R Training Notes

Lesson 7

Stefanie Molin

April 26, 2017

I. [redacted]

You will be happy to know [redacted] has its own R package ([redacted]) that makes it easier to query our databases, make [redacted]-themed graphs, generate PowerPoints, and send emails on the [redacted] network. The idea behind the package is to curate a collection of functions that everyone at [redacted] can use without having to write their own code for the same task, like querying Vertica. It is constantly being improved, so this document will just be able to highlight the current state of the package.

A. Installing [redacted]

[redacted] lives on gitlab, not CRAN, so, we need to use a function from the `devtools` package to allow us to install this (`install.packages()` won't work in this case).

- Follow the instructions here: [redacted]
- Any errors that arise naming a package mean that you need to install that package (i.e. run `install.packages("package")`) and try to install [redacted] again.

B. [redacted] Functions

Now that we have the [redacted] package, let's see what functions we have at our disposal. You can see the functions available in a package using: `ls("package:<package_name>")`. If you want to see the arguments for a particular function, but don't need the full help page, use `args(<function_name>)`.

```
# load [redacted]
library([redacted])

# show functions in [redacted]
ls("package:[redacted]")
```

```
##  [1] "add_ts_features"              "aggregate_data"
##  [3] "db_ip_country"                "draw_chord"
##  [5] "draw_marimekko"               "draw_pie"
##  [7] "draw_sankey"                  "draw_waterfall"
##  [9] "file_merger"                  "ftp_downloader"
## [11] "get_ftp_file_vector"          "ip_location_lookup"
## [13] "ppt_[redacted]_template"      "pull_data_sql"
## [15] "pull_data_vertica"            "pull_data_vertica_time_batches"
## [17] "put_quotes"                   "read_query_from_file"
## [19] "scale_color_[redacted]"       "scale_fill_[redacted]"
## [21] "send_email"                   "set_[redacted]_psw"
## [23] "theme_[redacted]_default"     "update_[redacted]"
```

```
# see arguments for [redacted]
args(send_email)
```

```
## function (username = Sys.info()[[6]], recipient = paste0(Sys.info()[[6]],
##     "@[redacted]"), attachment_files = NULL, password = NULL,
##     body = " ", verbose = FALSE, extra_css = NULL, ...)
## NULL
```

C. [redacted] Use Case

Now, we are going to build on our R knowledge and learn how to use the [redacted] package for various tasks in conjunction with the packages we covered in prior lessons.

1. Querying Databases

Let's pull last 30 days of client data for [redacted] We are going to use [redacted] to read the query in from a file and query Vertica. Then we use `dplyr` to pivot our data into [redacted]`_pivot` which you should remember from the other lessons; the only difference here is how we got the query and the data from Vertica. *Note that* [redacted] *also has a function to query the SQL server.*

```
# load dplyr
library(dplyr)

# get date for 30 days ago
startDate <- Sys.Date() - 30

# [redacted] client_id
[redacted]_client_id <- 4624

# read query from a file
query <- read_query_from_file("client_stats_query.sql")

# look at query using stringr to clean up whitespaces for printing
library(stringr)
cat(str_replace_all(query, "\\s{2,}", " "))
```

```
## SELECT * FROM [redacted] WHERE day >= '%s' AND client_id = %s
```

```
# query for last 30 days of client stats for [redacted]
# (select cluster in function call)
# here we provide the password but if you don't you will be prompted to enter it
[redacted] <- pull_data_vertica(sprintf(query, startDate, [redacted]_client_id), cluster = "pa4",
                                username = username, password = password)
```

```
# pivot dataframe and filter down to 25 days
[redacted]_pivot <-[redacted] %>%
  select(day, displays, clicks, revenue, pc_conv = post_click_conversions,
         pc_sales = post_click_sales) %>%
  filter(as.Date(day) >= Sys.Date() - 25) %>%
  group_by(day) %>%
  summarize(total_clicks = sum(clicks, na.rm = TRUE),
            total_imps = sum(displays, na.rm = TRUE),
            spend = sum(revenue, na.rm = TRUE),
            conv = sum(pc_conv, na.rm = TRUE)) %>%
  mutate(ctr = total_clicks/total_imps, cpc = spend/total_clicks) %>%
  arrange(day)

# see first few rows
head([redacted]_pivot)
```

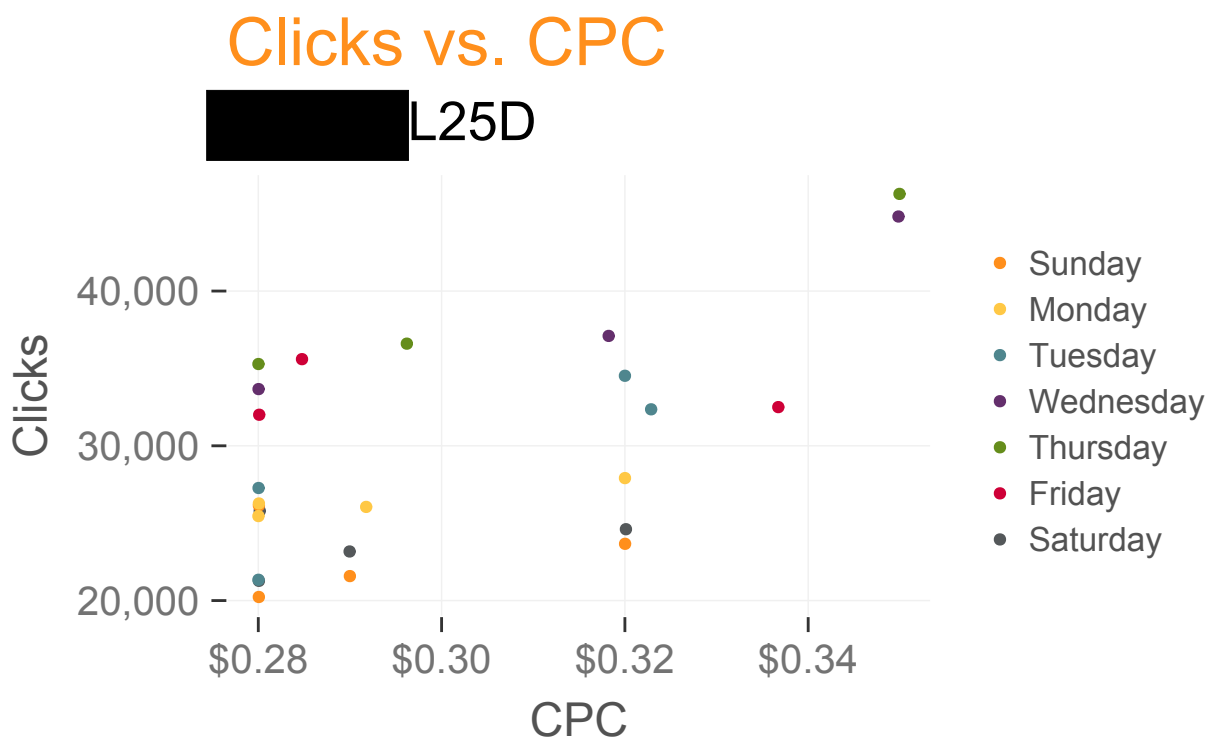
```
## # A tibble: 6 × 7
##          day total_clicks total_imps   spend  conv        ctr       cpc
##        <chr>        <dbl>      <dbl>   <dbl> <dbl>      <dbl>     <dbl>
## 1 2017-04-01 [redacted]
## 2 2017-04-02
## 3 2017-04-03
## 4 2017-04-04
## 5 2017-04-05
## 6 2017-04-06
```

2. [redacted]-themed ggplot2 Graphs

[redacted] has functions for easily formatting your ggplot2 graphs to [redacted] colors. `scale_color[redacted]()` and `scale_fill_[redacted]()` will format the colors used in the `col` and `fill` aesthetics, respectively. `theme_[redacted]_default()` will handle the rest. *Note that to use this in an RMarkdown PDF you will need to add* ***`dev = "cairo_pdf"`*** *to the* ***`knitr`*** *options at the top of the .Rmd file.*

```
# clicks vs. CPC for [redacted] colored by days of the week

# load library
library(ggplot2)

# define aesthetics
(plot <- [redacted]_pivot %>%
  ggplot(aes(x = cpc, y = total_clicks,
             col = factor(format(as.Date([redacted]_pivot$day), "%A"),
                          levels = c("Sunday", "Monday", "Tuesday",
                                     "Wednesday", "Thursday", "Friday",
                                     "Saturday")))) +
  # add points
  geom_point() +

  # format labels
  scale_x_continuous("CPC", labels = scales::dollar) +
  scale_y_continuous("Clicks", labels = scales::comma) +

  # add title
  ggtitle("Clicks vs. CPC", subtitle = "[redacted] L25D") +
```

```
# [redacted] themes and colors
[redacted]::scale_color_[redacted]() +
[redacted]::theme_[redacted]_default())
```



3. Generate [redacted]-themed PowerPoint Decks

[redacted] provides a function `ppt_[redacted]_template()` which will instantiate a PowerPoint object (using the [redacted] template slide) which you can use with the `ReporteRs` package to fill with the content of your choosing.

```
# create ppt object
ppt <- [redacted]::ppt_[redacted]_template()

# load ReporteRs library
library(ReporteRs)

# add slides and content
# add title slide
ppt <- addSlide(ppt, "main_title")
ppt <- addTitle(ppt, "[redacted] Performance Last 25 Days", width = 8)

# add slide for graph
ppt <- ppt %>%
  addSlide("text_and_xlarge_image_horizontal") %>%
  addTitle("Clicks vs. CPC") %>%
  addParagraph("Here is clicks vs. CPC for the last 25 days.") %>%
  addPlot(fun = print, x = plot)
```

```
# add slide for metrics by day
ppt <- ppt %>%
  addSlide("blank_slide") %>%
  addTitle("Daily Stats") %>%
  addFlexTable(FlexTable([redacted]_pivot))

# add an end slide
ppt <- addSlide(ppt, "end_slide")

# save the ppt
filename <- "[redacted]Metrics Last 25 Days.pptx"
writeDoc(ppt, file = filename)
```

Note that for each slide you create you will need to know the name of that particular layout. You can cycle through the options using the below code:

```
## not run (this is how you see what layouts are available)
# view available slide layouts (by name)
slide.layouts(ppt)

# see how the available slide layouts look
layouts <- slide.layouts(ppt)
for(i in layouts){
    slide.layouts([redacted]_ppt, i)
    title(sub = i)
    if(interactive()) readline(prompt = "show next slide layout")
}
##
```

4. Sending Emails

You can use the `send_email()` function to send emails (and attachments) using [redacted] credentials.

```
# send the email with the ppt
[redacted]::send_email(username = username, recipient = "[redacted]",
                       attachment_files = filename, password = password,
                       subject = "[redacted]PPT",
                       body = "[redacted]PPT from [redacted]Training.")
```