

R integration: ggplot and other visualizations, aggregation and data sorting

Author: Lorenzo Conforti Andreoni - 29/10/2019

Introduction

I've been working with Qlik for the last four years and in the past few months I've been more and more involved into bringing R functionalities into my dashboards. The integration works well but there are clearly some limitations, especially when compared to other solutions on the market; Qlik in my opinion still has a significant edge in terms of performance and associative engine and, when combined with R, it brings unique functionalities and features.

The purpose of this document is to review such limitations and provide solutions with the aim of fully utilizing all R capabilities in Qlik Sense; there are Qlik extensions that provide some of these features but, although very well developed and implemented, they are limited to the type of analysis they can perform; also, they require a certain level of technical skills to adapt them to the bespoke needs that very often arise when working with R.

These are the 3 topics I will cover:

- ggplot and other visualizations: how to easily and effectively display in Qlik Sense any kind of R visualization without the need to develop bespoke extensions
- Data aggregation through concatenation: how using the aggregation function allows to pass to R more complex data frames
- Data sorting: how data is exchanged between Qlik Sense and R and how to handle sorting

Prerequisites

- Good knowledge of Qlik Sense
- Good knowledge of R
- Qlik Sense/R integration set up and working
- The following R packages installed (to be able to use the example dashboard):
 - o tidyR
 - o dplyr
 - o RCurl
 - o ggplot2
 - o corrplot
 - o rpart
 - o rattle
 - o forecast

- zoo
- xts
- scales
- forcats
- Media Box for Qlik Sense extension
- Qlik Visualization and Dashboard bundles

There are very good posts on how to set up a working Qlik Sense / R environment so please refer to those if you struggle with the installation of the required components.

Demo dashboard

I've collected some R visualization examples in the dashboard provided. This dashboard includes a number of dataset such as the England house price paid data and crime statistics; please note that, in order to reduce the size of the file, I've randomly selected house price transactions for the years provided; the crime data is complete, although summarized at a local area level. The data model is organized in a star schema with the postcodes table at the center of it; I've also loaded the iris dataset from R. Some visualizations are complex so, depending on the system, they might take a few seconds to render.

Please note: the visualizations are taken from other works I've been doing on UK national statistics so please do not reproduce them; if you are interested in knowing more about the analysis itself please get in touch with me.

ggplot and other visualizations

One of the drawbacks of the R integration is the inability to natively visualize plots; there are some great libraries in R (e.g. ggplot2) that give access to producing very effective visualizations.

Luckily, it is not too complex to visualize such plots in Qlik Sense; the only requirement is to install the Media Box extension (<https://github.com/stefanwalther/sense-media-box>).

The approach is then to:

1. pass the data to R
2. create a temporary png device (i.e. in memory)
3. plot the required chart
4. base 64 encode the temp png device
5. include the encoded data in an image html element within the Media Box extension with the following code:

```
='  
<div>  
  
</div>  
'
```

Plots will then be visualized as static images, which is what you would get from R. There are libraries and extensions that bring to Qlik Sense more interactive charts (like the Advanced Analytics Toolbox): these are great tools but they are limited to specific use cases; I've used them effectively in the past but I've been able to bring to Qlik Sense more complex and flexible visualizations with the method described here.

Having direct access to the HTML code allows also to resize the image so to have pleasing high resolution plots.

As with any R project, it is important to remember to remove the device after plotting it (`dev.off()`); also, I've included a script at the beginning of these visualizations to remove any device that might still be active.

Since the Media Box object is effectively a dimensionless object, you might want to pass to R aggregated data by following the approach discussed further on in this document; this might be recommended when the dataset is very large.

R plots visualization examples



Ggplot and other visualisations

To be able to display any plot in Qlik Sense, the only requirement is to install the Media Box extension.

The approach is then to:

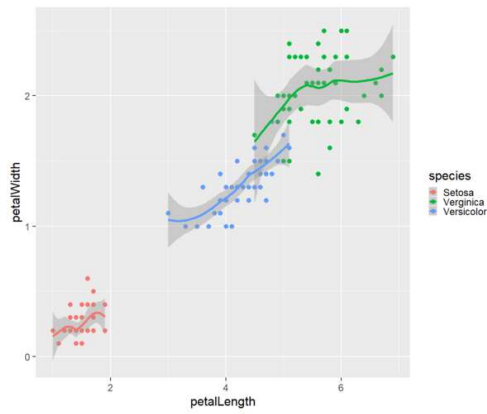
- pass the data to R
- create a in memory temporary png device
- plot the required chart
- base 64 encode the temp png device
- include the encoded data in a image html element within the Media Box extension

HTML code used within the Media Box object:

```
='  
<div>  
  
</div>  
'
```

Also, by varying the resolution of the png device in R and in the img html element, it's possible to display charts with a higher definition

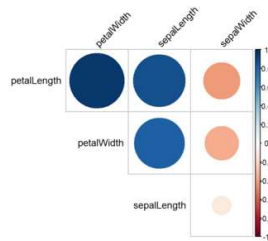
Petal width / petal length



Species

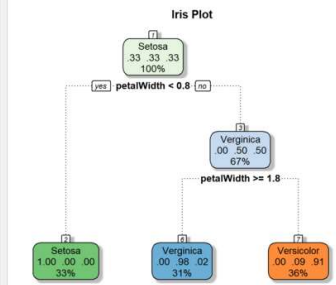
- Setosa
- Virginica
- Versicolour

Correlation matrix



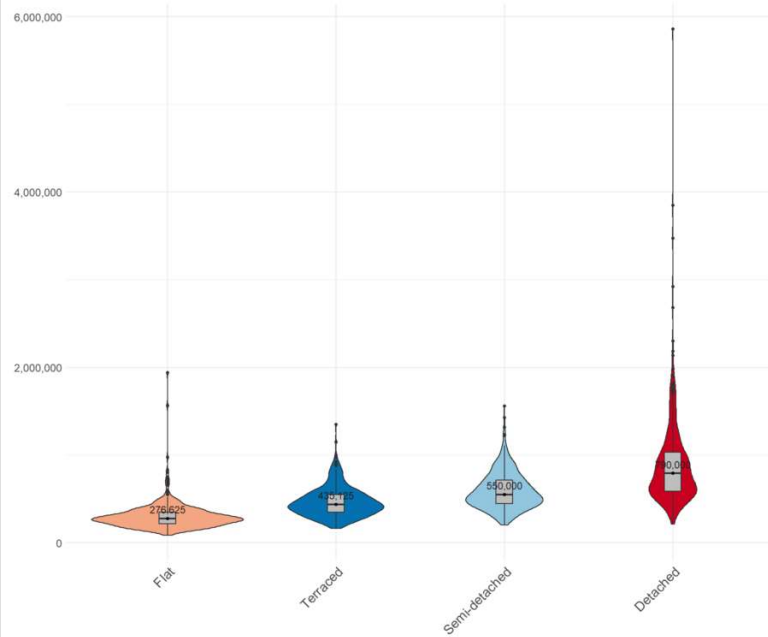
Classification tree

Select two or more species for the chart to visualize



Plot distribution of sales by property type; median prices displayed

Number of properties visualised: 1,629



ggplot2 visualisation

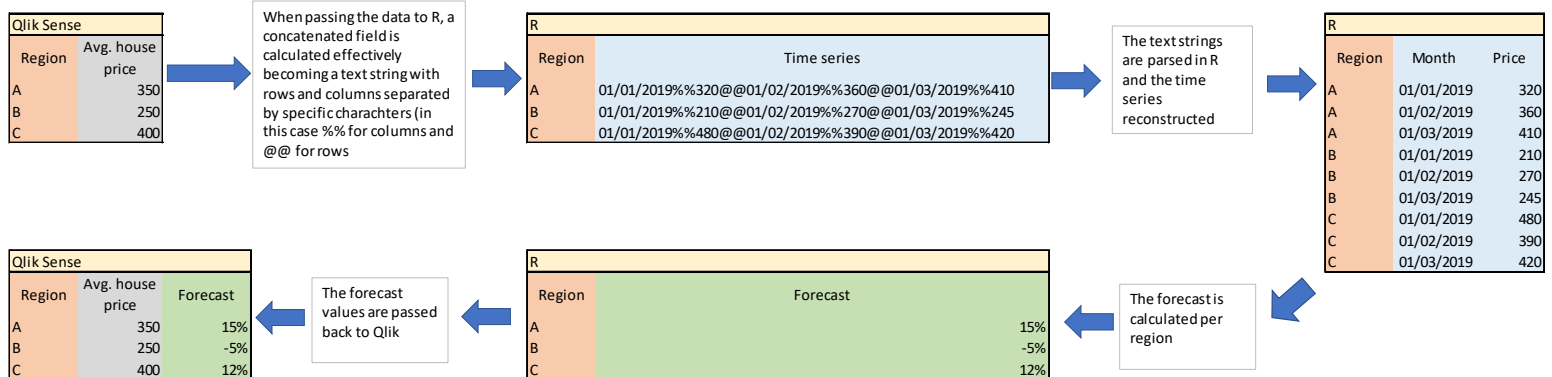
Data aggregation through concatenation

The main data limitation I've found with the R integration is that it does not natively support passing lists or more complex data frames; there are two scenarios here: chart objects (i.e. when dimensions are defined) and dimensionless objects (like a text box).

Chart objects

When working on a chart object, the data frame passed to R has the same length as the chart: this is a necessity but at the same time a limitation because all the functions in R will be limited to one data point per variable per observation.

Imagine you have a table with the list of regions with average house prices; you also want to forecast future price changes per region and have that value displayed next to the average price. This means that, for each region, you would need to pass to R all the historical prices; this poses a significant challenge because Qlik can only send a data frame to R of the same length of the chart object (i.e. only one row per region) and it expects back a vector of that same length. The solution is to encode, with a concatenation applied to an "aggr" function, multiple values per row which are then reformed into a data frame by R. Forecasting can then be calculated for each region and the result joined back to the original data frame (to keep the correct sorting and length) and passed back to Qlik Sense.

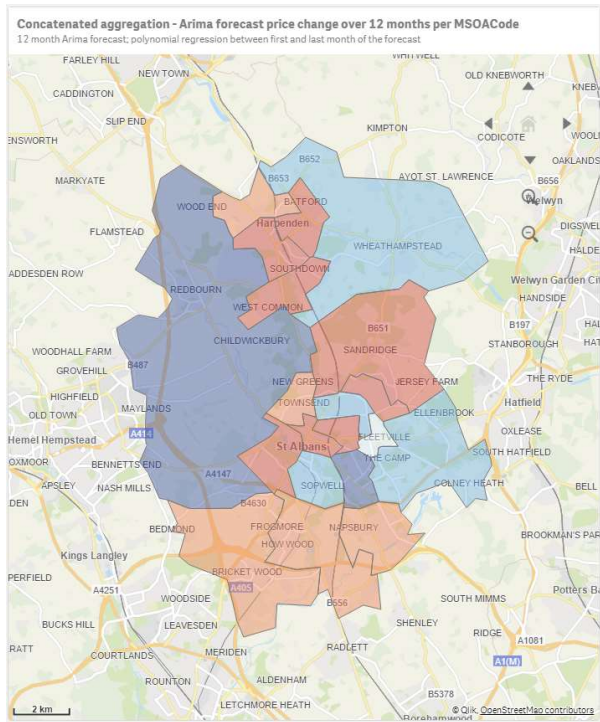


Dimensionless objects

An extension of the above scenario is the situation in which we want to use aggregation functions in dimensionless objects such as a text box. In these types of objects, since we cannot define dimensions, we would normally use the R.ScriptAggr function and pass a flat table (straight table in QlikView terminology) to R. This is perfectly acceptable, but it might become challenging when the data set in question is very large and we want just to pass few aggregated values to R. The R.ScriptAggr does not allow aggregation (rightly so as we don't have means to distinguish between dimensions and measures in its parameters); the solution in this case is to concatenate the results of an "aggr" function, and, through dollar sign expansion, have it as a text input to the function. We will then parse it in R and reconstruct the data frame accordingly.

Whilst this approach might not be very elegant, it is, on the other side, very effective and easy to manage. I've tested it extensively and passed hundreds of thousands of values to R without any issues; it's a workaround that opens up important R opportunities.

Data aggregation in a chart object: forecasted (Arima) price change per area; red color means price in the next 12 months is expected to increase, blue to decrease



Data aggregation in a dimensionless object: LSOA dimension aggregated data is passed to R reducing the length of the data frame by about 95% compared to passing a flat table with all the transactions in question and doing the aggregation in R

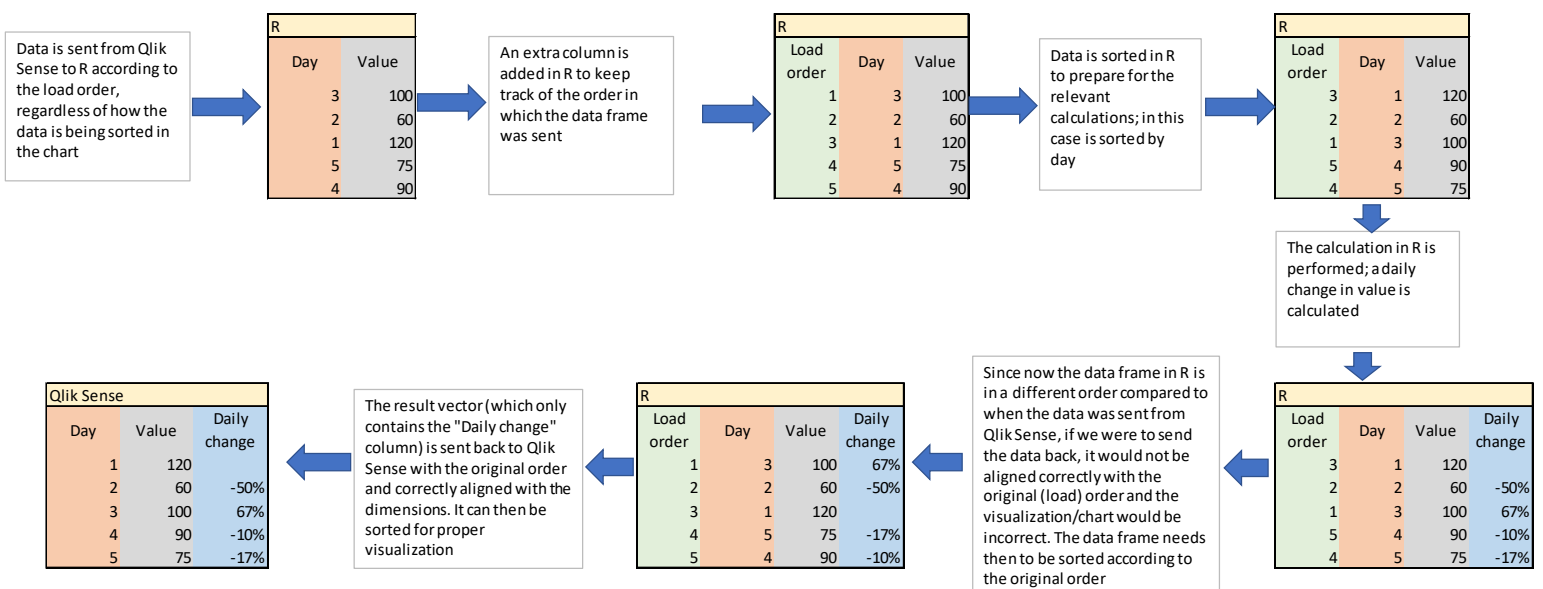
Data sorting

One of the first challenges to face when integrating R with Qlik Sense is to address data sorting; this is particularly important when working on a chart object: a number of elements might be visualized so we need to guarantee correct alignment between the data sent from Qlik Sense and the vector that is returned by R.

Generally data is sent to R based on the load order; this could be manipulated in the load script but I consider this approach risky as we might not have the possibility to amend the load script; also, different visualization of the same data might have different sorting requirements.

The examples below and in the dashboard show how this can be achieved in R; the critical aspect is that, when the data frame is passed to R, the original sort order is encoded in an extra column ("Load order" in the example). All the necessary calculations are performed in R and, just before sending back the results to Qlik Sense, the data frame is sorted back to the original order.

The example below is very basic but in the example dashboard there are more complex implementations, especially when dealing with time series.



It is also important to consider that the length of the resulting vector we are trying to pass back to Qlik Sense might be different (for example we might have removed in R observations with null values). Qlik will accept the vector and display the resulting data; we clearly run the risk of displaying the information in the chart incorrectly, skipping the necessary gaps. In this case, the approach is similar to what has

been described above with the difference that, instead of simply re-sorting the data, the result of the calculation is joined to the original vector by the "Load order" variable. This will take care of the alignment and insert gaps in the result vector. The time series charts in the example dashboard make use of this approach.

Example of how sorting the data in R solves the issue of a load order sorting not aligned with the analysis we intend to perform. If the data was not sorted in R we would be visualizing incorrectly the information as per the last column on the right.

Day	Value	Daily change - correctly sorted (from R)	Load order (from R)	Daily change - not re-sorted (from R)
Totals	1037	-	-	-
1	33	-	3	-65%
2	34	3%	15	50%
3	12	-65%	14	69%
4	65	442%	5	-52%
5	31	-52%	13	39%
6	51	65%	2	3%
7	31	-39%	6	65%
8	75	142%	9	-20%
9	60	-20%	18	7%
10	76	27%	7	-39%
11	92	21%	10	27%
12	28	-70%	11	21%
13	39	39%	17	168%
14	66	69%	12	-70%
15	99	50%	19	-29%
16	28	-72%	20	-91%
17	75	168%	16	-72%
18	80	7%	1	-
19	57	-29%	4	442%
20	5	-91%	8	142%