Revolution R

David Chiu

2016/09/21

R語言與Revolution R 對照表

Summary of rx Functions

https://msdn.microsoft.com/en-us/library/mt652103.aspx

rx function	Description	Nearest base R function
rxImport	Creates an .XDF file or data frame from a data source such as a text file, data file, ODBC or Teradata connection, or data frame)	
rxXdfToText	Creates a text file from an .XDF file	

rxGetInfo	Retrieves header information from an .XDF file or summary information from a data frame	`str()`
		`names()`
		`colNames()`

rxSetInfo	Sets a file description in an .XDF file or a description attribute in a data frame	
rxGetVarInfo	Retrieves variable information from an .XDF file or data frame	`names()` `str()`

dplyrXdf

敘述性統計分析

- 多數資料分析，80% 在於如何加總與平均
  - 銷售份額
  - 客戶數量
  - 業績成長量

- 用SQL做敘述性統計
  - select * from tb1 where col1 >= 100 limit 3

- R有類似的工具嗎?
  - plyr
  - reshape2
  - dplyr

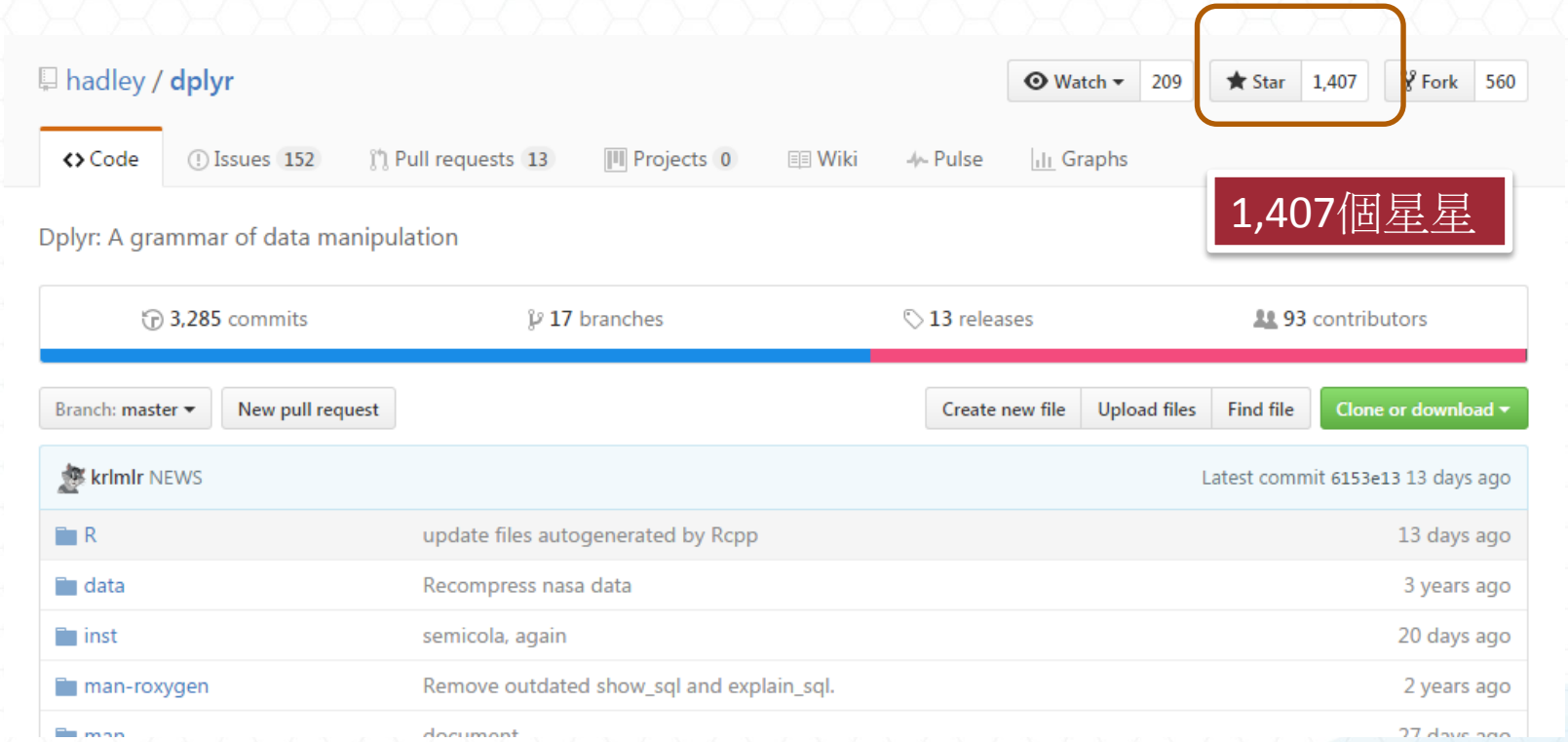
如何操作資料

- 關於操作資料，你需要
  - 可以分割資料(Split)
  - 可以轉換資料(Transformation)
  - 可以聚合資料(Aggregation)
  - 可以探索資料(Exploration)

- 需要如同SQL的語法操作

dplyr

https://github.com/hadley/dplyr



使用 dplyr

- 讓 R 可以像SQL一樣可以使用結構化語句快速聚合、分析資料
- 可以使用Magrittr 套件的管道 (Pipeline) 傳遞資料

%>%
magrittr

Ceci n'est pas un pipe.

Hadley Wickham
http://hadley.nz/

為什麼要使用dplyr

- 提供操作資料的基本語法
  - filter, select, arrange, mutate, summarise, group_by
- 提供資料合併功能(JOIN)
  - Inner join, left join
- 可以操作資料表(data table) 或資料庫 (Database) 的資料

安裝與使用dplyr

- 安裝dplyr
  - install.packages("dplyr")
- 使用dplyr
  - library(dplyr)
- 觀看說明頁
  - help(package='dplyr')

聚合房價資料

- 讀取資料集

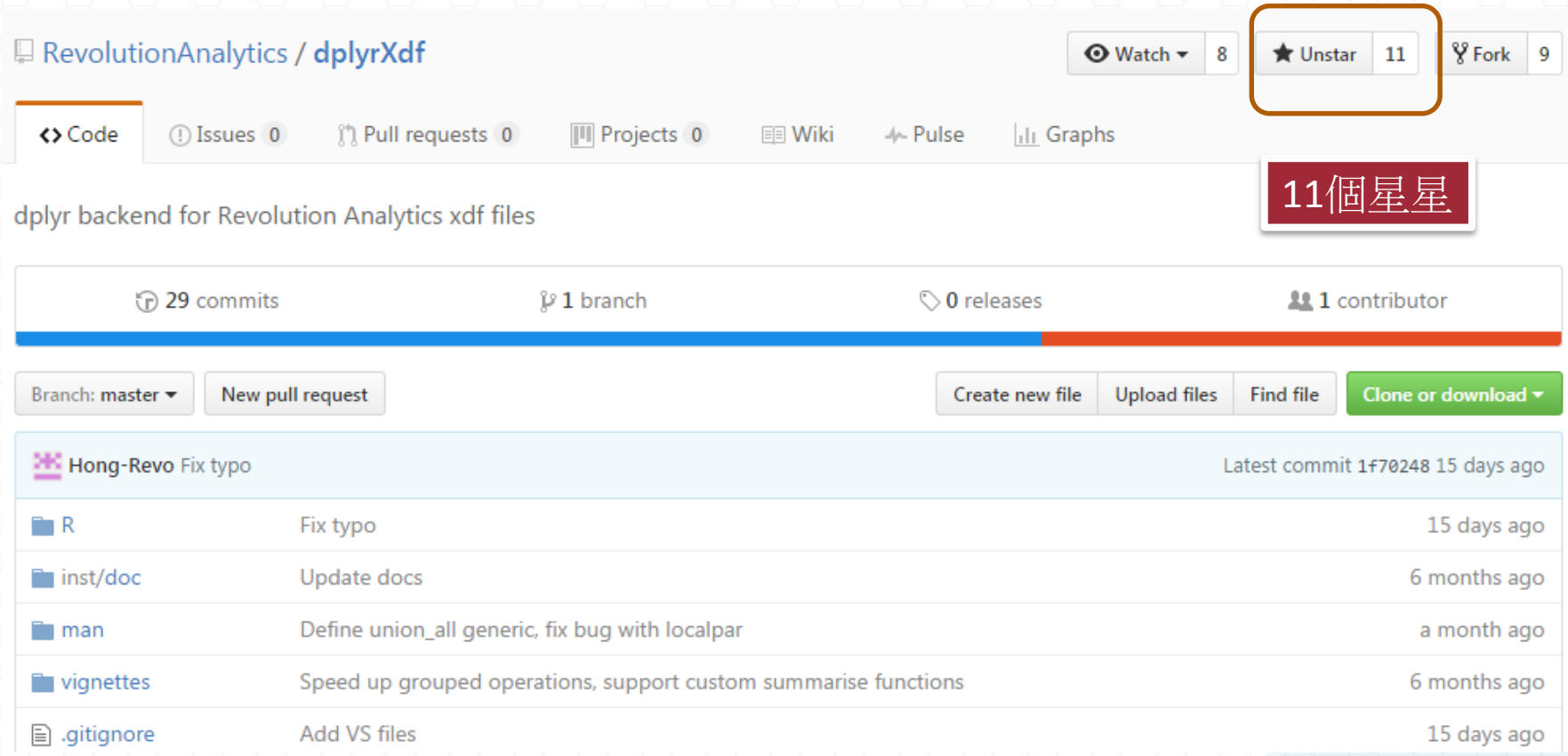
```
lvr_df <- read.csv('lvr_prices.csv')
```

- 聚合資料

```
total_means_by_area <- lvr_df %>%
filter(trading_target == '房地(土地+建物)') %>%
group_by(area) %>%
summarise_each(funs(mean),total_price)
```

dplyrXdf

- https://github.com/RevolutionAnalytics/dplyrXdf



透過GitHub 安裝

- 安裝devtools

```
install.packages('devtools')
```

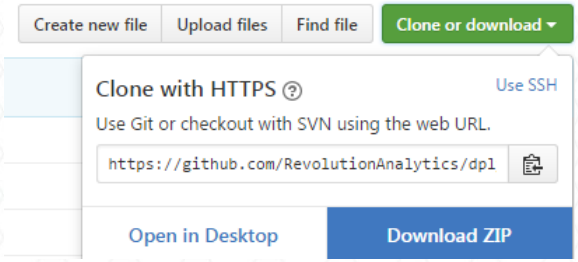
- 從GitHub 安裝dplyrXdf

```
devtools::install_github('RevolutionAnalytics/dplyrXdf')
```

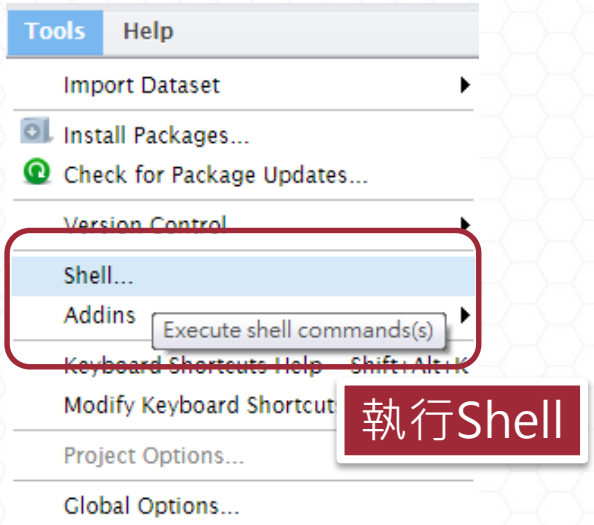
但是devtools安裝會有點久

自行包裝

- 找到ZIP 檔連結



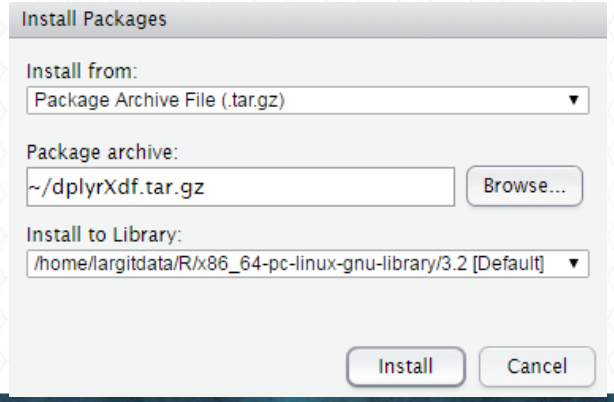
- 執行Shell
  - Tools -> Shell



- 使用wget下載
  - wget https://github.com/RevolutionAnalytics/dplyrXdf/archive/master.zip

解壓縮以後包裝成tar.gz 檔

- 解壓縮zip 檔
  - unzip master.zip
- 重新包裝dplyrXdf 成tar.gz 檔
  - tar -zcvf dplyrXdf.tar.gz dplyrXdf-master
- 安裝dplyrXdf



計算與檢視資料

- 讀取資料

```
lvr_data = RxXdfData('lvr.xdf')
```

- 計算平均總價

```
total_means_by_area <- lvr_data %>%
filter(trading_target == '房地(土地+建物)') %>%
group_by(area) %>%
summarise_each(funs(mean),total_price)
```

- 檢視資料

```
head(total_means_by_area )
class(total_means_by_area )
```

rxMerge

RxMerge 做資料合併

```
indData <- data.frame(id = 1:12, state = rep(c("CA","OR", "WA"), times = 4))
stateData <- data.frame(state=c("CA","OR", "WA"), stateVal = c(1000, 400, 500))

rxDataStep(inData = indData, outFile = 'data1.xdf', overwrite=TRUE)
rxDataStep(inData = stateData, outFile = 'data2.xdf', overwrite=TRUE)

mergedf<-rxMerge(inData1 = 'data1.xdf', inData2 = 'data2.xdf', outFile = 'merge.xdf',matchVars = "state")

df <- rxDataStep(mergedf)
```

transformFunc

rxSummary

```
## Compute the summary statistics
(csSummary <- rxSummary(~ creditScore, data = mortData))

## Extract the mean and std. deviation
meanCS <- csSummary$sDataFrame$Mean[1]
sdCS <- csSummary$sDataFrame$StdDev[1]

## Create a function to compute the scaled variable
scaleCS <- function(mylist){
  mylist[["scaledCreditScore"]] <- (mylist[["creditScore"]] - myCenter) / myScale
  return(mylist)
}
```

transformFunc

```
## Run it with rxDataStep
myMortData <- "myMD.xdf"
rxDataStep(inData = mortData, outFile = myMortData,
           transformFunc = scaleCS,
           transformObjects = list(myCenter = meanCS, myScale = sdCS)
           )

## Check the new variable:
rxGetVarInfo(myMortData)
rxSummary( ~ scaledCreditScore, data = myMortData)
```

THANK YOU