Cheatography

Data Visualization in R: ggvis continued Cheat Sheet

by shanly3011 via cheatography.com/20988/cs/3867/

ggvis & Group_by

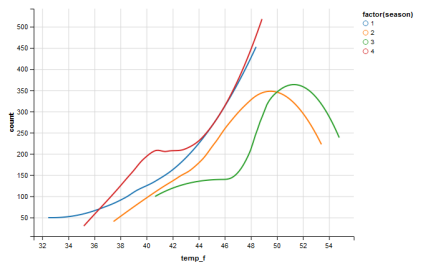
When these 2 are used in conjunction, we can create powerful visualizations.

Code:

```
train_tbl %>%
group_ by( season) %>%
ggvis( ~te mp_ f,~ count, stroke = ~facto r(s eason)) %>%
layer_ smo oths()
```

Here, season is a categorical variable. And we have grouped it and then used stroke to highlight the different seasons.

Output



In-Built plot types

1. `layer_points()`
2. `layer_ lines()`
3. `layer_ bars()`
4. `layer_ smo oths()`
5. `layer_ his tog rams()`

Most popular ones cited

Global Vs Local properties

A property that is set inside `ggvis()` is applied globally. While a property set inside `layer_ <ma rks >` is applied locally.

Local properties can override global properties when applicable.

Scale Types

Any visual property in the visualization can be adjusted with scale().

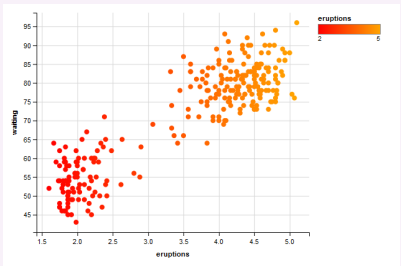
ggvis provides several different functions for creating scales:

```
scale_ dat eti me(), scale_ log - ical(), scale_ nom inal(), scale_ num eric(), scale_ sin - gular()
```

Code

```
faithful %>%
ggivs( ~er upt ion s,~ wai ting, fill = ~erupt ions) %>%
layer_ poi nts() %>%
scale_ num eri c("f ill ", range = c("r ed", " ora nge "))
```

Output



ggvis & interaction ()

We can also group data based on interaction of two or more variables.

`group_by()` creates unique groups for each distinct combination of values within the grouping variables.

`ungroup()` can remove the grouping information.

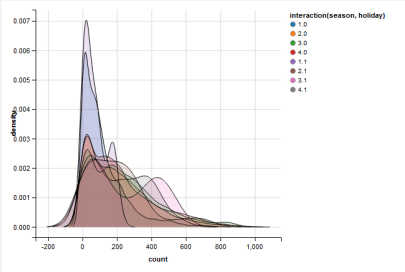
ggvis & interaction () (cont)

`intera ction()` can map the properties to unique combinations of the variables

Code:

```
train_tbl %>%
group_ by( sea son ,ho liday) %>%
ggvis( ~count, fill = ~inter - act ion (se aso n,h oli day)) %>%
layer_ den sit ies()
```

Output



Model Prediction

layer_model_predictions() plots the prediction line of a model fitted to the data.

```
layer_ mod el_ pre dic tio ns( - model = " lm")
```

Code:

```
faithful %>%
ggvis( ~er upt ion s,~ wai ting) %>%
layer_ poi nts (fill := " gre - en", fillOp acity := 0.5) %>%
layer_ mod el_ pre dic tio ns( - model = " lm", stroke := " - red ") %>%
layer_ smo oth s(s troke := " - sky blu e")
```


By **shanly3011**

cheatography.com/shanly3011/

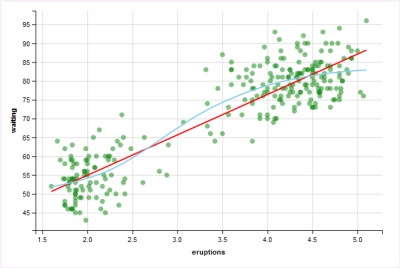
Published 10th April, 2015.

Last updated 12th May, 2016.

Data Visualization in R: ggvis continued Cheat Sheet

by shanly3011 via cheatography.com/20988/cs/3867/

Output



Interactive Plots

ggivs comes several widgets such as

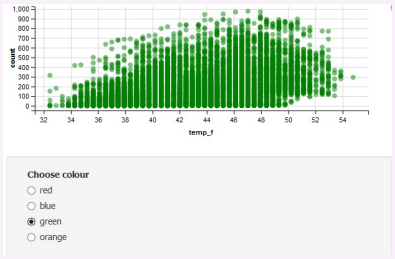
```
input_ che ckb ox(),
input_ che ckb oxg roup(),
input_ num eric(),
input_ rad iob utt ons(),
input_ sel ect(),
input_ sli der(), and input_ -
text().
```

label = "ABCD " , choices = c("red","black") - value = "black" - Used with input_text()

map = as.name used when we want to return variable names

Are the common arguments inside these functions.

Output



Legends & Axis

Axis

You can add axes with `add_axis()`

Syntax:

```
faithful %>%
ggvis( ~er upt ion s,~ wai ting)
%>%
add_ax is( " x", label = " Eru -
pti ons ", values = c(1,2, 3,4),
subdivide = 9, orient = top")
%>%
layer_ poi nts()
```

Legends

`ggvis` adds a legend for each property that is specified. To combine multiple legends into a single legend with common values, use a vector of property names.

```
add_le gend()
hide_l egend()
```

Syntax

```
faithful %>%
ggvis( ~wa iting, ~erupt ions,
opacity := 0.6,
fill = ~facto r(r oun d(e rup -
tio ns)), shape = ~facto r(r -
oun d(e rup tio ns)),
size = ~round (er upt ions)) %>%
layer_ poi nts() %>%
add_le gen d(c ("fi ll", " sha -
pe", " siz e"),
title = "~ duration (m)", values
= c(2, 3, 4, 5))
```

By **shanly3011**

cheatography.com/shanly3011/

Published 10th April, 2015.
Last updated 12th May, 2016.