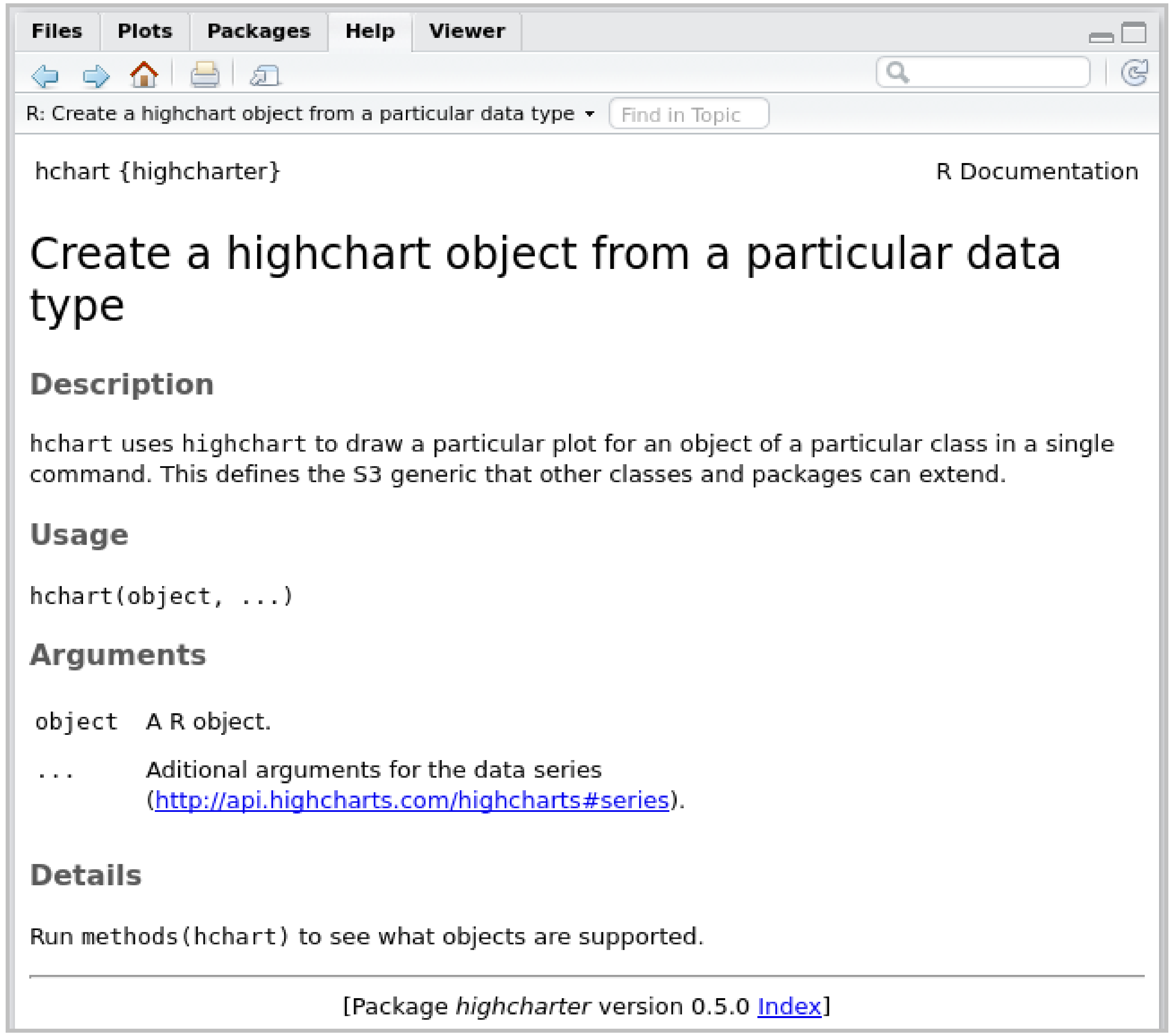
[Package *highcharter* version 0.5.0 Index]

hchart() vs highchart()

- `hchart` is a shorthand version of the `highchart` function

```
?hchart

hchart(Some_data,         #<- dataset to use
       "plot_type",       #<- plot type to use
       hcaes(x = variable1, #<- x-axis mapping
             y = variable2, #<- y-axis mapping
             group = variable3, #<- group by
             ...))
```

- It takes the following arguments:
  - a **dataset** to use
  - the **type of plot** to create (e.g., `scatter`, `bar`, `column`, `line`, etc.)
  - `hcaes` (i.e., highchart **aesthetics**) for mapping variables as layers (just as with `ggplot2`)



hchart {highcharter} R Documentation

Create a highchart object from a particular data type

Description

hchart uses highchart to draw a particular plot for an object of a particular class in a single command. This defines the S3 generic that other classes and packages can extend.

Usage

```
hchart(object, ...)
```

Arguments

`object`	A R object.
`...`	Aditional arguments for the data series (http://api.highcharts.com/highcharts#series).

Details

Run `methods(hchart)` to see what objects are supported.

[Package *highcharter* version 0.5.0 Index]

Layers in Highcharter: series

- The highcharter library has its own vocabulary
- Each new data / graphic layer in highcharter is called a **series**
- Series can be of different types; some common ones are listed below:

Highcharter series type	Plot type
`scatter`	scatterplot
`line`	line graph
`boxplot`	boxplot
`column`	bar plot
`bar`	horizontal bar plot
`histogram`	histogram
`area`	density

Module completion checklist

Objective	Complete
Install the highcharter package and discuss its application to build interactive visualizations	✔
Create a scatterplot using highcharter with tidy data	

Directory settings

- In order to maximize the efficiency of your workflow, use the `box` package and encode your directory structure into `variables`
- Let the `main_dir` be the variable corresponding to your materials folder

```
# Set `main_dir` to the location of your materials folder.

path = box::file()
main_dir = dirname(dirname(path))
```

Directory settings (cont’d)

- We will store all datasets in the `data` directory inside the materials folder in your environment; hence we will save their path to a `data_dir` variable
- We will save all the plots in the `plots` directory corresponding to `plot_dir` variable

- To append one string to another, use `paste0` command and pass the strings you would like to paste together

```
# Make `data_dir` from the `main_dir` and
# remainder of the path to data directory.
data_dir = paste0(main_dir, "/data")
# Make `plots_dir` from the `main_dir` and
# remainder of the path to plots directory.
plot_dir = paste0(main_dir, "/plots")
```

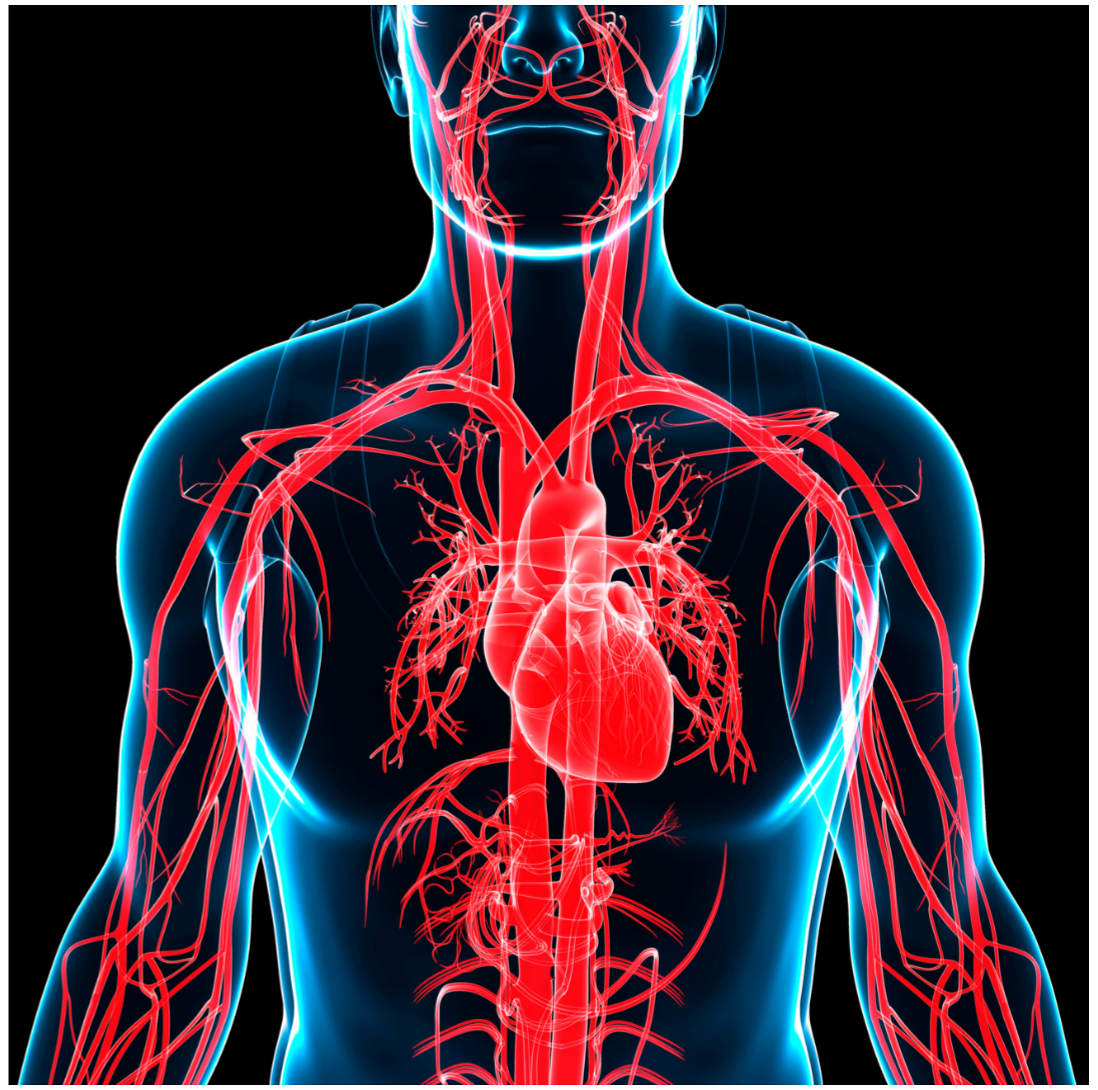
Introducing HDS data set

- We will begin by exploring a dataset called `healthcare-dataset-stroke-data`
- This dataset contains information about age, gender, hypertension, bmi, and other parameters to know the chances of getting a stroke
- The goal is to understand how different variables in the dataset affect the chances of a person suffering from a stroke
- The dataset has 12 characteristics (columns), of which:
  - **10 columns** relate to the **quality and characteristics** of the life of different people
  - The **stroke column** represents whether the people had a stroke or not

Load HDS dataset

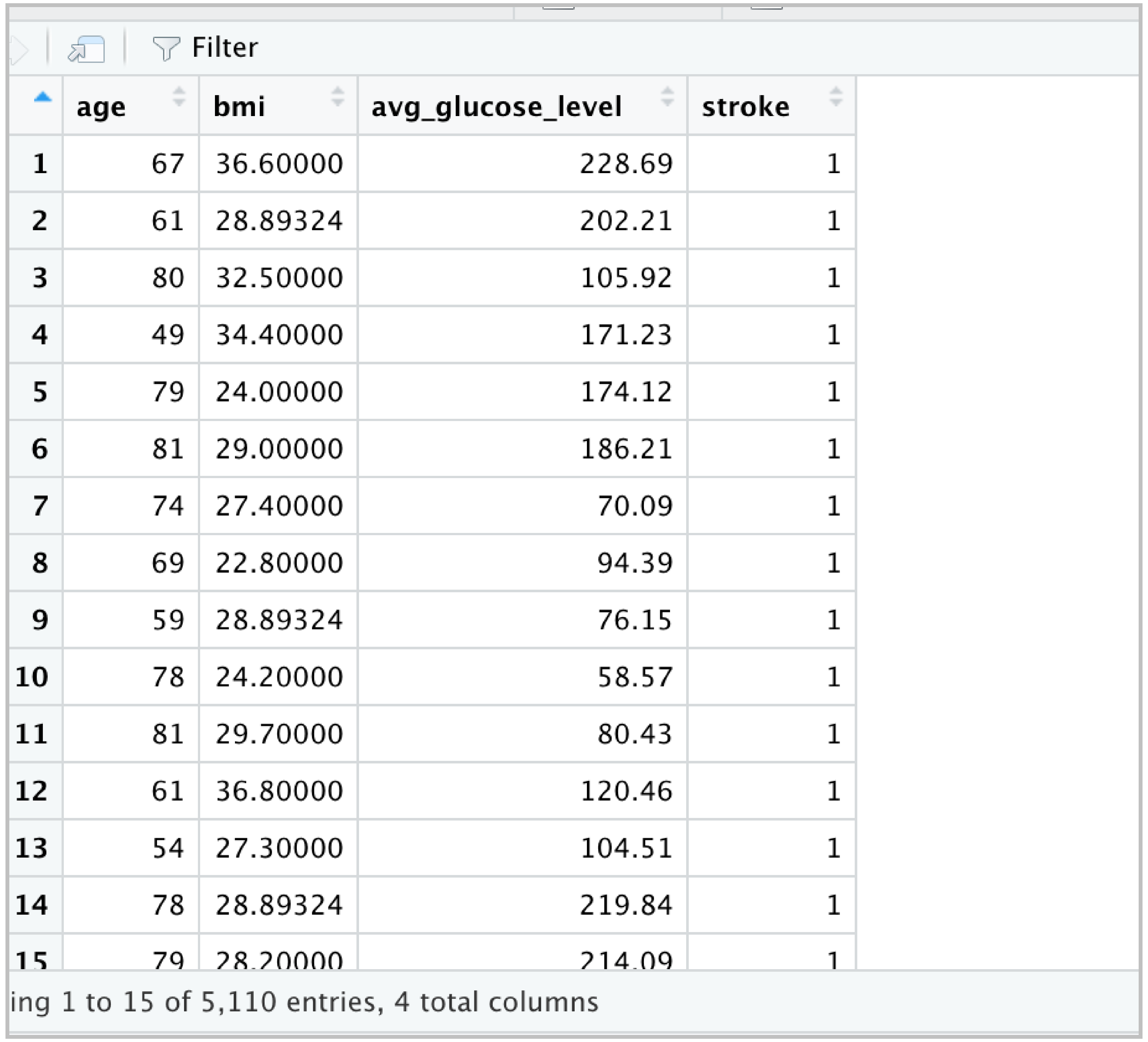
- Let's load the HDS dataset from our `data_dir` into R's environment and subset it

```
# Read CSV file called "healthcare-dataset-
stroke-data.csv"
HDS = read.csv(file =
file.path(data_dir,"/healthcare-dataset-stroke-
data.csv"), #<- provide file path
               header = TRUE,            #<- if
file has header set to TRUE
               stringsAsFactors = FALSE) #<- read
strings as characters, not as factors
```



Subset data

- In this module, we will explore a dataset subset, including the following variables:
  - age
  - bmi
  - average_glucose_level and
  - stroke



Filter

	age	bmi	avg_glucose_level	stroke
1	67	36.60000	228.69	1
2	61	28.89324	202.21	1
3	80	32.50000	105.92	1
4	49	34.40000	171.23	1
5	79	24.00000	174.12	1
6	81	29.00000	186.21	1
7	74	27.40000	70.09	1
8	69	22.80000	94.39	1
9	59	28.89324	76.15	1
10	78	24.20000	58.57	1
11	81	29.70000	80.43	1
12	61	36.80000	120.46	1
13	54	27.30000	104.51	1
14	78	28.89324	219.84	1
15	79	28.20000	214.09	1

ing 1 to 15 of 5,110 entries, 4 total columns

Prepare Data

- But before sub-setting the data, let’s handle the missing data in the dataset
- Then convert `bmi` into a numeric column followed by imputing the missing values with the mean

```r
HDS$bmi <- as.numeric(as.character(HDS$bmi)) ##converting bmi column to numeric
# NA imputation
# we can use is.na() function to know about NA values
HDS$bmi[is.na(HDS$bmi)]<-mean(HDS$bmi,na.rm=TRUE) # Replacing na values of bmi column with it's
mean bmi
```

Prepare Data

- Let’s tidy the data and transform it from a wide to a long format
- This will be especially useful later for univariate visualizations

```
library(tidyverse)
```

```
# Now Let's make a vector of column indices we would like to save.

column_ids = select(HDS, age,bmi,avg_glucose_level,stroke)
HDS_subset = column_ids
```

Create a subset

- Now let's create a different subset to help us build a scatterplot and inspect the head

```
# Prep data for scatterplot

HDS_subset_long = HDS_subset %>%
  gather(-age, #<- gather all variables but `age`
         key = "variable",
         value = "value") %>%
  # All other transformations we've done before.
  group_by(variable) %>%
  mutate(norm_value = value/mean(value, na.rm = TRUE))

head(HDS_subset_long)
```

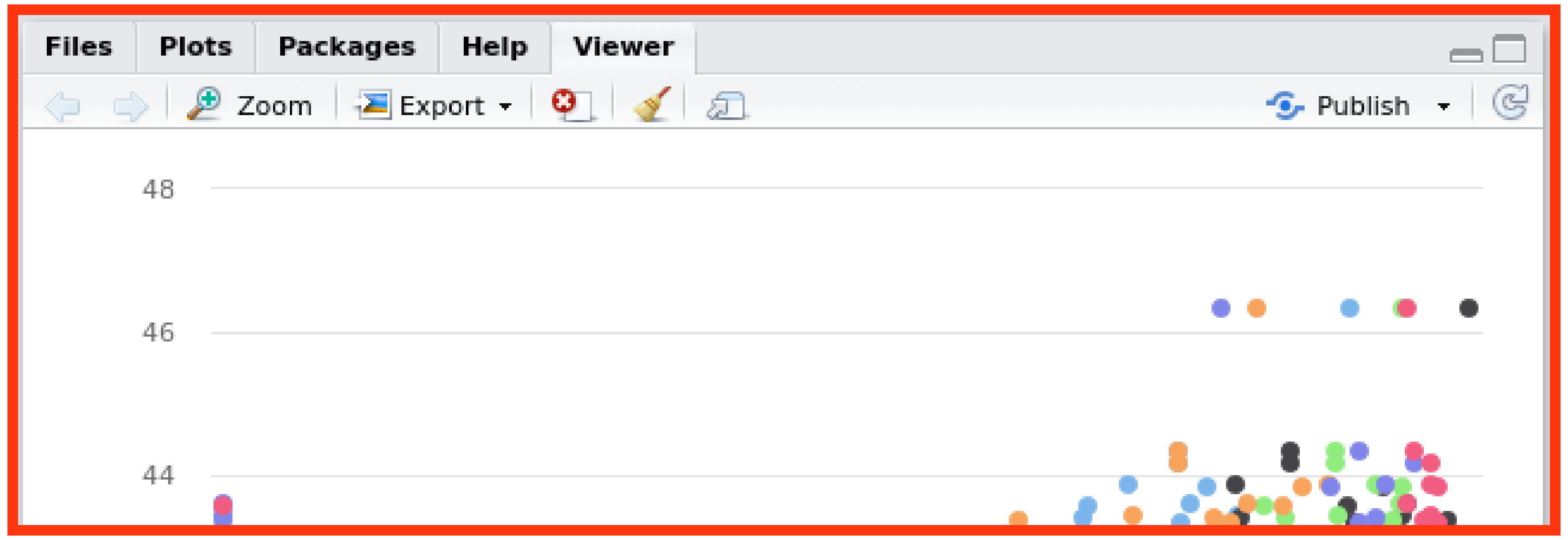
```
# A tibble: 6 x 4
# Groups:   variable [1]
    age variable value norm_value
  <dbl> <chr>    <dbl>      <dbl>
1    67 bmi       36.6      1.27
2    61 bmi       28.9      1
3    80 bmi       32.5      1.12
4    49 bmi       34.4      1.19
5    79 bmi       24        0.831
6    81 bmi       29        1.00
```

Construct a scatterplot using `hchart`

- To construct a scatterplot, we use the `hchart()` function and pass the **data**, **plot type** (`scatter`), and **aesthetics** to it as arguments

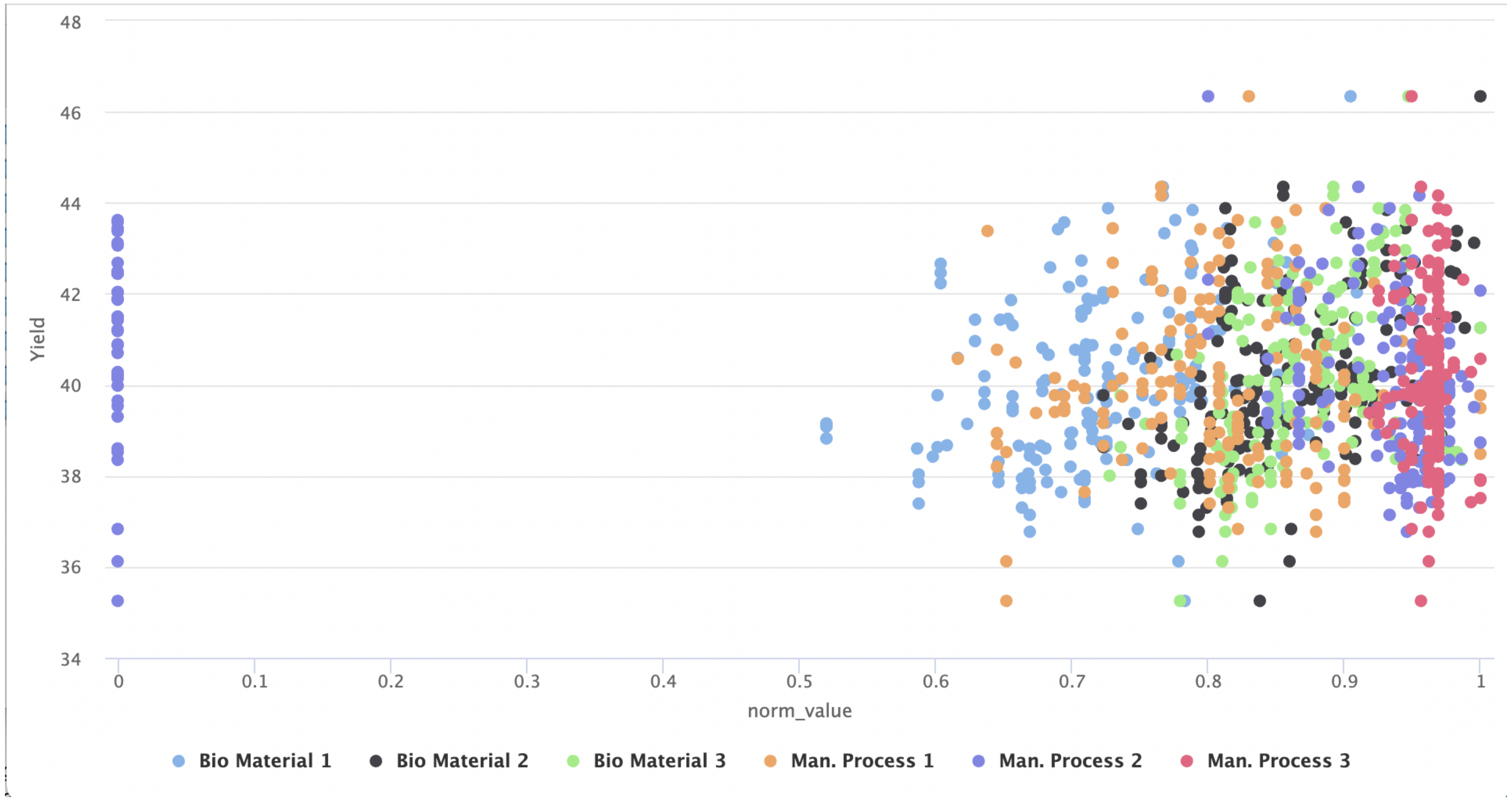
```
# Construct an interactive scatterplot.
scatter_interactive =                   #<- name the plot
  hchart(HDS_subset_long,              #<- set data
         "scatter",                     #<- plot type "scatter"
          hcaes(x = norm_value,         #<- set aesthetics to map x-axis
                y = age,               #<- set aesthetics to map y-axis
                group = variable))     #<- group by
```

- In R, interactive charts appear in the `Viewer` pane, right next to the `Help` tab



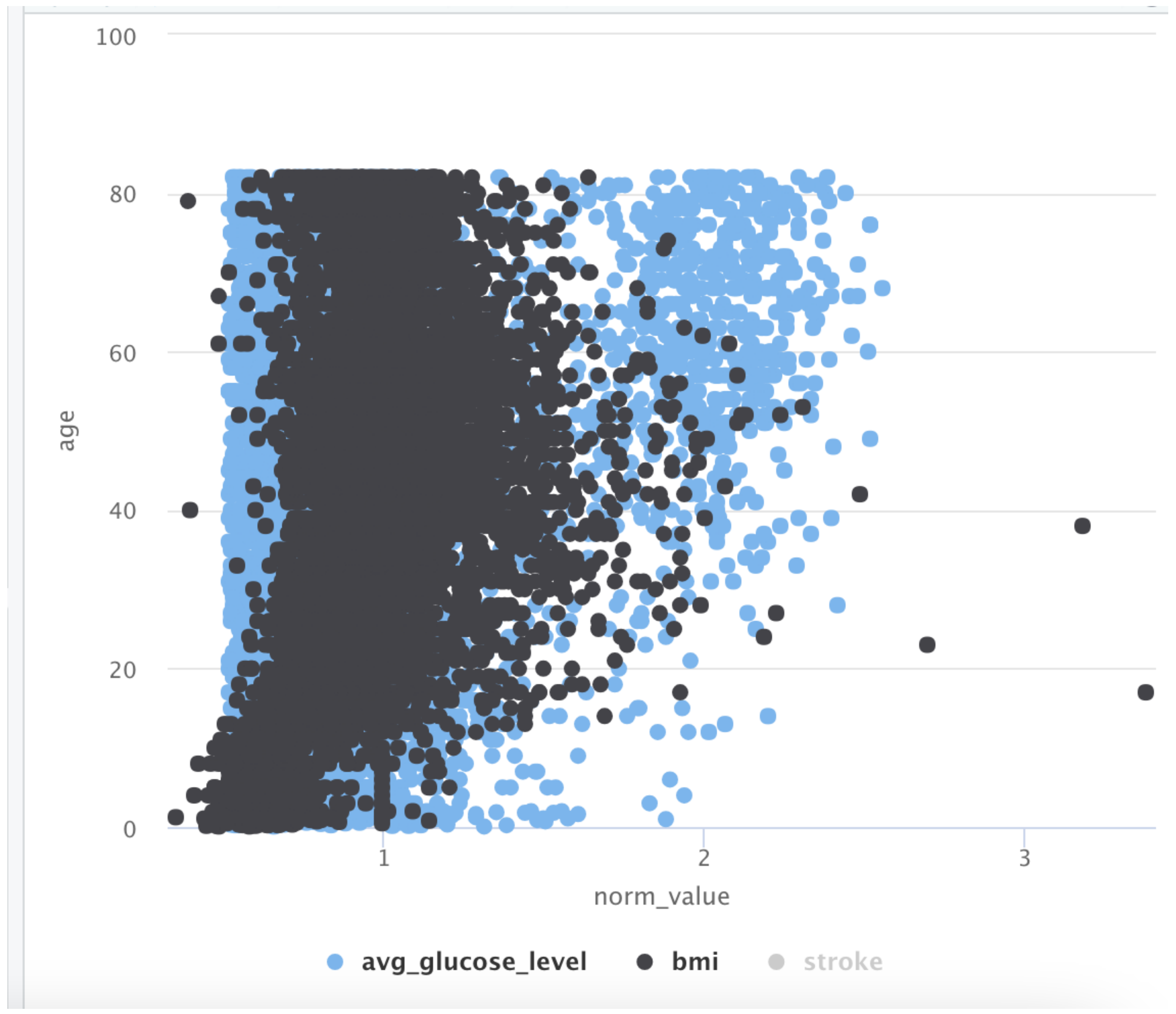
Construct a scatterplot using `hchart` (cont'd)

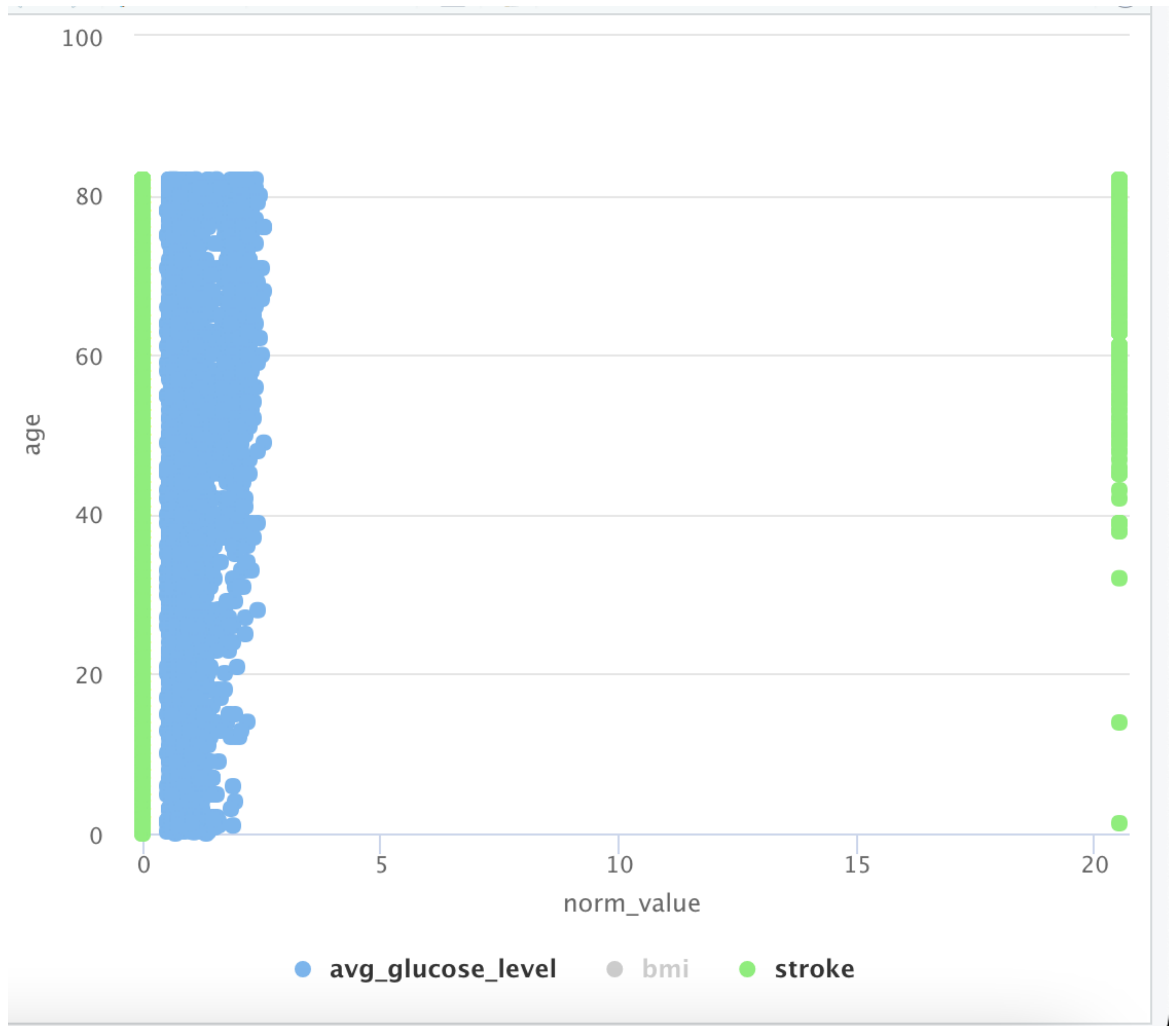
```
scatter_interactive
```



Selecting categories

- Every plotted category seen in the legend is a series in highcharter
- When `hchart()` detects more than one category, it auto-colors by series
- We can interactively select and de-select which series to display by clicking on the series names in the legend

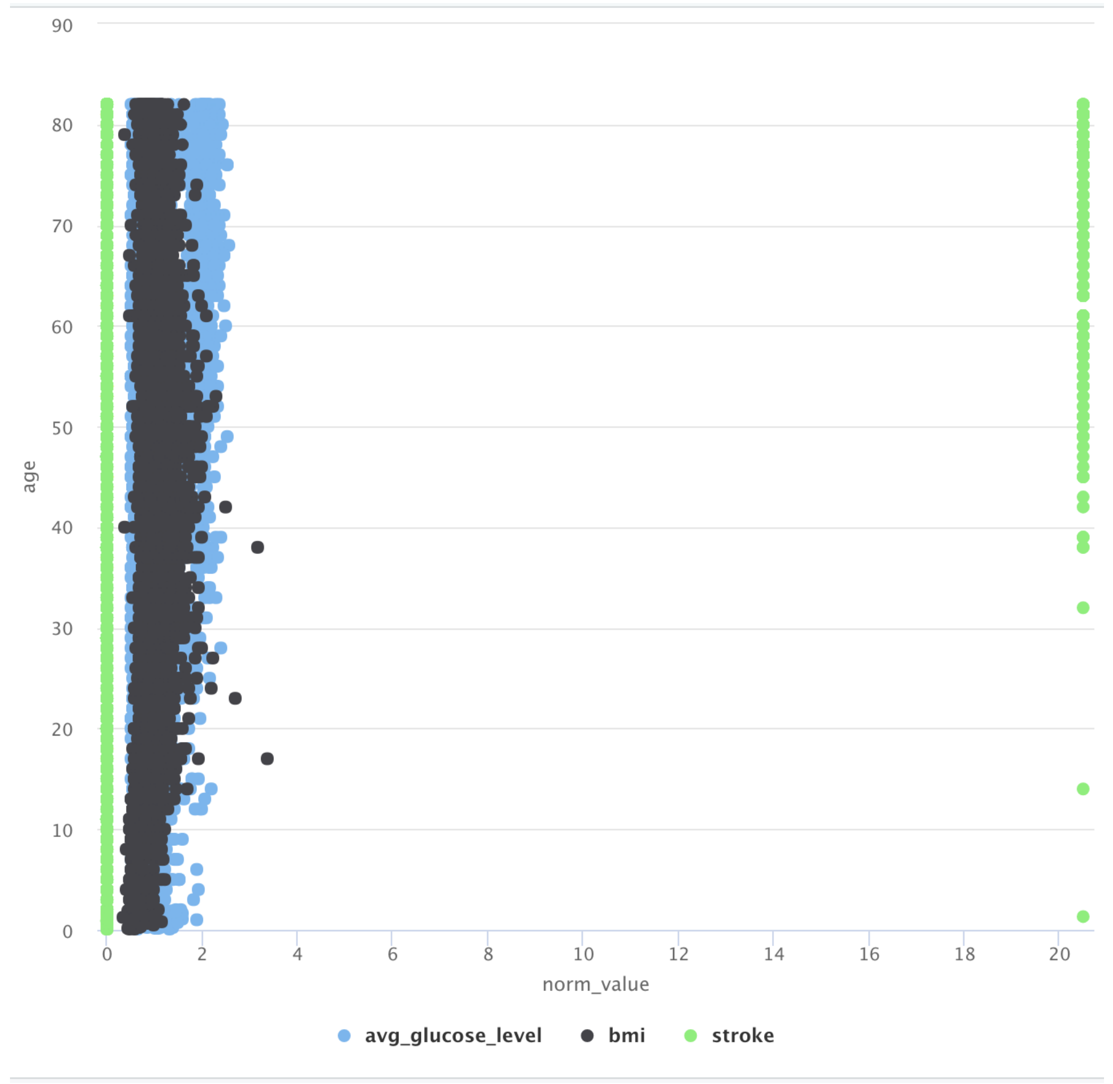




Customizing plots with the pipe operator

- You can add a new option or layer using the pipe operator (`%>%`)
- The `hc_chart()` function also controls global chart options like zoom, size, and theme
- Let's zoom in on our plot by passing the `zoomType` argument to `hc_chart()`
    - `xy` zoom allows zooming across both `x` and `y` axes

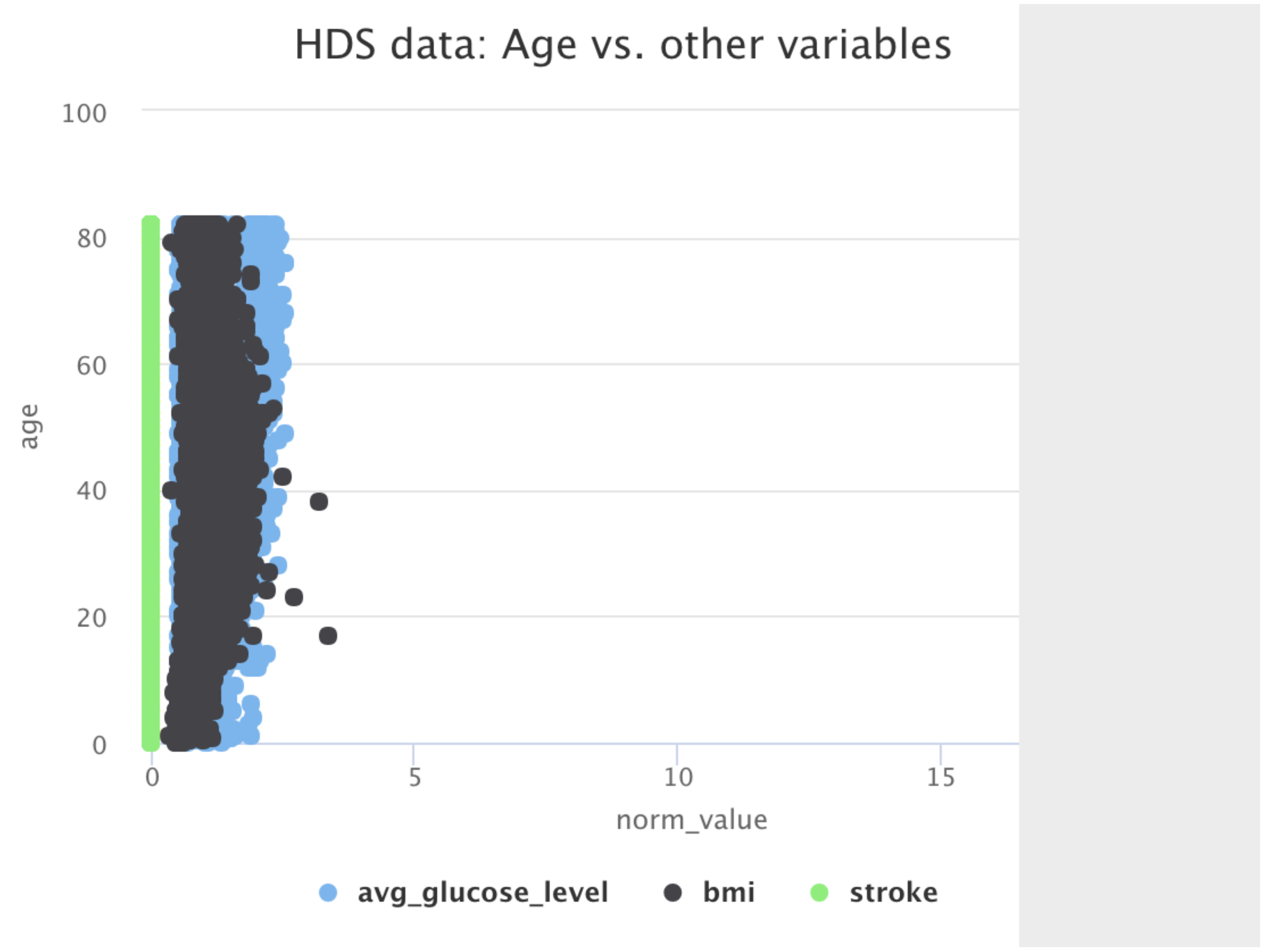
```
# Pipe chart options to original chart.

scatter_interactive = scatter_interactive %>%
  # Use chart options to specify zoom.
  hc_chart(zoomType = "xy")

scatter_interactive
```



Adding a title

- Use the `hc_title()` function to add a title to highcharter plots

```
# Pipe chart options to original chart.
scatter_interactive = scatter_interactive %>%
 # Add title to the plot.
 hc_title(text = "HDS data: Age vs. other
variables")

scatter_interactive
```



Knowledge check

Exercise

You are now ready to try tasks 1-4 in the Exercise for this topic

Module completion checklist

Objective	Complete
Install the highcharter package and discuss its application to build interactive visualizations	✔
Create a scatterplot using highcharter with tidy data	✔

Interactive plotting libraries: topic summary

In this part of the course, we have covered:

- Discovering different functions to build interactive visualization
- Visualizing data with `highcharter`

Congratulations on completing this module!