用定性数据分析包 RQDA tm 进行文本挖掘

Written by Benson Ye (bensonye@189.cn)
Revised by Ronggui Huang (ronggui.huang@gmail.com)
First reversion 2010-07-22
Last revision 2010-08-03

在对访谈内容或剧本、小说部分内容进行文本挖掘时，如果用不断的剪粘保存的方法非常繁琐而且容易漏掉一些内容。好在黄荣贵开发的 RQDA 包可以进行文档管理和内容编码及提取，大大方便了利用 tm 包进行文本挖掘，既提高了效率又提高了准确性，下面举一个小例子：

对（人民网 >> 时政 >> 时政专题 >> 网友进言）中的公安部回应进行分析

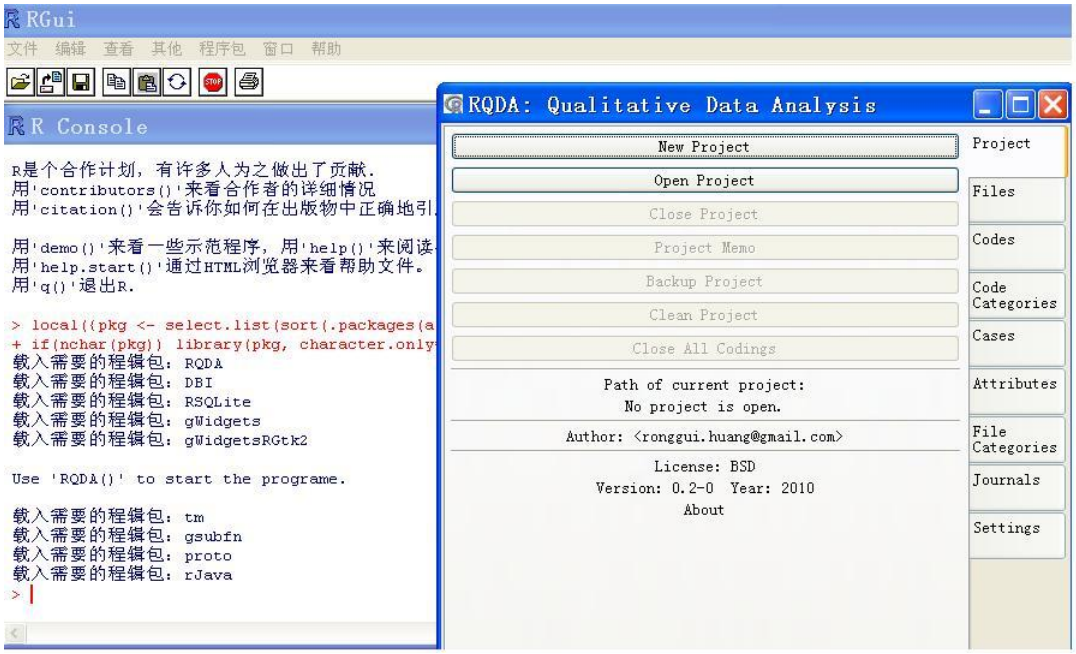
相关链接：http://politics.people.com.cn/GB/8198/138817/index.html

1、安装 RQDA 包、tm 包和中文分词软件；

```
> install.packages(c("rJava","tm", "gsubfn"))
> install.packages(c("RQDA","RQDAtm"),repos="http://R-Forge.R-project.org",type='source')
```

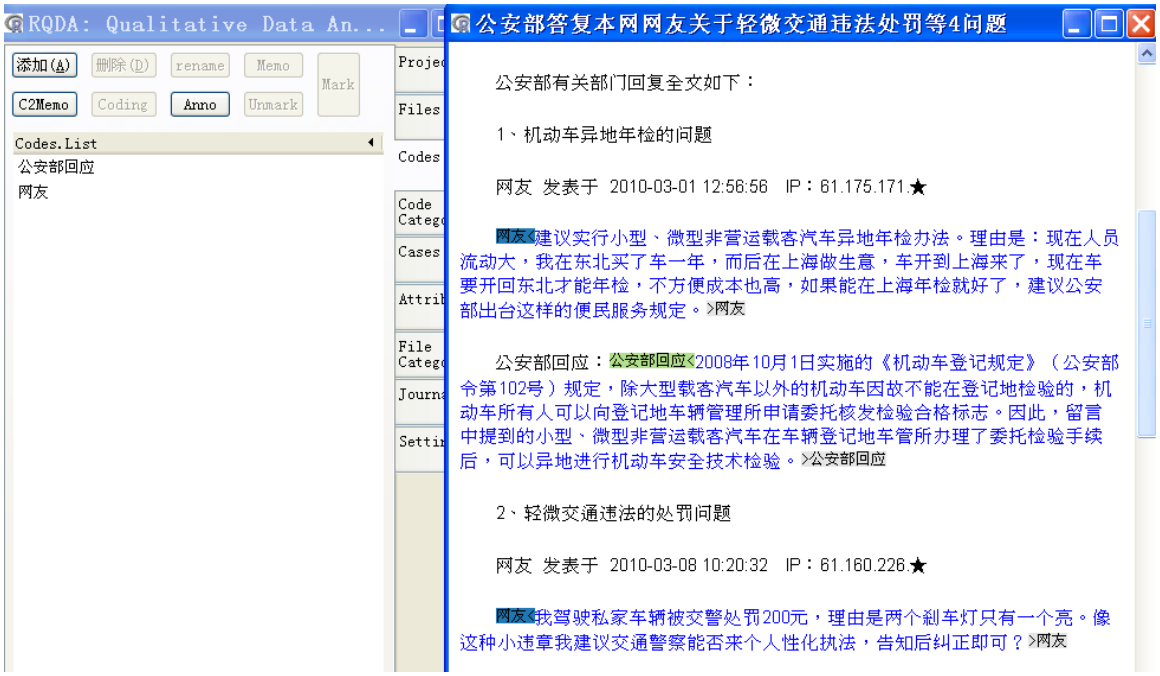
2、装载 RQDA 包并建立一个新的工程项目；

```
> library(RQDAtm)
```

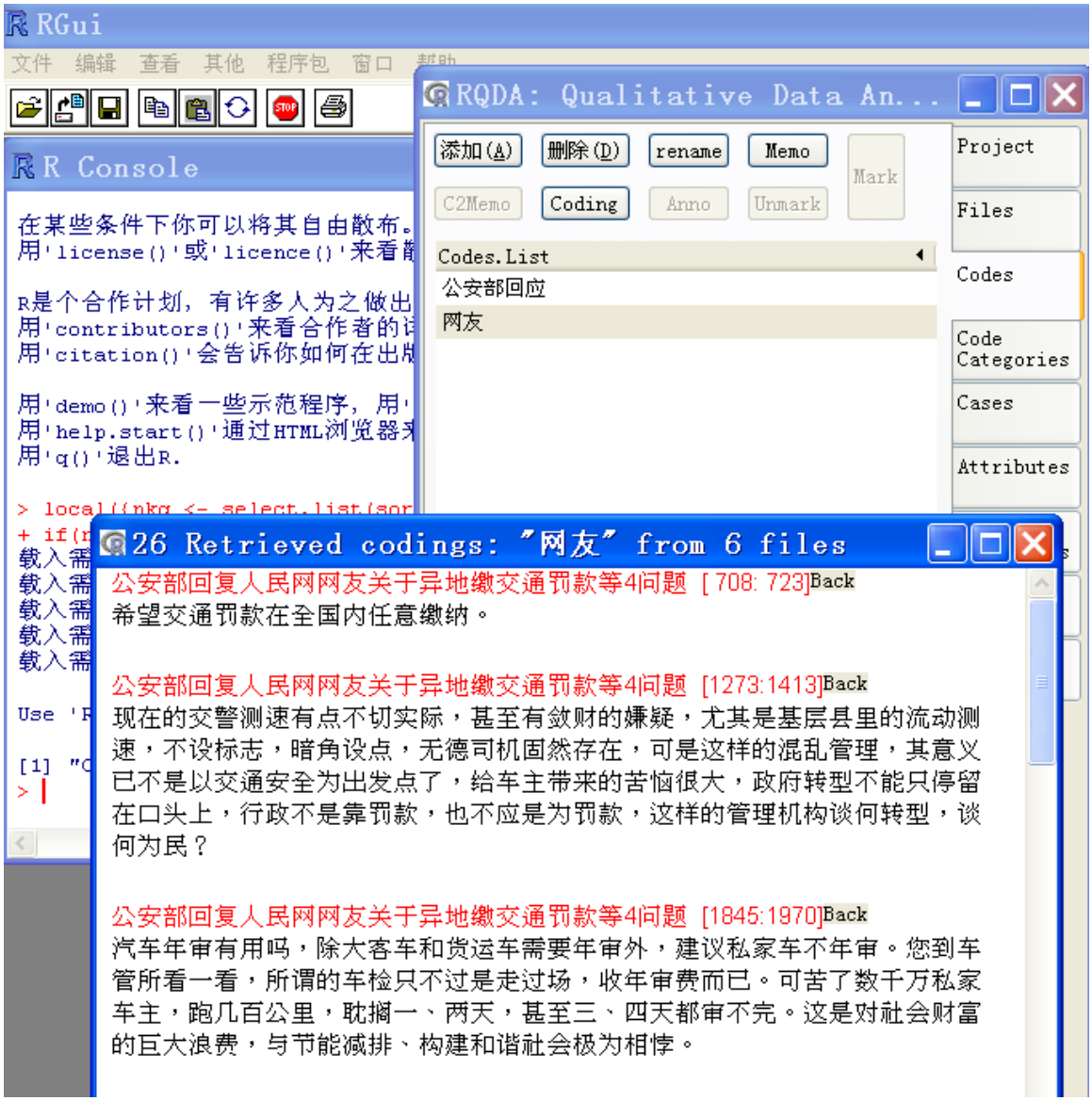


3、输入相关文本文件；

4、进行编码和作标记；



5、双击想要提取的编码即可提取相关文本；



6、运行下面下载的程序进行文本提取、转换、分词、文本挖掘工作。
(以上步骤的结果为 RQDA2tm_example.rqda)，可直接打开该文件继续如下步骤。

```
> gg <- RQDA2tm("公安部回应" )
> summary(gg)
A corpus with 26 text documents

The metadata consists of 2 tag-value pairs and a data frame
Available tags are:
   create_date creator
Available variables in the data frame are:
   MetaID cid fid selfirst selend fname
> inspect(gg)
----------------------------------------------
> ##    去掉多余空格    ####
```

```
> reuters <- tm_map(gg, stripWhitespace)
> reuters[[3]]
```
公安部规定，县级公安机关交通管理部门车辆管理所可以办理本行政辖区内初次申领和增加准驾车型为低速载货汽车、三轮汽车、普通三轮摩托车、普通二轮摩托车、轻便摩托车的机动车驾驶证业务，具体业务范围和办理条件由省级公安机关交通管理部门确定。目前，全国仅有个别县级车辆管理所受条件限制无法开展增加准驾车型为摩托车的考试业务。

```
------------------------------------------------
> ##    全文搜索      ##
> searchFullText(gg[[1]], "是临[时]?改")
[1] FALSE
------------------------------------------------
> ###    查找以某字开头、结尾等的词条    ###
> stemCompletion(gg, c("机", "交", "证"))
                                                    机
                                "机动车驾驶证申领和使用规定"
                                                    交
                                  "交通管理服务群众十项措施"
                                                    证
"证件所有人不应该为自己没有从事的行为承担法律责任"
------------------------------------------------
> ###    中文分词    ###
> txt <- prescindMeta(gg,c("ID"))
> re <- list()
> for (i in 1:nrow(txt)) {
+         re[[i]]<-    CWS(PlainTextDocument(reuters)[[i]],TRUE) ##  包括停用词
+         }
> ###    生成新的文集    ###
> reuters <- Corpus(VectorSource(re))
------------------------------------------------
> ###    元数据管理    ###
> DublinCore(reuters[[2]], "title") <- "建国 60 周年"
> meta(reuters[[2]])
Available meta data pairs are:
   Author           :
   DateTimeStamp: 2010-07-22 01:03:57
   Description    :
   Heading          :  建国 60 周年
   ID                  : 2
   Language        : eng
   Origin           :

------------------------------------------------
> ###    创建词条-文件矩阵
```

```
> dtm <- DocumentTermMatrix(reuters,control = list(minWordLength=2))##最短词两个字
> dtm
A document-term matrix (26 documents, 778 terms)

Non-/sparse entries: 1521/18707
Sparsity           : 92%
Maximal term length: 7
Weighting          : term frequency (tf)
> inspect(dtm[1:2, 3:6])   ## 结果有一定随机性
A document-term matrix (2 documents, 4 terms)

Non-/sparse entries: 3/5
Sparsity           : 62%
Maximal term length: 5
Weighting          : term frequency (tf)

      Terms
Docs 0.016 10 102 105
   1       0   1     1     0
   2       0   2     0     0
----------------------------------------------------------------------
> ##   操作词条-文件矩阵   ##
> ##   1、找出最少出现过 10 次的词条   ##
> findFreqTerms(dtm, 10)
[1] "汽车"    "驾驶"    "部门"    "居民"    "身份证" "使用"    "安全"    "检验"
[9] "公民"
----------------------------------------------------------------
> #   2、找出与"应该"相关度到少达 0.9 的词条   ###
> findAssocs(dtm, "应该", 0.9)
  保密   必须   便捷   表面   参考   常识   承担   读取   负有   复印 复印件
  1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
  公众   过程   核对   核实   经营   快速   留存   切实   权益   确认   确实
  1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
  十分   实践   司法 同一性   外观   伪造   文字   无误 行为人   行业   一致
  1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00
  义务   意识   应该   有损   责任   真伪   职能   只能   作用   法律   社会
  1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   1.00   0.97   0.97
  证件   事务   相应   从事   使用   相关
  0.96   0.95   0.95   0.94   0.92   0.91

> ###   去掉较少词频（保留 80%以上）的词条后   ####
> inspect(removeSparseTerms(dtm, 0.8))
> ##  结果省略
----------------------------------------------------------------
```

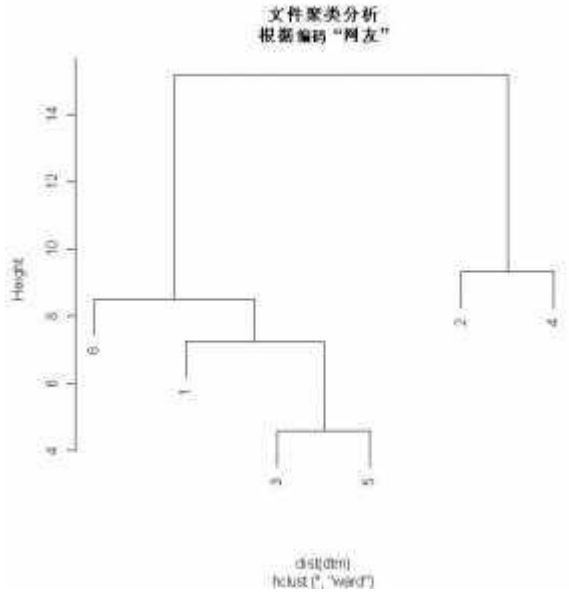
```
> ###   词典   ###   它通常用来表示文本挖掘有关词条
> (d <- Dictionary(c("车辆", "驾驶证")))
[1] "车辆"     "驾驶证"
attr(,"class")
[1] "Dictionary" "character"
> inspect(DocumentTermMatrix(reuters, list(dictionary = d)))
A document-term matrix (26 documents, 1 terms)

Non-/sparse entries: 7/19
Sparsity           : 73%
Maximal term length: 3
Weighting          : term frequency (tf)

      Terms
Docs  驾驶证
   1        0
   2        0
   3        1
   4        0
   5        4
   6        6
   7        4
   8        0
   9        0
   10       3
   11       0
   12       1
   13       0
   14       0
   15       0
   16       4
   17       0
   18       0
   19       0
   20       0
   21       0
   22       0
   23       0
   24       0
   25       0
   26       0

------------------------------------------------------------
> ##   根据词条频率对文件进行聚类分析   ##
```

```
> gg <- RQDA2tm("公安部回应" ,byFile = TRUE)
> reuters <- tm_map(gg, stripWhitespace)
> txt <- prescindMeta(gg,c("ID"))
> re <-    list()
> for (i in 1:nrow(txt)) {
+           re[[i]]<-    CWS(PlainTextDocument(reuters)[[i]],TRUE)
+         }
> reuters <- Corpus(VectorSource(re))
> dtm <- DocumentTermMatrix(reuters,control = list(minWordLength=2))
> reHClust <- hclust(dist(dtm), method = "ward")
> plot(reHClust,main ="文件聚类分析")
> ##  图形省略
> head(txt)
```

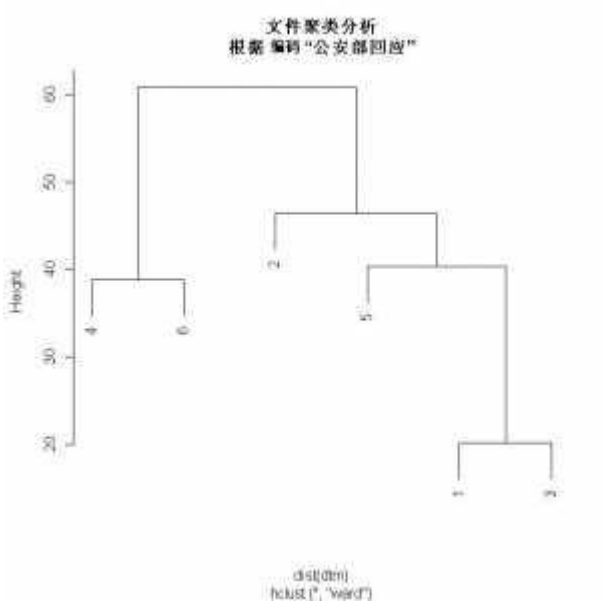
	MetaID	fname	fid	ID
1	0	公安部答复本网网友关于轻微交通违法处罚等 4 问题	1	1
2	0	公安部答复本网网友关于驾龄计算、异地购车上牌、老人驾车等 8 问题	2	2
3	0	公安部答复本网网友关于如何转回农业户口等 3 问题	3	3
4	0	公安部回复本网网友关于驾驶证年检被注销等 3 问题	4	4
5	0	公安部回复人民网网友关于异地缴交通罚款等 4 问题	5	5
6	0	公安部回复人民网网友关于身份证重号错号等 4 问题	6	6

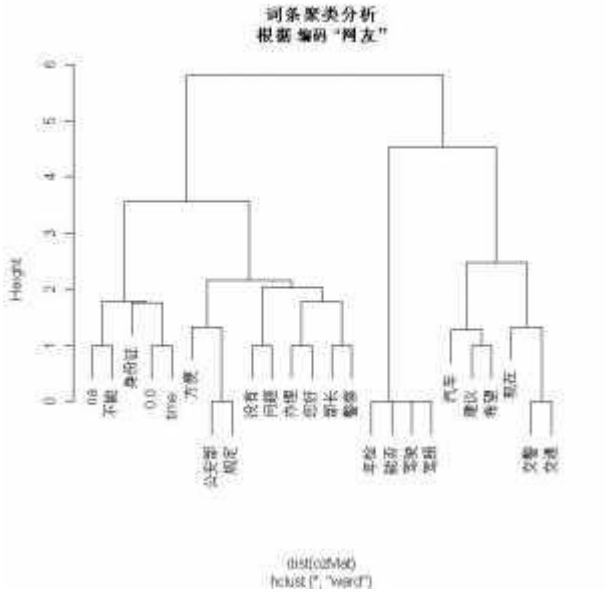
```
> ##   对词条进行分类   ###
> kmeans(dtm, 3)
##   结果省略
```

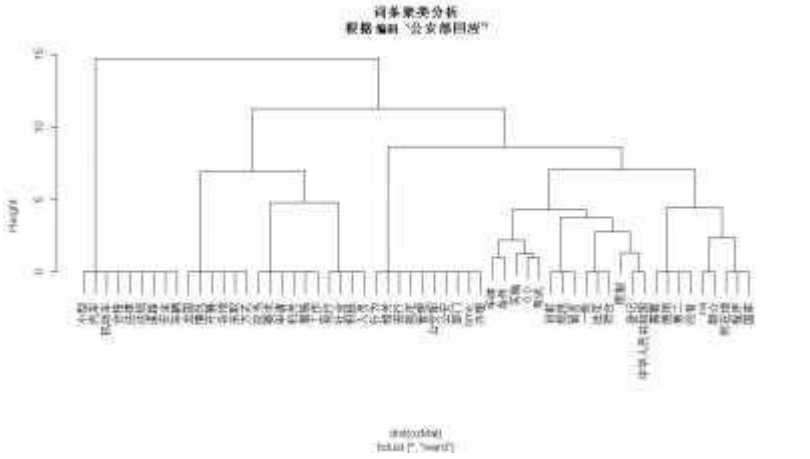
下面是按照以上方法对文档对不同编码进行聚类分析所绘树图：



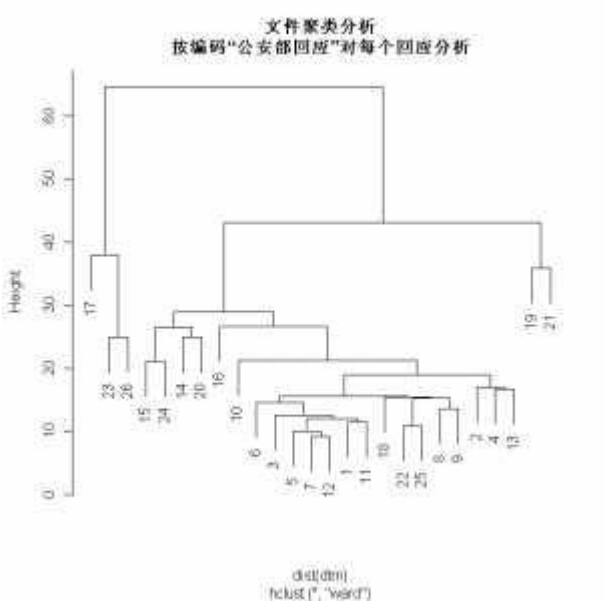
这是用编码“网友”提取相关文档进行分类的结果。其中，数字是指文件 ID（fname ID），后面几个图是对去掉较少词频的词条后的结果。







对上面的数据改为将每条回应为研究对象进行文档聚类分析，结果如下：



综合上面两种聚类分析可以判断：公安部负责对人民网网友进行回应的工作人员有两名，因为每个人的写作用词习惯是比较固定的。

```
> ###     主成分分析   ###
> ozMat <- TermDocumentMatrix(makeChunks(reuters, 50),
+           list(weighting = weightBin,minWordLength=2))
> ##将文档按 50 个字为单位分开形成多个文件，保留最短词两个字
>
> k <- princomp(as.matrix(ozMat), features = 2)
> windows()
> screeplot(k,npcs=6,type='lines')
> windows()
> biplot(k)
```

```
> ###     对词条进行聚类分析   ####
> ozHClust <- hclust(dist(ozMat), method = "ward")
> windows()
> plot(ozHClust,main="词条聚类分析")
> (x <- identify(ozHClust))
> memb <- cutree(ozHClust, k = 5)   #按 5 分类砍树
> memb
> cutree(ozHClust, h = 20)   #按 20 高度砍树
```

其他看上面的链接中的内容，其实生成词条-文件矩阵后还有许多工作可以做，比如用支持向量机进行文件分类、话题分类、根据话题用词频率分析作者所熟悉的行业等等……

运行环境：

```
> sessionInfo()
R version 2.11.0 (2010-04-22)
i386-pc-mingw32

locale:
[1] LC_COLLATE=Chinese (Simplified)_People's Republic of China.936
[2] LC_CTYPE=Chinese (Simplified)_People's Republic of China.936
[3] LC_MONETARY=Chinese (Simplified)_People's Republic of China.936
[4] LC_NUMERIC=C
[5] LC_TIME=Chinese (Simplified)_People's Republic of China.936

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] RQDAtm_0.1-0              rJava_0.8-4               gsubfn_0.5-3
 [4] proto_0.3-8               tm_0.5-3                  RQDA_0.2-0
 [7] gWidgetsRGtk2_0.0-65 gWidgets_0.0-41           RSQLite_0.9-0
[10] DBI_0.2-5

loaded via a namespace (and not attached):
[1] RGtk2_2.12.18 slam_0.1-13   tools_2.11.0
```

第二个小例子：

结合支持向量机对三位房地产大佬在搜房网博客近期文章的分析

搜房网博客链接：
潘石屹
http://blog.soufun.com/blog_132261.htm
王石
http://blog.soufun.com/blog_1525150.htm
任志强
http://blog.soufun.com/blog_1796106.htm

博客原文已经导入 RQDA 数据库中，请先下载、并打开 RQDA2tm_2nd_example.rqda 项目，在 R 的 GUI 中运行如下代码进行分析：

```
> library(e1071)
载入需要的程辑包：class
> ## install.packages(c("e1071")) ## use this command to install e1071 if not installed yet
> ##   提取相关编码的文本     ##
> ws <- RQDA2tm("王石" ,byFile = TRUE)
> rzq <- RQDA2tm("任志强" ,byFile = TRUE)
> psy <- RQDA2tm("潘石屹" ,byFile = TRUE)
```

第一步：先看看这三个人近期最喜欢用的词语是什么。

```
> ##   去掉多余空格并做中文分词   ##
> ws <- tm_map(ws, stripWhitespace)
> txt <- prescindMeta(ws,c("ID"))
> re <- vector()
>   for (i in 1:nrow(txt)) {
+         re[i]<- CWS(PlainTextDocument(ws)[[i]], TRUE) ##  包括停用词
+   }
> ws <- Corpus(VectorSource(re))

> rzq <- tm_map(rzq, stripWhitespace)
> txt <- prescindMeta(rzq,c("ID"))
> re <- vector()
>   for (i in 1:nrow(txt)) {
+         re[i]<- CWS(PlainTextDocument(rzq)[[i]], TRUE) ##  包括停用词
+   }
> rzq <- Corpus(VectorSource(re))

> psy <- tm_map(psy, stripWhitespace)
```

```
> txt <- prescindMeta(psy,c("ID"))
> re <- vector()
>   for (i in 1:nrow(txt)) {
+         re[i]<- CWS(PlainTextDocument(psy)[[i]], TRUE) ##  包括停用词
+   }
> psy <- Corpus(VectorSource(re))


> ##     生成文本-词条矩阵    ##
> dtm_ws <- DocumentTermMatrix(ws,control = list(minWordLength=2,removeNumbers
+             =TRUE))##最短词两个字
> dtm_rzq <- DocumentTermMatrix(rzq,control = list(minWordLength=2,removeNumbers
+             =TRUE))##最短词两个字
> dtm_psy <- DocumentTermMatrix(psy,control = list(minWordLength=2,removeNumbers
+             =TRUE))##最短词两个字
> ##    ------------------------------------------------------------


> top_rzq <-    findFreqTerms(dtm_rzq,sort(dtm_rzq$v ,decreasing = TRUE)[20])
> ##    任志强最喜欢的用词：
> m_rzq <-inspect(dtm_rzq[,top_rzq])
A document-term matrix (6 documents, 21 terms)

Non-/sparse entries: 76/50
Sparsity                    : 40%
Maximal term length: 2
Weighting                   : term frequency (tf)

      Terms
Docs  房价 价格 经济 企业 市场 住房 租赁 发展 改革 公平 机会 就业 没有 努力
   1     3    0    5    1    7    8    5    4    7    0    0    0    0    0
   2    20   31   19    2    5    8    0    3    1    0    0    0    7    0
   3     0    0    2   42    3    0    0    4    0    0    0    0    5    2
   4     5    9    8    1    9    0    0    0    0    0    0    0    7    0
   5     0    1    6    0   22   19   25    3    0    0    0    0    6    0
   6     1    1   11    5    2   13    0   25   48   23   20   21   30   24
      Terms
Docs  社会 幸福 许多 一代 知道 制度 中国
   1     3    0    0    0    0    1    6
   2     4    0    2    0    4    1   15
   3     4    0    1    0    0    4    2
   4     4    0    2    0    3    0    4
   5     4    0    3    0    2    0    4
   6    55   22   21   41   35   21   51
```

> top_psy <- findFreqTerms(dtm_psy,sort(dtm_psy$v ,decreasing = TRUE)[20])
> ## 潘石屹最喜欢的用词：
> m_psy <- inspect(dtm_psy[,top_psy])
A document-term matrix (10 documents, 14 terms)

Non-/sparse entries: 69/71
Sparsity : 51%
Maximal term length: 4
Weighting : term frequency (tf)

	Terms													
Docs	商业	一个	上海	外滩	项目	soho	银河	房地产	房价	市场	土地	政策	力量	建筑
1	12	17	0	0	0	1	0	4	1	4	1	3	0	0
2	17	6	15	27	21	6	0	1	0	2	2	5	0	8
3	0	8	0	0	2	0	0	11	0	2	10	11	0	0
4	16	6	0	0	2	13	13	0	0	5	0	2	0	6
5	2	10	0	0	6	9	0	0	0	0	0	0	2	1
6	4	5	3	0	1	0	0	17	13	37	15	12	0	0
7	0	5	0	0	1	0	0	0	0	1	0	0	13	0
8	0	1	0	0	8	7	4	0	0	0	0	0	0	12
9	15	1	0	0	4	18	14	0	1	9	0	11	0	4
10	0	7	0	0	0	0	0	0	0	0	1	0	13	0

> top_ws <- findFreqTerms(dtm_ws,sort(dtm_ws$v ,decreasing = TRUE)[20])
> ## 王石最喜欢的用词
> m_ws <- inspect(dtm_ws[,top_ws])
A document-term matrix (3 documents, 29 terms)

Non-/sparse entries: 30/57
Sparsity : 66%
Maximal term length: 6
Weighting : term frequency (tf)

	Terms												
Docs	冰川	超过	穿越	淡水	登山	海拔	攀登	融化	速度	威胁	喜马拉雅山脉	珠峰	
1	18	3	6	3	3	3	4	8	3	3	3	4	
2	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	0	0	

	Terms												
Docs	处理	东京	发展	焚烧	建筑	垃圾	比赛	等级	了解	日本	森严	体重	喜欢	相扑
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	4	4	3	8	3	11	0	0	0	1	0	0	0	0
3	0	0	0	0	0	0	3	4	4	7	3	3	4	10

Terms

```
Docs 运动 秩序 追求
   1    0    0    0
   2    0    0    0
   3    4    6    3
```

结论：

由此看来王石仍然在到处玩儿，怪不得从万科 A 到万科 B，再到万科债券 08G1、08G2 都跌的一塌糊涂。

任志强的兴趣仍在研究国家政策，忧国忧民啊！

老潘的工作重点已经转到上海的商业地产了。

第二步：看能不能用支持向量机方法建模来判断某些话是谁说的。

首先要建立建模用的训练集和测试集：

```
> ##   合并三人的常用词条   ##
> hh <- union(union(top_rzq,top_psy),top_ws)

> ##   生成训练用的数据     ##
> tt <- matrix(data = NA, nrow = dim(m_rzq)[1], ncol = length(hh))
> tt <- as.data.frame(tt)
> colnames(tt) <- hh
> tt[,top_rzq] <- m_rzq
> tt[,"作者"]   <- c("任志强")

> tt1 <- matrix(data = NA, nrow = dim(m_psy)[1], ncol = length(hh))
> tt1 <- as.data.frame(tt1)
> colnames(tt1) <- hh
> tt1[,top_psy] <- m_psy
> tt1[,"作者"]   <- c("潘石屹")
> tt <- rbind(tt,tt1)

> tt1 <- matrix(data = NA, nrow = dim(m_ws)[1], ncol = length(hh))
> tt1 <- as.data.frame(tt1)
> colnames(tt1) <- hh
> tt1[,top_ws] <- m_ws
> tt1[,"作者"]   <- c("王石")
> tt <- rbind(tt,tt1)

> tt[is.na(tt)]<-0
> tt[,"作者"] <- factor(tt[,"作者"])

> ##     用支持向量机进行建模   ##
> model <- svm(作者  ~ ., data = tt[c(1:4,7:14,17:18),], kernel = "sigmoid")
> summary(model)
Call:
```

```
svm(formula = 作者 ~ ., data = tt[c(1:4, 7:14, 17:18), ], kernel = "sigmoid")
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  sigmoid
       cost:  1
      gamma:  0.01666667
     coef.0:  0
Number of Support Vectors:  10
 ( 4 4 2 )
Number of Classes:  3
Levels:
 潘石屹 任志强 王石

> ##    模型拟合测试  ##
> ##  训练集（样本内）拟合      ##
> pred <- predict(model, tt[c(1:4,7:14,17:18),1:length(hh)])
> table(pred, tt[c(1:4,7:14,17:18),"作者"])

pred      潘石屹 任志强 王石
  潘石屹       7      0    0
  任志强       1      4    0
  王石         0      0    2

> ##  测试集（样本外）预测      ##
> pred <- predict(model, tt[c(5:6,15:16,19),1:length(hh)])
> table(pred, tt[c(5:6,15:16,19),"作者"])

pred      潘石屹 任志强 王石
  潘石屹       2      0    0
  任志强       0      2    0
  王石         0      0    1
```

样本内有一个判别错误，样本外预测全中，效果不错！那么现在随机找两段他们的文字，看模型是否能判断出是谁的文章。

```
> test <- c("中午听说中国一支民间登山队日前在道拉吉里峰遭遇山难，得知深航机长李斌不幸遇难。心里非常难过。
+       深圳登山运动大概在 2000 年前后开始蓬勃发展，我从 00 年玉珠峰开始登山，和李斌是在 07 年卓奥友峰认识，我们前后脚冲顶，李斌所在队伍由于当天暴风雪未能攻顶成功。08 年我参加的登山队攀登了希夏邦马峰，途径拉萨，巧遇再次冲顶卓奥友的李斌，当时见面都很高兴，互相祝福。下山后得知，李斌顺利登顶卓奥友，当时心里替他高兴。对李斌机长的印象也是一直感觉非常顽强执着，热爱着登山这项运动。
+       今天得知李斌机长遇难的消息，非常突然，心里也是非常难过。
+       大本营队员们得知消息，大家心里也是很不好受，默默祝福其他受伤队员安全下撤。
```

```
")
> ##   生成文档并进行中文分词
> test <- Corpus(VectorSource(test))
> test <- tm_map(test, stripWhitespace)
> txt <- prescindMeta(test,c("ID"))
> re <- vector()
>   for (i in 1:nrow(txt)) {
+         re[i]<- CWS(PlainTextDocument(test)[[i]], TRUE) ##  包括停用词
+   }
> test <- Corpus(VectorSource(re))
> dtm_test <- DocumentTermMatrix(test,control = list(minWordLength=2,removeNumbers
+           =TRUE))##最短词两个字
> ##   建立预测数据框
> test_h   <-   intersect(Terms(dtm_test),hh)
> tt1 <- matrix(data = NA, nrow = dim(dtm_test)[1], ncol = length(hh))
> tt1 <- as.data.frame(tt1)
> colnames(tt1) <- hh
> test_tt <- inspect(dtm_test)
> tt1[,c(test_h)] <- test_tt[,c(test_h)]
> tt1[is.na(tt1)]<-0
> ##  预测判断
> predict(model, tt1)
    1
王石
Levels:  潘石屹  任志强  王石
```

完全正确，这是《珠峰零公里口述之二十二》**(2010-5-18 16:30:58)**中的一段文字。

那么下面这段呢？

```
> test <- c("大多数人将降房价的希望寄托在房屋持有税的征收上，不管这个税是叫物业税、叫房产税、还是叫个什么其他的名称，总之多数人（也许还包括政府管理部门）都认为中国的房价过高原因在于持有住房（其他房屋已有房产税的征收了）的成本太低，从而造成大量的人在炒房，导致了市场中的供不应求。")
> ##  同样运行上面的代码
> ##  生成文档并进行中文分词
> test <- Corpus(VectorSource(test))
> test <- tm_map(test, stripWhitespace)
> txt <- prescindMeta(test,c("ID"))
> re <- vector()
>   for (i in 1:nrow(txt)) {
+         re[i]<- CWS(PlainTextDocument(test)[[i]], TRUE) ##  包括停用词
+   }
> test <- Corpus(VectorSource(re))
> dtm_test <- DocumentTermMatrix(test,control = list(minWordLength=2,removeNumbers
```

```
+              =TRUE))##最短词两个字
> ##   建立预测数据框
> test_h   <-   intersect(Terms(dtm_test),hh)
> tt1 <- matrix(data = NA, nrow = dim(dtm_test)[1], ncol = length(hh))
> tt1 <- as.data.frame(tt1)
> colnames(tt1) <- hh
> test_tt <- inspect(dtm_test)
> tt1[,c(test_h)] <- test_tt[,c(test_h)]
> tt1[is.na(tt1)]<-0
> ##  预测判断
> predict(model, tt1)
```

任志强
Levels: 潘石屹 任志强 王石

又猜对了，太棒了！快赶上章鱼哥了！这是《不能把希望建立在冰上》**(2010-5-27 14:11:34)**开头的一段。

至此，模型建立还是相当成功的，需要注意的是，支持向量机的内核选择是很关键的，我这里用的是"sigmoid"，线性核的表现也不错。