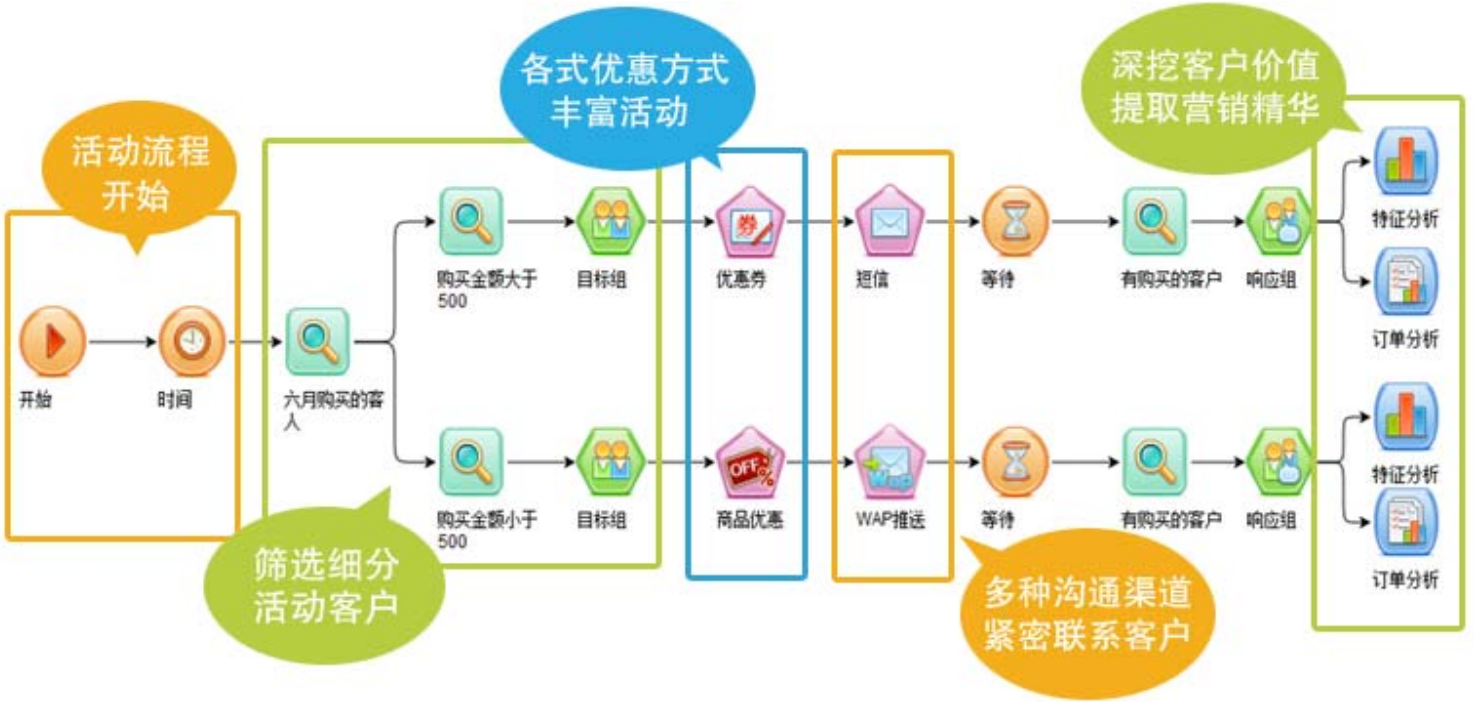
基于RHadoop的关联规则挖掘

陈逸波@华院数云

目录

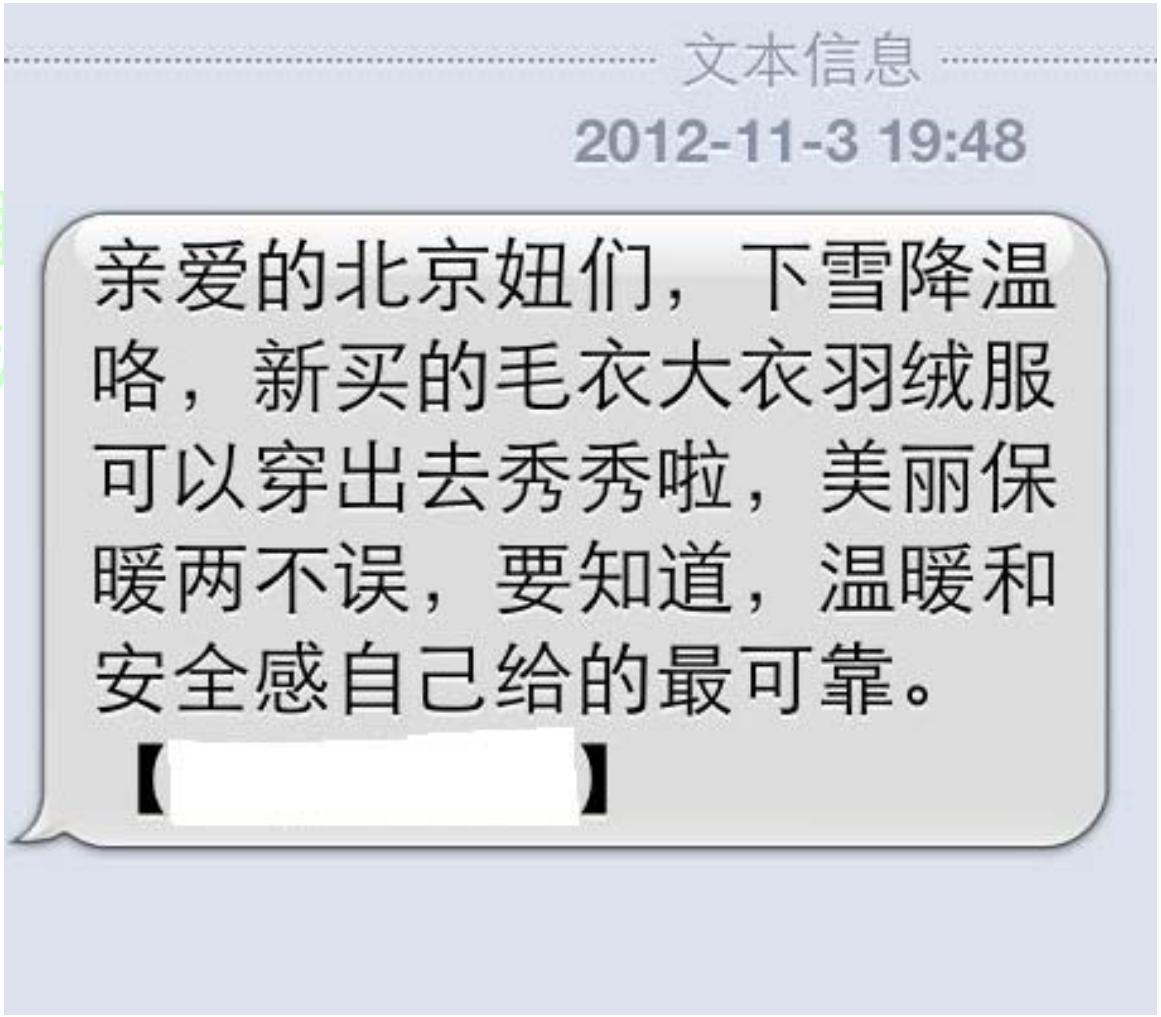
业务背景——CRM

- CRM
- 客户关系管理 客户关系营销
- 精准营销 数据库营销
- ……



业务背景——CRM

- CRM
- 客户关系管理
- 精准营销 数据
- ……



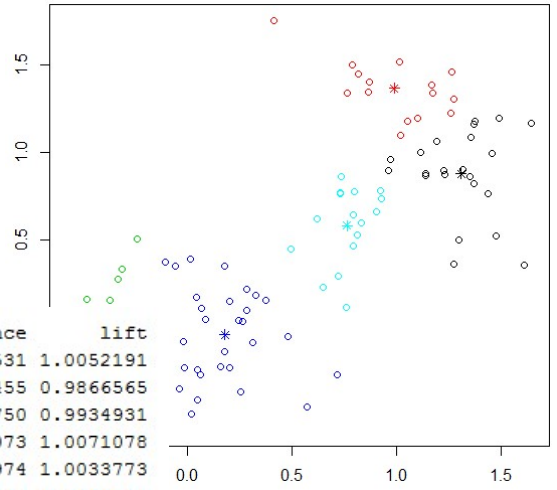
业务背景——CRM

- CRM
- 客户关系管理 客户
- 精准营销 数据库营
- ……

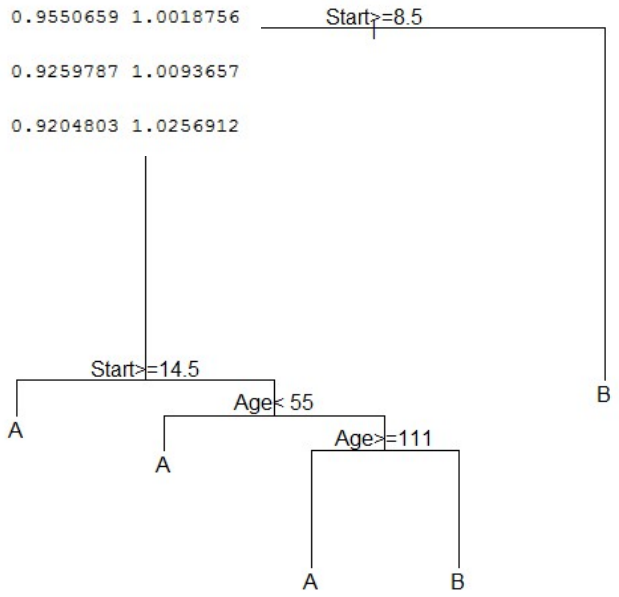
客户

业务背景——典型案例

- 聚类　　客户细分
- 关联　　交叉销售
  关联推荐
- 分类　　流失预警
  营销响应



	lhs		rhs	support	confidence	lift
1	{a_70}	=>	{a_66}	0.5606650	0.9582531	1.0052191
2	{a_62}	=>	{a_63}	0.6050735	0.9051455	0.9866565
3	{a_62}	=>	{a_66}	0.6331027	0.9470750	0.9934931
4	{a_8}	=>	{a_63}	0.6413742	0.9239073	1.0071078
5	{a_8}	=>	{a_66}	0.6639982	0.9564974	1.0033773
6	{a_60}	=>	{a_66}	0.8136849	0.9516307	0.9982720
7	{a_111}	=>	{a_63}	0.8219565	0.9159062	0.9983862
8	{a_63}	=>	{a_66}	0.8706646	0.9490705	0.9955863
9	{a_66}	=>	{a_63}	0.8706646	0.9133376	0.9955863
10	{a_63, a_70}	=>	{a_66}	0.5191638	0.9550659	1.0018756
11	{a_66, a_70}	=>	{a_63}	0.5191638	0.9259787	1.0093657
12	{a_60, a_62}	=>	{a_111}	0.5415421	0.9204803	1.0256912



业务背景——关联推荐

- 个性化推荐　协同过滤
- 看了还看，看了会买，买了还买

模型及算法——关联规则

- 形式

  A => B

- 指标

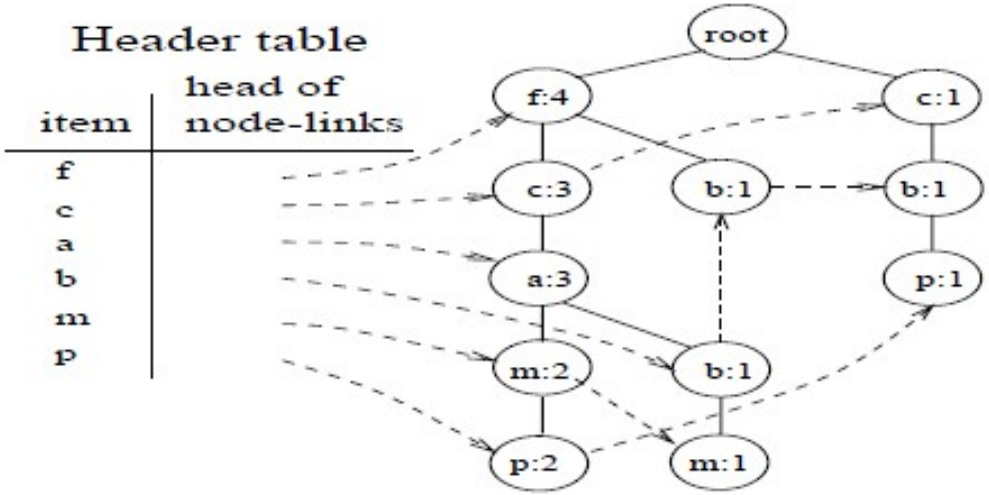
  支持度 support　　P(AB)

  置信度 confidence　　P(AB)/P(A)

  提升度 lift　　P(AB)/(P(A)*P(B))

模型及算法——频繁模式挖掘

- 频繁模式挖掘算法（frequent pattern）
  - Apriori
    - 逐层迭代连接产生候选，利用先验信息进行剪枝
  - FP-Growth
    - 将信息压缩为FP树结构，在树中进行递归的挖掘

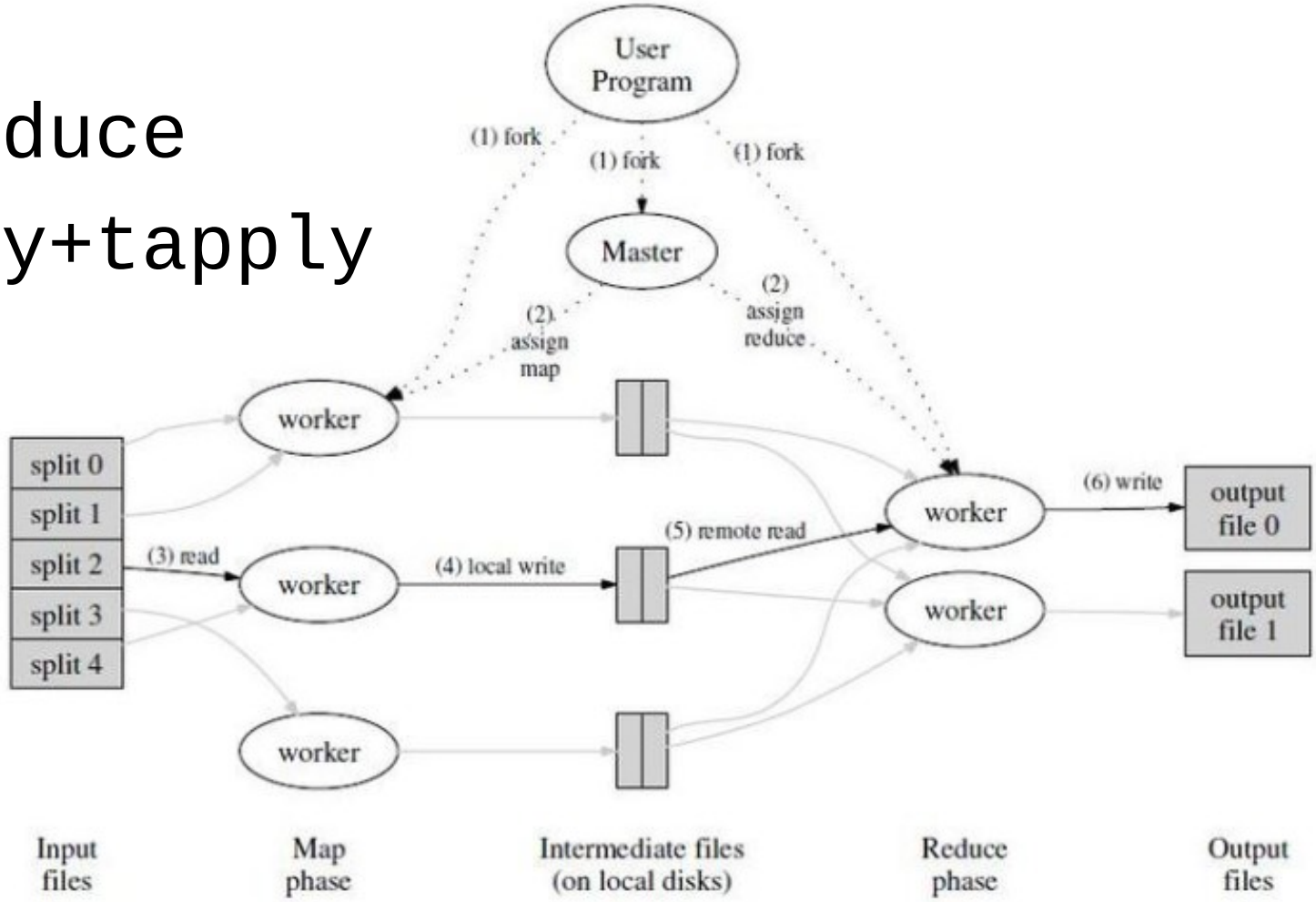


模型及算法——算法流程

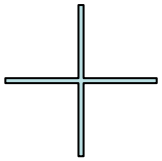
- 输入数据　　dp_id, tid, item
- 预处理　　t_1, x_11 + x_12 + ...
- FPG　　([x1, x7], 2), ([x2, x4, x5], 3), ...
- 生成规则　　x1, NULL, NULL, x7, supp, conf, ...
  x2, x4, NULL, x5, supp, conf, ...
  x2, x5, NULL, x4, supp, conf, ...
  ......

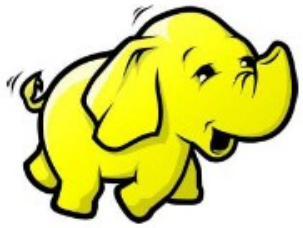
工具的使用——Hadoop

- HDFS
- MapReduce
- lapply+tapply



工具的使用——RHadoop





rmr	**MapReduce**
rhdfs	**HDFS**
rhbase	**HBASE**

工具的使用——rmr

- to.dfs()
- from.dfs()
- mapreduce(input,
  output,
  map,
  reduce,
  ...)
- keyval()

工具的使用——一个例子

```
system('start-all.sh')
system('hadoop dfsadmin -safemode leave')
require(rmr)

lines <- c('Are you sleeping,',
           'are you sleeping?',
           'Brother John, Brother John?',
           'Morning bells are ringing,',
           'Morning bells are ringing,',
           'Ding, Ding Dong, Ding, Ding Dong')
lines_dfs <- to.dfs(lines)
from.dfs(lines_dfs)
s <- '[[:punct:][:space:]]+'
table(unlist(strsplit(lines, s)))
```

工具的使用——一个例子

```
wordcount <- function(input, output=NULL,
                      s='[[:punct:][:space:]]+'){
  mapreduce(input=input, output=output,
    map=function(k, v){
      v2=unlist(strsplit(x=v, split=s))
      lapply(v2, function(w){keyval(w, 1)})},
    reduce=function(k, v){
      keyval(k, sum(unlist(v)))},
    combine=T)}

wc <- wordcount(input=lines_dfs)
do.call(rbind, from.dfs(wc))
```

工具的使用——一个例子

```
wordcount_vec <- function(input, output=NULL,
                          s='[[:punct:][:space:]]+'){
  mapreduce(input=input, output=output,
    map=function(k, v){
      v2=unlist(strsplit(x=unlist(v), split=s))
      lapply(v2, function(w){keyval(w, 1)})},
    reduce=function(k, v){
      keyval(k, sum(unlist(v)))},
    combine=T,
    vectorized=T)}

wc_vec <- wordcount_vec(input=lines_dfs)
do.call(rbind, from.dfs(wc_vec))
```

工具的使用——一个例子

```
wordcount_vec <- function(input, output=NULL,
                          s='[[:punct:][:space:]]+'){
  mapreduce(input=input, output=output,
    map=function(k, v){
      v2=unlist(strsplit(x=unlist(v), split=s))
      lapply(v2, function(w){keyval(w, 1)})},
    reduce=function(k, v){
      keyval(k, sum(unlist(v)))},
    combine=T,
    vectorized=T)}

wc_vec <- wordcount_vec(input=lines_dfs)
do.call(rbind, from.dfs(wc_vec))
```

工具的使用——几处细节

- 向量化

  `vectorized`

- 输入输出格式

  ```
  input.format  output.format
  text  native  json  csv  ...
  ```

- 后台参数

  `backend.parameters`

- `drop=F`
- `rmr2`

后续工作

1、mahout fpg -i -o -s -k -2
   - Top-K
   - 闭频繁模式
2、多层关联规则，协同过滤
3、分类模型

谢谢！

陈逸波@华院数云