jiebaR & wordcloud2

组长：2000022752-文胜熙
组员：2000022764-赵宁
2000022745-彭瑞
2000022773-郭宇桐

jiebaR

- 最大似然法分割模型
- 算法基础：前序树，有向无环图，动态规划，

  隐马尔可夫(HMM)模型（基于人民日报语言库训练）

  可根据用户自定义的字典和HMM模型进行分割

- 主要功能：对给定的文本（文件）进行单词划分；关键词提取
- 使用场景：词频统计、情感分析……

worker()

worker(type = "mix", dict = DICTPATH, hmm = HMMPATH, user = USERPATH, idf = IDFPATH, stop_word = STOPPATH, write = T, qmax = 20, topn = 5, encoding = "UTF-8", detect = T, symbol = F, lines = 1e+05, output = NULL, bylines = F)

type可选参数：
mp（最大概率模型）、hmm（HMM模型）、mix（混合模型）、query（索引模型）、tag（标记模型）、keywords（关键词模型）、simhash（Simhash 模型）

待分词文字

曹操闻听，大吃一惊，想当初关公在白马坡斩颜良之时，曾对某家言道，他有一结拜三弟，姓张名飞字翼德，在百万军中取上将之首如探囊取物，反掌观纹一般，今日一见，果然英勇。“撤去某家青罗伞盖，观一观那莽撞人武艺如何！”

mp（最大概率模型）

说人话：能分的地方都分，可能是词的组合都当成单词

```
> sp_mp = worker(type="mp")
> segment(word,sp_mp)
 [1] "曹操"     "闻听"     "大吃一惊" "想当初"   "关公"     "在"       "白马"
 [8] "坡"       "斩"       "颜良之"   "时"       "曾"       "对"       "某"
[15] "家"       "言"       "道"       "他"       "有"       "一"       "结拜"
[22] "三"       "弟"       "姓张"     "名"       "飞"       "字"       "翼"
[29] "德"       "在"       "百万"     "军中"     "取"       "上将"     "之"
[36] "首"       "如"       "探囊取物" "反"       "掌"       "观"       "纹"
[43] "一般"     "今日"     "一"       "见"       "果然"     "英勇"     "撤"
[50] "去"       "某"       "家"       "青罗"     "伞盖"     "观"       "一"
[57] "观"       "那"       "莽撞"     "人"       "武艺"     "如何"
```

hmm（HMM模型）

说人话：有可能产生新词的地方就产生新词

```
> sp_hmm = worker(type = 'hmm')
> segment(word,sp_hmm)
 [1] "曹操闻"   "听"     "大吃"     "一惊"   "想"     "当初"   "关公在"
 [8] "白马坡"   "斩"     "颜良"     "之时"   "曾"     "对"     "某家言道"
[15] "他"       "有"     "一结"     "拜"     "三弟"   "姓"     "张名"
[22] "飞字翼德" "在"     "百万军"   "中取"   "上"     "将"     "之首"
[29] "如"       "探囊"   "取物"     "反掌"   "观纹"   "一般"   "今日"
[36] "一见"     "果然"   "英勇"     "撒"     "去"     "某"     "家"
[43] "青"       "罗"     "伞"       "盖"     "观一观" "那莽"   "撞"
[50] "人"       "武艺"   "如何"
```

mix（默认，混合模式）

说人话：先mp，再hmm

```
> spliter = worker()
> segment(word,spliter)
 [1] "曹操"     "闻听"     "大吃一惊" "想当初"   "关公"     "在"       "白马"
 [8] "坡"       "斩"       "颜良之"   "时"       "曾"       "对"       "某家言道"
[15] "他"       "有"       "一"       "结拜"     "三弟"     "姓张"     "名飞"
[22] "字翼德"   "在"       "百万"     "军中"     "取"       "上将"     "之首"
[29] "如"       "探囊取物" "反掌"     "观纹"     "一般"     "今日"     "一见"
[36] "果然"     "英勇"     "撒"       "去"       "某"       "家"       "青罗"
[43] "伞盖"     "观一观"   "那"       "莽撞"     "人"       "武艺"     "如何"
```

query

说人话：给mix再切一刀

```
> sp_query = worker(type = "query")
> segment(word,sp_query)
 [1] "曹操"     "闻听"     "大吃一惊" "当初"     "想当初"   "关公"     "在"
 [8] "白马"     "坡"       "斩"       "颜良之"   "时"       "曾"       "对"
[15] "某家"     "某家言道" "他"       "有"       "一"       "结拜"     "三弟"
[22] "姓张"     "名飞"     "字翼德"   "在"       "百万"     "军中"     "取"
[29] "上将"     "之首"     "如"       "探囊取物" "反掌"     "观纹"     "一般"
[36] "今日"     "一见"     "果然"     "英勇"     "撤"       "去"       "某"
[43] "家"       "青罗"     "伞盖"     "观一观"   "那"       "莽撞"     "人"
[50] "武艺"     "如何"
```

tag

说人话：给分出来的词做词性标注

```
> sp_tag = worker(type = "tag")
> segment(word,sp_tag)
      nr       nr        i        l       nr       p      ns
"曹操"   "闻听" "大吃一惊"  "想当初"  "关公"   "在"  "白马"
       n        v       nr        n        d        p        x
  "坡"     "斩"  "颜良之"     "时"     "曾"     "对" "某家言道"
       r        v        m        v        x        n        x
  "他"     "有"      "一"   "结拜"   "三弟"   "姓张"   "名飞"
       x        p        m        s        v        n        f
"字翼德"     "在"   "百万"   "军中"   "取"   "上将"   "之首"
       v        i        x        x        a        t        x
  "如" "探囊取物"   "反掌"   "观纹"  "一般"  "今日"  "一见"
       d       nr        v        v        r        q      ns
"果然"   "英勇"    "撤"    "去"    "某"    "家"  "青罗"
       n        x        r        z        n        n        r
"伞盖" "观一观"    "那"   "莽撞"    "人"  "武艺"  "如何"
```

new_user_word() 添加自建词库

```
> new_user_word(spliter,c("白马坡","反掌观纹"),c("ns","v"))
[1] TRUE
> segment(word,spliter)
 [1] "曹操"     "闻听"     "大吃一惊" "想当初"   "关公"     "在"       "白马坡"
 [8] "斩"       "颜良之"   "时"       "曾"       "对"       "某家言道" "他"
[15] "有"       "一"       "结拜"     "三弟"     "姓张"     "名飞"     "字翼德"
[22] "在"       "百万"     "军中"     "取"       "上将"     "之首"     "如"
[29] "探囊取物" "反掌观纹" "一般"     "今日"     "一见"     "果然"     "英勇"
[36] "撤"       "去"       "某"       "家"       "青罗"     "伞盖"     "观一观"
[43] "那"       "莽撞"     "人"       "武艺"     "如何"
```

```
> spliter = worker()
> segment(word,spliter)
 [1] "曹操"     "闻听"     "大吃一惊" "想当初"   "关公"     "在"       "白马"
 [8] "坡"       "斩"       "颜良之"   "时"       "曾"       "对"       "某家言道"
[15] "他"       "有"       "一"       "结拜"     "三弟"     "姓张"     "名飞"
[22] "字翼德"   "在"       "百万"     "军中"     "取"       "上将"     "之首"
[29] "如"       "探囊取物" "反掌"     "观纹"     "一般"     "今日"     "一见"
[36] "果然"     "英勇"     "撤"       "去"       "某"       "家"       "青罗"
[43] "伞盖"     "观一观"   "那"       "莽撞"     "人"       "武艺"     "如何"
```

```
> new_user_word(sp_tag,c("白马坡","反掌观纹"),c("ns","v"))
[1] TRUE
> segment(word,sp_tag)
      nr       nr        i        l       nr        p       ns
"曹操"   "闻听" "大吃一惊"  "想当初"  "关公"    "在"  "白马坡"
       v       nr        n        d        p        x        r
  "斩"  "颜良之"    "时"    "曾"    "对" "某家言道"    "他"
       v        m        v        x        n        x        x
  "有"    "一"  "结拜"  "三弟"  "姓张"  "名飞"  "字翼德"
       p        m        s        v        n        f        v
  "在"  "百万"  "军中"    "取"  "上将"  "之首"    "如"
       i        v        a        t        x        d       nr
"探囊取物" "反掌观纹"  "一般"  "今日"  "一见"  "果然"  "英勇"
       v        v        r        q       ns        n        x
  "撤"    "去"    "某"    "家"  "青罗"  "伞盖" "观一观"
       r        z        n        n        r
  "那"  "莽撞"    "人"  "武艺"  "如何"
> sp_tag = worker(type = "tag")
> segment(word,sp_tag)
      nr       nr        i        l       nr        p       ns
"曹操"   "闻听" "大吃一惊"  "想当初"  "关公"    "在"   "白马"
       n        v       nr        n        d        p        x
  "坡"    "斩"  "颜良之"    "时"    "曾"    "对" "某家言道"
       r        v        m        v        x        n        x
  "他"    "有"    "一"  "结拜"  "三弟"  "姓张"  "名飞"
       x        p        m        s        v        n        f
"字翼德"    "在"  "百万"  "军中"    "取"  "上将"  "之首"
       v        i        x        x        a        t        x
  "如" "探囊取物"  "反掌"  "观纹"  "一般"  "今日"  "一见"
       d       nr        v        v        r        q       ns
"果然"   "英勇"    "撤"    "去"    "某"    "家"  "青罗"
       n        x        r        z        n        n        r
"伞盖" "观一观"    "那"  "莽撞"    "人"  "武艺"  "如何"
```

segment（分词）

segment函数可以进行分词，返回的是一个向量，还可以直接对一个文件进行分词。

```
> mixseg <- worker() #worker()函数是用来初始化分词引擎的,默认为混合模型。
> text <- "我们来一起学习数据科学"
> segment(text, mixseg) #segment函数可以进行分词，返回的是一个向量。
[1] "我们" "来"   "一起" "学习" "数据" "科学"
```

```
还有另外一些写法，可以代替segment函数:
mixseg[text]
[1] "我们" "来"   "一起" "学习" "数据" "科学"
mixseg<=text
[1] "我们" "来"   "一起" "学习" "数据" "科学"

#当然还可以直接对一个文件进行分词，比如:
segment('D:/test.txt', mixseg)
```

segment（分词）

分行输出 $bylines

```
分词器 = worker(bylines = TRUE)

segment(c("这是第一行文本。","这是第二行文本。"), 分词器)
#> [[1]]
#> [1] "这是"   "第一行" "文本"
#>
#> [[2]]
#> [1] "这是"   "第二行" "文本"
```

segment（分词）

保留符号 $symbol

```
分词器 = worker(symbol = TRUE)
segment(c("Hi，这是第一行文本。"), 分词器)
#> [1] "Hi"     "，"     "这是"   "第一行" "文本"   "。"
segment(c("。，。；las"), 分词器)
#> [1] "。"  "，"  "。"  "；"  "las"

分词器 = worker(symbol = FALSE)
segment(c("Hi，这是第一行文本。"), 分词器)
#> [1] "Hi"     "这是"   "第一行" "文本"
segment(c("。，。；las"), 分词器)
#> [1] "las"
```

segment（分词）

readLines / writeLines -对文件进行分词

使用 readLines 函数读取对应文本，

```
texts = readLines("./index.rmd", encoding="UTF-8")
分词器$bylines = TRUE
分词结果 = segment(texts, 分词器)
```

使用 writeLines 写入文件

```
合并各行分词结果 =sapply(分词结果, function(x){ paste(x, collapse = " ")})
writeLines(合并各行分词结果, "./某个文件.txt")
file.remove("./某个文件.txt")
#> [1] TRUE
```

tagger（标注词性）

对已经分好词的文本进行标记

```
分词器 = worker()
分词结果 = segment(一段文本, 分词器)
分词结果
#> [1] "我"     "爱"     "北京"   "天安门"
vector_tag(分词结果, 标记器)
#>        r        v       ns       ns
#>     "我"     "爱"   "北京" "天安门"
```

tagger（标注词性）

可以使用 <=.tagger 或者 tag 来进行分词和词性标注，词性标注使用混合模型模型分词，标注采用和 ictclas 兼容的标记法。

```
> words = "我爱北京天安门"
> tagger = worker("tag")
> tagger <= words
     r      v     ns     ns
  "我"   "爱"  "北京" "天安门"
```

tagger （标注词性）

标签	含义	标签	含义	标签	含义	标签	含义
n	普通名词	f	方位名词	s	处所名词	t	时间
nr	人名	ns	地名	nt	机构名	nw	作品名
nz	其他专名	v	普通动词	vd	动副词	vn	名动词
a	形容词	ad	副形词	an	名形词	d	副词
m	数量词	q	量词	r	代词	p	介词
c	连词	u	助词	xc	其他虚词	w	标点符号
PER	人名	LOC	地名	ORG	机构名	TIME	时间

keywords（关键词提取）

topn参数为关键词的个数

```
keys = worker("keywords", topn = 1)
keys <= "我爱北京天安门"
  8.9954
"天安门"
```

相对词频值

keywords（关键词提取）

```
keys = worker("keywords", topn = 1)
keys <= "我爱北京天安门"
  8.9954
"天安门"
```

相对词频值

TF-IDF关键词提取法的基本思想：

词语的重要性与它在文件中出现的次数成正比，但同时会随着它在语料库中出现的频率成反比下降。

TF-IDF = TF * IDF

TF（Term Frequency）表示一个词在当前文档中出现的次数。

IDF（Inverse Document Frequency）为逆文档频率，IDF= log(语料库中文档总数/(包含该词的文档数+1))

由公式可知：一个词在文档中出现的次数越多，其TF值就越大，整个语料库中包含某个词的文档数越少，则IDF值越大，因此某个词的TF-IDF值越大，则这个词是关键词的概率越大。

keywords（关键词提取）

对古龙《多情剑客无情剑》第一章进行关键词提取

```
keys = worker("keywords", topn = 4)
keys <= gulong

 516.525   264.77  247.599  234.873
"李寻欢"   "诸葛"   "白蛇"   "少年"
```

freq（词频统计）

```
mixseg <- worker()
text <- "We need to deal with the enemy superhuman cou
words <- segment(text,mixseg)
wordfreqs <- jiebaR::freq(words)
wordfreqs
```

返回的数据类型是数据框

"We need to deal with the enemy superhuman courage,and to adhere to a friend in front of their position,but also a great deal of courage"

	char	freq
1	great	1
2	position	1
3	but	1
4	their	1
5	a	2
6	deal	2
7	superhuman	1
8	to	3
9	also	1
10	We	1
11	courage	2
12	the	1
13	need	1
14	enemy	1
15	and	1
16	in	1
17	adhere	1
18	friend	1
19	front	1
20	with	1
21	of	2

freq（词频统计）

```
wordfreqs <- dplyr::arrange(wordfreqs,freq)
wordfreqs
```

```
wordfreqs <- dplyr::arrange(wordfreqs,-freq)
wordfreqs
```

dplyr包提供了统一形式的函数用于操作数据框

	char	freq
1	great	1
2	position	1
3	but	1
4	their	1
5	superhuman	1
6	also	1
7	We	1
8	the	1
9	need	1
10	enemy	1
11	and	1
12	in	1
13	adhere	1
14	friend	1
15	front	1
16	with	1
17	a	2
18	deal	2
19	courage	2
20	of	2
21	to	3

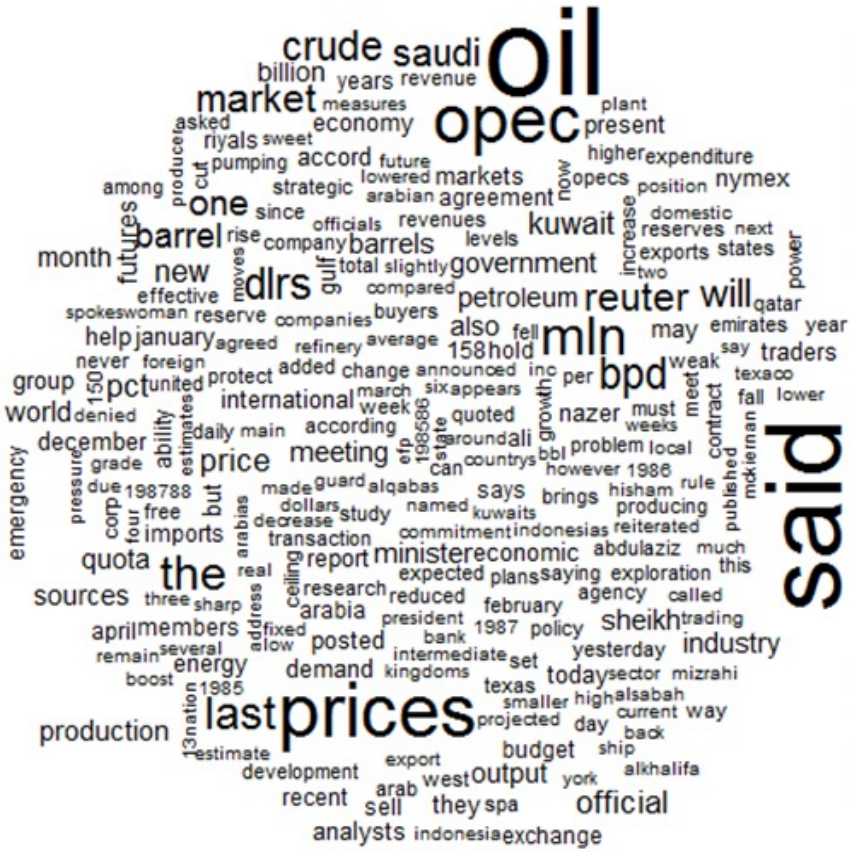
按词频正序

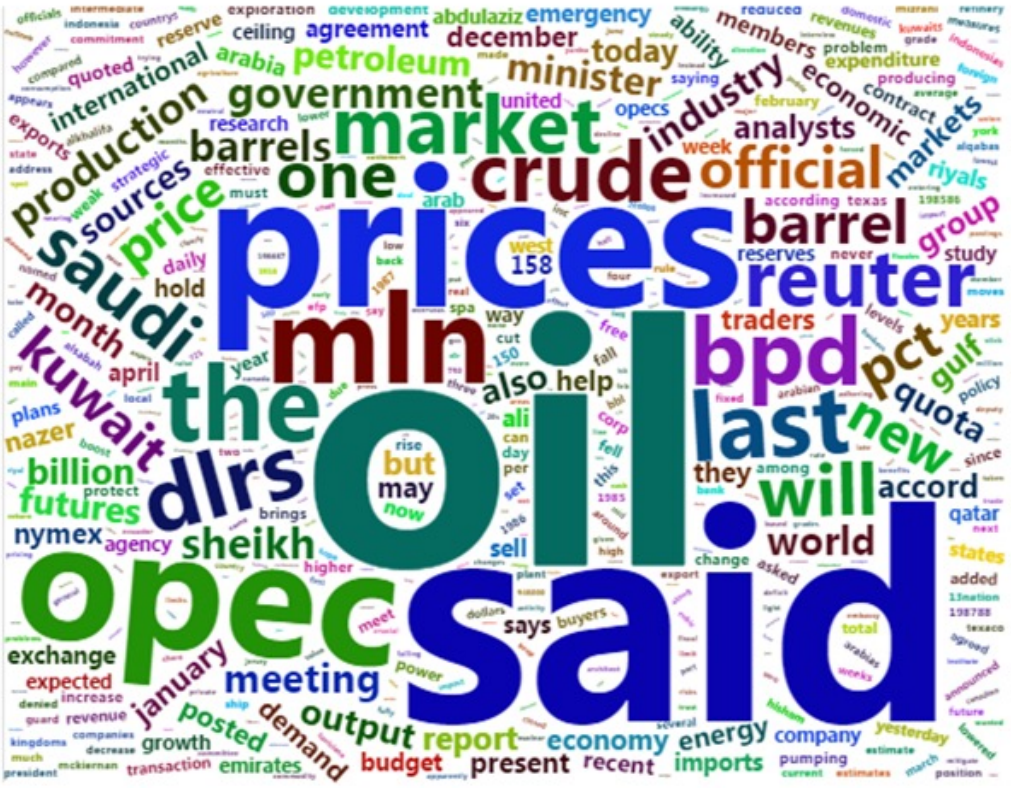
	char	freq
1	to	3
2	a	2
3	deal	2
4	courage	2
5	of	2
6	great	1
7	position	1
8	but	1
9	their	1
10	superhuman	1
11	also	1
12	We	1
13	the	1
14	need	1
15	enemy	1
16	and	1
17	in	1
18	adhere	1
19	friend	1
20	front	1
21	with	1

按词频倒序

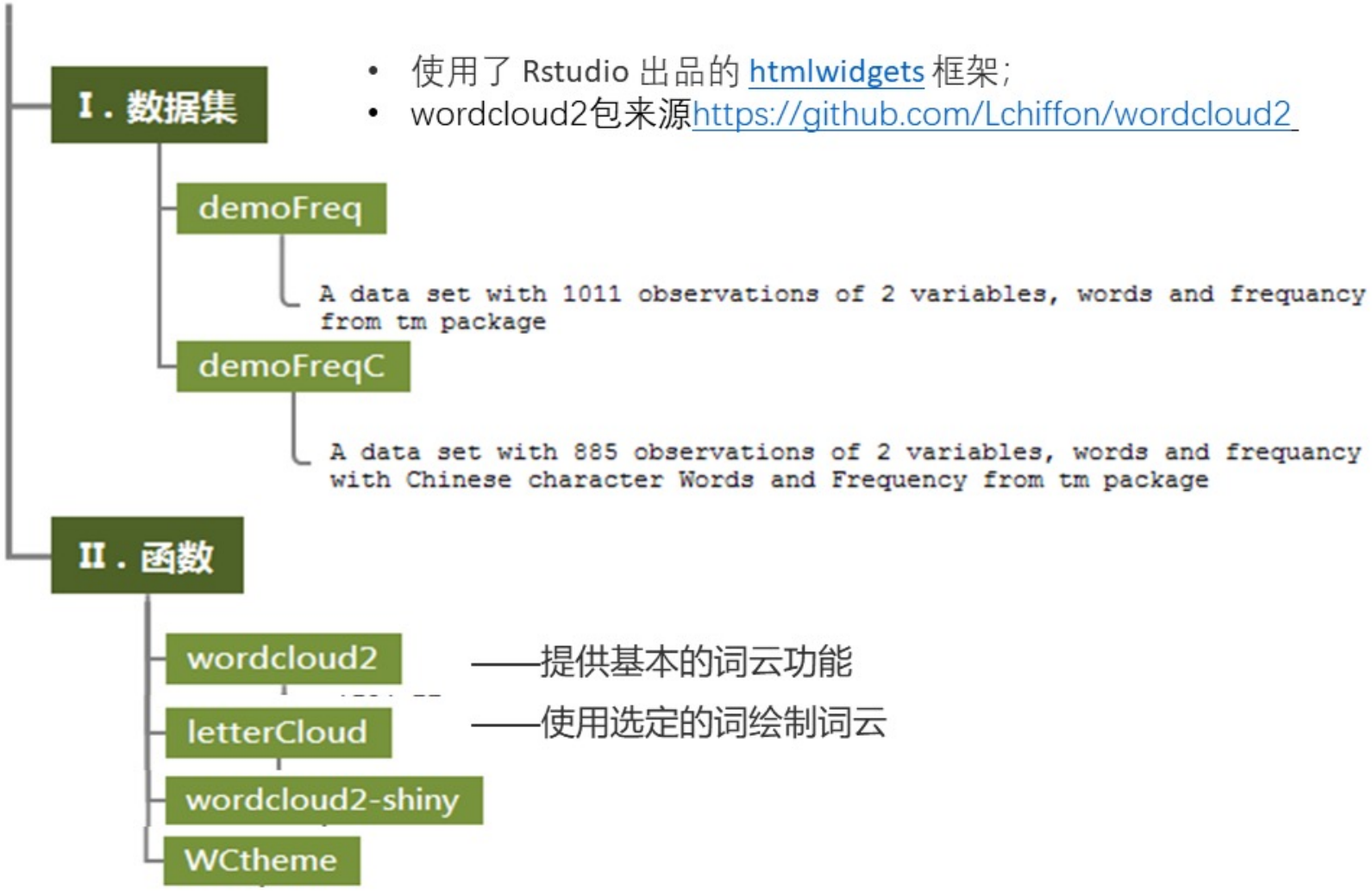
-wordcloud vs wordcloud2-

`wordcloud(demoFreq$word, demoFreq$freq)` `wordcloud2(demoFreq)`





wordcloud2 package



- 使用了 Rstudio 出品的 htmlwidgets 框架;
- wordcloud2包来源https://github.com/Lchiffon/wordcloud2

II. 函数

wordcloud2

```
wordcloud2(data, size = 1, minSize = 0, gridSize =  0,
    fontFamily = NULL, fontWeight = 'normal',
    color = 'random-dark', backgroundColor = "white",
    minRotation = -pi/4, maxRotation = pi/4, rotateRatio = 0.4,
    shape = 'circle', ellipticity = 0.65, widgetsize = NULL)
```

基于wordcloud2.js创建文字云

常用参数：

（1）**data**：**词云生成数据**，包含具体词语以及频率；

（2）**size**：**字体大小**，默认为1，一般来说该值越小，生成的形状轮廓越明显；

（3）**fontFamily**：**字体**，如‘微软雅黑’；

（4）**fontWeight**：**字体粗细**，包含‘normal’，‘bold’以及‘600’；；

（5）**color：字体颜色**，可以选择‘random-dark’以及‘random-light’，其实就是颜色色系；

（6）**backgroundColor：背景颜色**，支持R语言中的常用颜色，如‘gray’，‘black’，但是还支持不了更加具体的颜色选择，如‘gray20’；

（7）**minRontatin与maxRontatin：字体旋转角度范围的最小值以及最大值**，选定后，字体会在该范围内随机旋转；

（8）**rotationRation：字体旋转比例**，如设定为1，则全部词语都会发生旋转；

（9）**shape：词云形状选择**，默认是‘circle’，即圆形。还可以选择‘cardioid’（苹果形或心形），‘star’（星形），‘diamond’（钻石），‘triangle-forward’（三角形），‘triangle’（三角形），‘pentagon’（五边形）；

```
wordcloud2(demoFreq, size = 1,shape = 'star')
```

Chinese version

```
## Sys.setlocale("LC_CTYPE","eng")
wordcloud2(demoFreqC, size = 2, fontFamily = "微软雅黑",
           color = "random-light", backgroundColor = "grey")
```

中文乱码的问题
•如果是 windows，需要 Sys.setlocale("LC_CTYPE", "eng")
•输入的文字需要是中文 UTF-8

使用自定义图片作为词云的背景形状

```
figPath = system.file("examples/t.png",package = "wordcloud2")
wordcloud2(demoFreq, figPath = figPath, size = 1.5,color = "skyblue")
```

letterCloud

基于某个word的形状绘制文字云

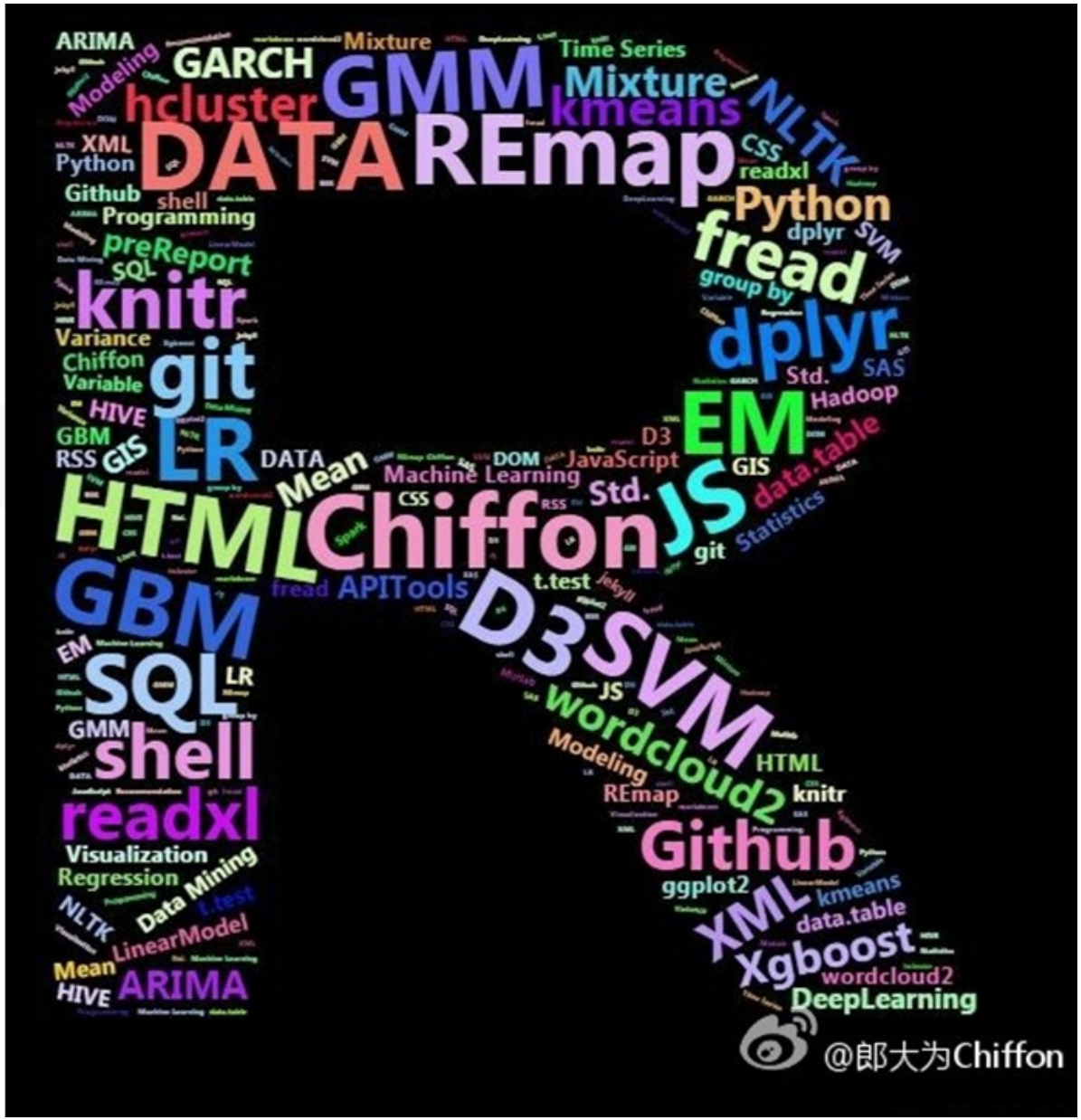
```
letterCloud(data, word, wordSize = 0, letterFont = NULL, ...)
```

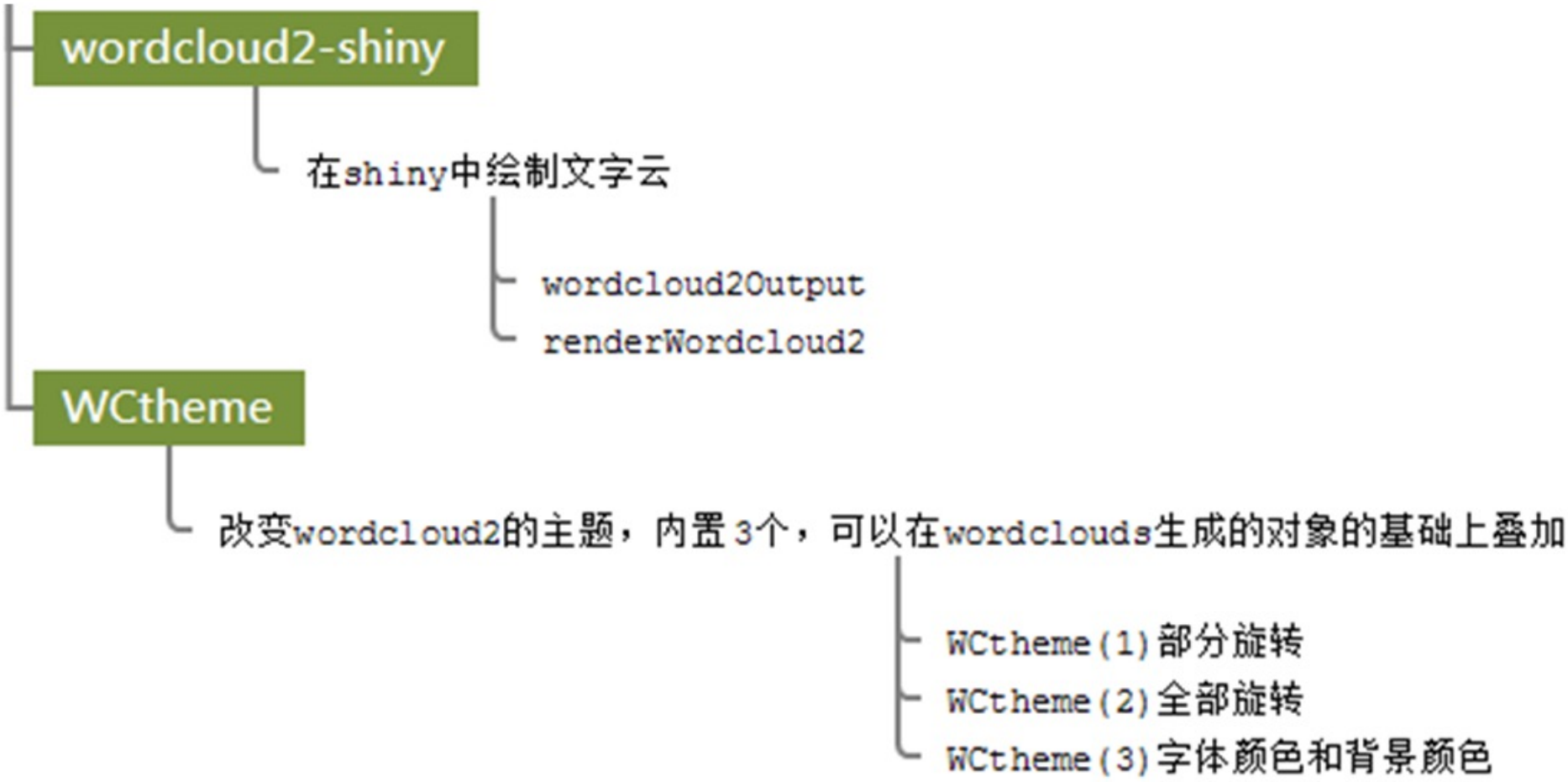
data:包含word和frequency的数据框
word: 文字云所展示的word
wordSize: 字号
letterFont: 字体
该函数可使用wordcloud2中的参数

```
letterCloud(dat,letter = "R", color = "random-light",
    backgroundColor = "black",size = 0.3)
```

```
letterCloud(dat,letter = "R", color = "random-light",
   backgroundColor = "black",size = 0.3)
```

- letter=“R” 选择了字符 R 的形状，
- 颜色随机亮色，
- 背景黑色，
- 大小 0.3；




wordcloud2-shiny
在shiny中绘制文字云
wordcloud2Output
renderWordcloud2
WCtheme
改变wordcloud2的主题，内置3个，可以在wordclouds生成的对象的基础上叠加
WCtheme(1)部分旋转
WCtheme(2)全部旋转
WCtheme(3)字体颜色和背景颜色

谢谢观看！