Intro to markdown

Ryan

October 1, 2015

# Markdown Introduction

Markdown is a lightweight markup language with plain text formatting syntax designed so that it can be converted to HTML -- [wikipedia](https://en.wikipedia.org/wiki/Markdown)

Markdown uses different symbols to mark headers:

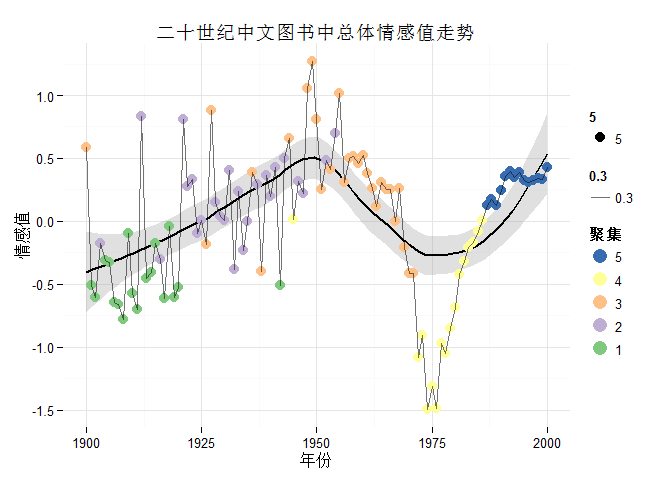
# Header Level 1

## Header Level 2

### Header Level 3

make things **BOLD** or *italic*

create a link  
[This is a link](http://www.example.com)

add a picture  


add a footnote[[1]](#footnote-28)

add a code block

@requires\_authorization  
def somefunc(param1='', param2=0):  
 '''A docstring'''  
 if param1 > param2: # interesting  
 print 'Greater'  
 return (param2 - param1 + 1) or None  
class SomeClass:  
 pass  
>>> message = '''interpreter  
... prompt'''

a quote

常识是十八岁之前在头脑中所铺下的偏见层 -- 忘记是谁说的了

My favourite LaTeX Formula

* Inline formula example: $\Gamma(n) = (n-1)!\quad\forall n\in\mathbb N$
* Block formula example:

Table made easy

|  |  |  |
| --- | --- | --- |
| Item | Value | Qty |
| Computer | 1600 USD | 5 |
| Phone | 12 USD | 12 |
| Pipe | 1 USD | 234 |

## See Markdown Quick Reference for more

So what R Markdown have to offer?

inline r code evaluation

The average miles per gallon is 20.090625 in our dataset

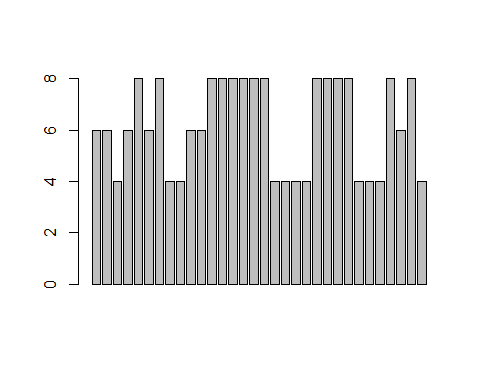
or r code chunck

summary(mtcars$mpg)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 10.40 15.42 19.20 20.09 22.80 33.90

plot

barplot(mtcars$cyl)



try knit this document to generate documents of the format that you want  
...you need to install it first

## more chuck options

R code not evaluated , setting eval = F

install.packages("rmarkdown")

Hide the r code but display the result using echo = F

## [1] 0.3333333

Hide only part of the code echo = c()

print(a/b)

## [1] 0.3333333

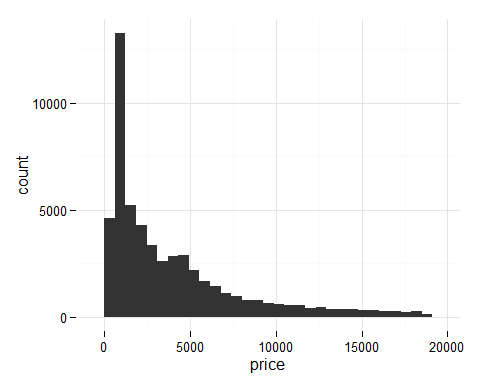
print("Can you see this?")

## [1] "Can you see this?"

Message by default will shown

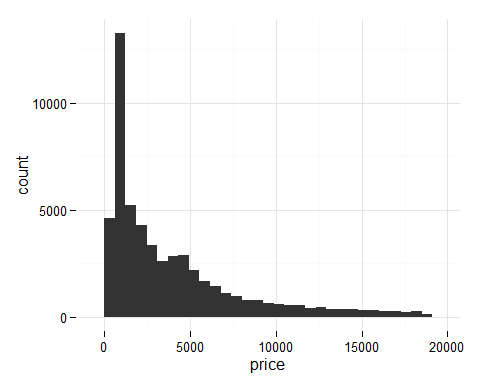
qplot(x = price, data = diamonds)

## stat\_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.



suppress by setting message = F

qplot(x = price, data = diamonds)

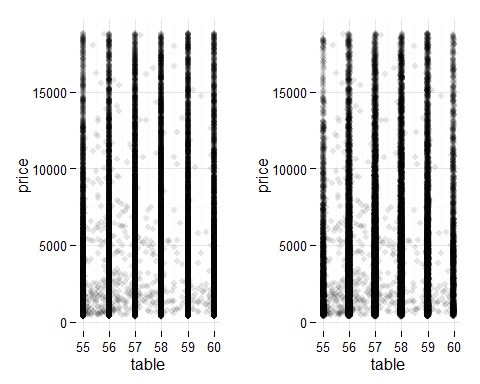


there could be warnings as well...

p1 <- ggplot(aes(x= table , y = price), data = diamonds) +   
 geom\_point(alpha = 0.1) +   
 xlim(55,60)  
p2 <- ggplot(aes(x= table , y = price), data = diamonds) +   
 geom\_jitter(alpha = 0.1) +  
 xlim(55,60)  
grid.arrange(p1,p2, ncol = 2)

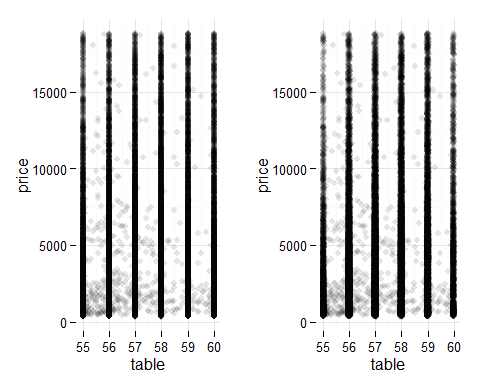
## Warning: Removed 8421 rows containing missing values (geom\_point).

## Warning: Removed 13768 rows containing missing values (geom\_point).



get rid of it by setting waring = F

p1 <- ggplot(aes(x= table , y = price), data = diamonds) +   
 geom\_point(alpha = 0.1) +   
 xlim(55,60)  
p2 <- ggplot(aes(x= table , y = price), data = diamonds) +   
 geom\_jitter(alpha = 0.1) +  
 xlim(55,60)  
grid.arrange(p1,p2, ncol = 2)



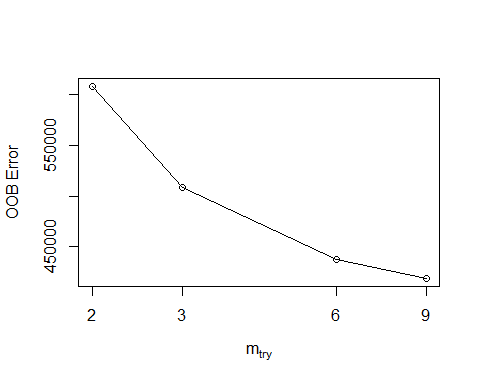
You can cache the chunk that requires large amount of computation by cache = T  
This is not that slow, but we only work with 1/10 of the dataset, if you try the full dataset or use ntreeTry of larger values... it is going to be SLOW....

library(randomForest)

## randomForest 4.6-10  
## Type rfNews() to see new features/changes/bug fixes.

set.seed(123)  
pick <- sample(1:nrow(diamonds), size = nrow(diamonds)/10, replace = F)  
X = diamonds[pick, -7]  
Y = diamonds$price[pick]  
tuneRF(x = X, y = Y)

## mtry = 3 OOB error = 508881.7   
## Searching left ...  
## mtry = 2 OOB error = 608380.4   
## -0.1955244 0.05   
## Searching right ...  
## mtry = 6 OOB error = 437976.3   
## 0.1393356 0.05   
## mtry = 9 OOB error = 418858.5   
## 0.04365023 0.05



## mtry OOBError  
## 2 2 608380.4  
## 3 3 508881.7  
## 6 6 437976.3  
## 9 9 418858.5

This is even slower... You don't want to try this... even on 1/10 of the diamonds dataset

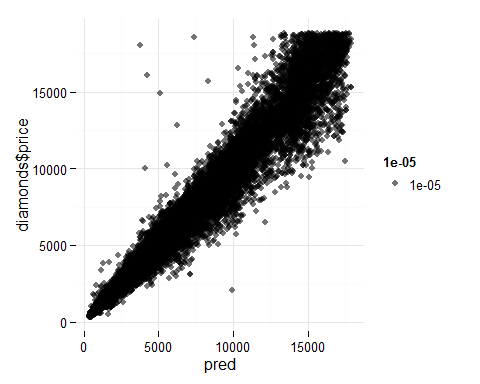
result <- rfcv(trainx = X,   
 trainy = Y,  
 cv.fold= 5,  
 step = 3)

If we do this on the entire dataset, we shall push the button and go home, come back to see result tomorrow.

set.seed(0306)  
RFModel <- randomForest(x = X,  
 y = Y,  
 mtry = 6,  
 importance = T,  
 proximity = F)  
RFModel$importance

## %IncMSE IncNodePurity  
## carat 8314560.38 21980179338  
## cut 21235.29 166611132  
## color 1078929.43 2283716075  
## clarity 2053404.66 4549672796  
## depth 24628.49 354046238  
## table 16757.71 219485840  
## x 2824632.78 11206070193  
## y 13258166.37 39857859442  
## z 559685.01 1898682964

pred <- predict(RFModel, diamonds[,-7])  
qplot(x = pred, y = diamonds$price, alpha = 0.00001)



1. This is a demo footnote [↑](#footnote-ref-28)