Code ▾

Generating Automatic Analysis in R Markdown

Hide

```
library(pacman)
p_load(knitr,rmarkdown,broom,ggplot2)
```

R Markdown allows one to quickly prototype and automate analysis and then embed it in a document. Recently, I ran into a problem: If I repeat a set of summaries many different data-sets and want to summarize the results, how can I iterate through them without a large amount of copying and pasting?

I was surprised to find relatively few guides on this topic on Stack Overflow or Reddit. The best example I could find was located here (https://stackoverflow.com/questions/21729415/generate-dynamic-r-markdown-blocks).

I've decided to take the post I linked above and provide a more involved example based on my small project from work.

For this example I'll perform some summary tests and plots based on the mtcars data-set:

1. Generate regression models for mpg.
2. Create a function to summarize the results (a summary table and a residual plot in this case).
3. Generate a list models and output them automatically (no copying or pasting).

Let's begin!

Below, I've written functions to generate the formulas, plots, and regression summaries:

Hide

```
gen_formulas <- function(dep,indep){as.formula(paste(dep,paste(indep,collapse = "+"),sep = "~"))}

reg_plot <- function(dep,indep,dat){  return(ggplot(dat,aes_string(x = dep,y = indep))+geom_point()+stat_smooth(method = "lm",col = "blue")+theme_bw())}

gen_reg_summaries <- function(dep,indep,dat){
  for_reg <- gen_formulas(dep = dep,indep = indep)
  temp_reg  <- lm(data = dat,formula = for_reg)
  reg_summary <- tidy(temp_reg)
  ret_plot <- reg_plot(dat = dat,dep = dep,indep = indep)
  return(list("Reg" = temp_reg,
             "Summary" = reg_summary,
             "Plot" = ret_plot))}
```

The regression results are then created as a list for each model. These are the values we'll be returning for our report:

Hide

```
formula_info <- c("disp","drat","wt","qsec")

regression_results <- lapply(formula_info, function(x){gen_reg_summaries(dep = "hp",indep = x,dat = mtcars) })
names(regression_results) <- formula_info
```

To avoid the pain of repeatedly copying the results that we will write functions to generate them dynamically.

To do this, we apply the knight_expand function from the knitr package. It's a sort of "meta markdown" function which will allow us to generate the output we seek repeatedly in a loop. In this case, we want to go through each of the results we have so far, and display the table and the plot.

Simple Example: A single a summary of the regression using HPI:

The code below is fairly simple: We use the knit_expand function. The argument is a text string consisting of a header and a reference to the summary table. We mark any pieces of code with two curly-braces:

Hide

```
gen_knit_text <- function(num){
  header <- paste('## Regression: {{names(regression_results)[',num,']}}',sep = "")
  smry <- paste('{{regression_results[[',num,']]$Summary}}',sep = "")
  plot <- paste(' ```{r, fig.width = 10 }\n{{regression_results[[',num,']]$Plot}}\n```',sep = "")
  res <- paste(paste(header,smry,plot,sep = "\n"),"\n\n" ,sep = "")
  return(res)}
```

Now, all we need to do to generate the output is a single call to sapply which will create all of the analysis we were looking for:

Hide

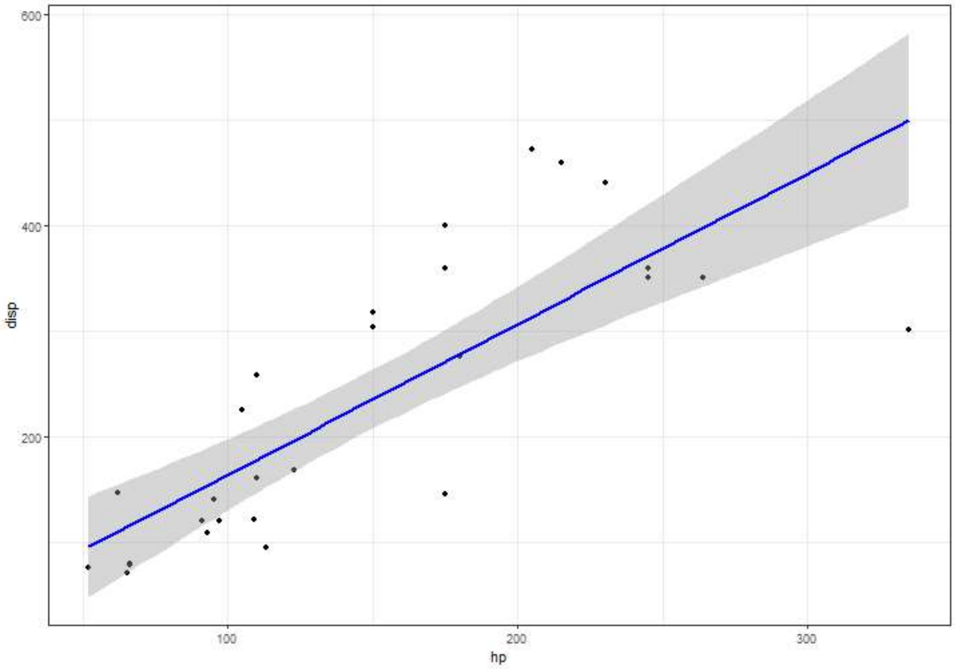
```
text_to_knit <- sapply(X = 1:length(regression_results),FUN = function(x){ knit(text = gen_knit_text(x)) })
```

Regression: disp

term <chr>	estimate <dbl>	std.error <dbl>	statistic <dbl>	p.value <dbl>
(Intercept)	45.7345322	16.12893522	2.835558	8.113909e-03
disp	0.4375526	0.06180015	7.080122	7.142679e-08

2 rows

```
r  list(mpg = c(21, 21, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, 17.8, 16.4, 17.3, 15.2, 10.4, 10.4, 14.7, 32.4, 30.4, 33.9, 21.5, 15.5, 1
```



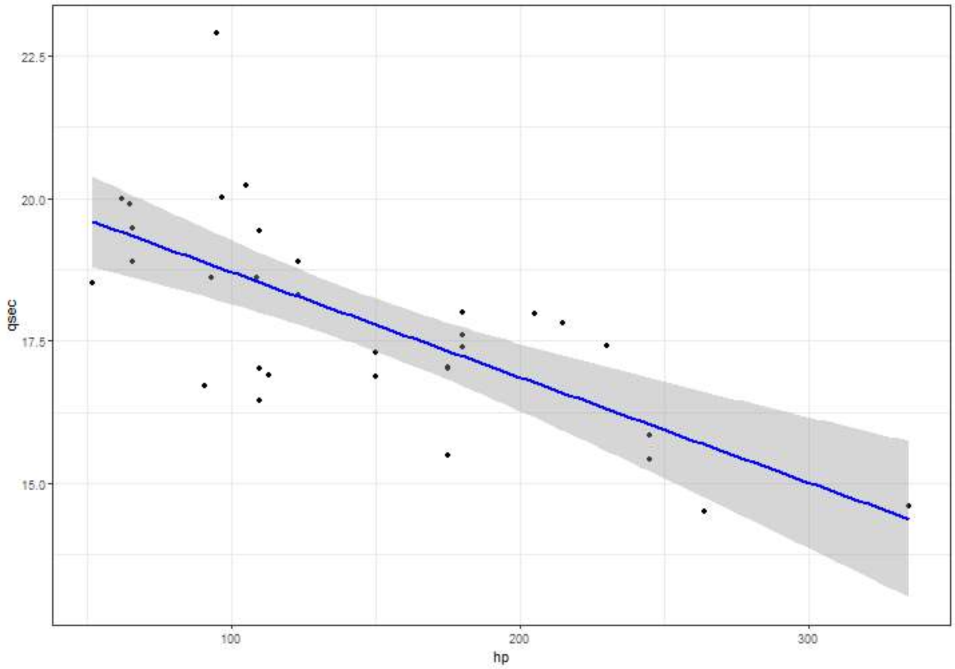
plot of chunk unnamed-chunk-5

Regression: drat

term <chr>	estimate <dbl>	std.error <dbl>	statistic <dbl>	p.value <dbl>
(Intercept)	353.65253	76.04873	4.650341	6.242031e-05
drat	-57.54523	20.92205	-2.750459	9.988772e-03

2 rows

```
r  list(mpg = c(21, 21, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, 17.8, 16.4, 17.3, 15.2, 10.4, 10.4, 14.7, 32.4, 30.4, 33.9, 21.5, 15.5, 1
```



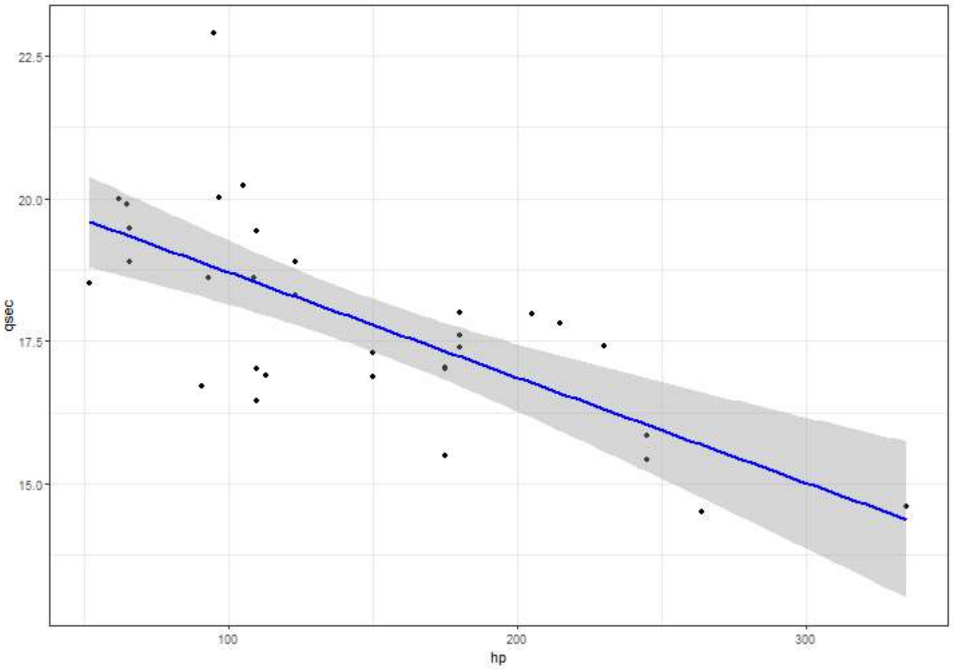
plot of chunk unnamed-chunk-1

Regression: wt

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	-1.820922	32.32462	-0.05633236	9.554506e-01
wt	46.160050	9.62530	4.79569988	4.145827e-05

2 rows

```
r  list(mpg = c(21, 21, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, 17.8, 16.4, 17.3, 15.2, 10.4, 10.4, 14.7, 32.4, 30.4, 33.9, 21.5, 15.5, 1
```



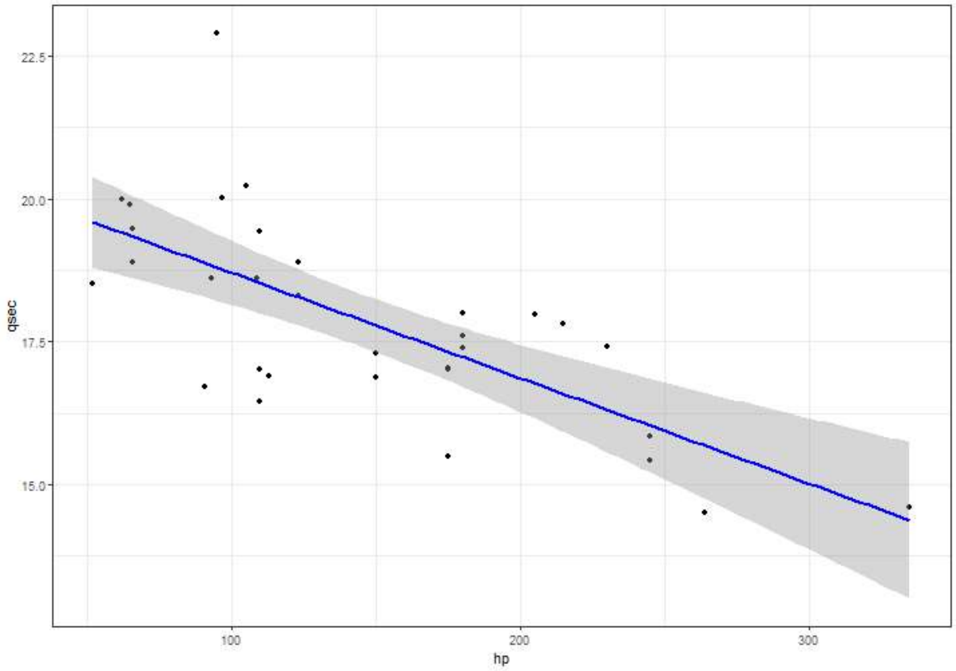
plot of chunk unnamed-chunk-1

Regression: qsec

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	631.70375	88.699525	7.121839	6.382739e-08
qsec	-27.17368	4.945556	-5.494565	5.766253e-06

2 rows

```
r  list(mpg = c(21, 21, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, 17.8, 16.4, 17.3, 15.2, 10.4, 10.4, 14.7, 32.4, 30.4, 33.9, 21.5, 15.5, 1
```



plot of chunk unnamed-chunk-1