姓　名	王凯
学　号	201912190060

成　绩	
评卷人	

中 南 财 经 政 法 大 学

研 究 生 课 程 考 试 试 卷

（《商务统计建模与决策》课程论文）

论文题目　基于数据挖掘的某网站综合分析

课程名称　商务统计建模与决策

完成时间　2020 年 6 月 15 日

专业年级　大数据商务统计专硕 2019 级

注：研究生必须在规定期限内完成课程考试论文，并用 A4 页面打印，加此封面装订成册后，送交评审教师。教师应及时评定成绩，并至迟在下学期开学后两周内将此课程论文及成绩报告单一并交本单位研究生秘书存档。（涉及外单位的，由研究生秘书转交学生所在单位研究生秘书存档）

基于数据挖掘的某网站综合分析

1 研究背景

如今，数据挖掘应用在日常生活中无处不在，这给我们的生活带来了极大的便利，而要产生这些数据挖掘服务，最重要的就是“数据”。4G 和 5G 等网络技术的快速发展与普及，给互联网行业注入了源源不断地新的活力，这给社交网络、在线平台以及电子商务网站等网络应用的发展奠定了基础。对于这些网络应用，它们最为重要的特点就是数据驱动，通过数据来进行决策，通过数据来服务用户。用户既是数据的产生者，同时又是数据的受益者。随着网络技术的不断发展和普及，互联网用户的数量不断攀升，随之产生的数据也不断增加。并且在 2010 年以后，智能手机的快速普及，使得人们上网更加方便也更加普遍，这更使得互联网信息呈现出了爆炸增长的态势。根据有关部门资料显示，近些年在互联网产生的数据占到了所有互联数据的 80% 以上，而且其还正在高速增长，并没有出现减缓的迹象，而我们也真正迎来了“大数据时代”。这爆炸似的信息增长给互联网行业带来了前所未有的机遇，通过挖掘数据中的信息并对其进行处理，可以将之服务于用户，给用户带来更好的体验。然而这海量的数据也给数据处理技术提出了严峻的挑战，其最大的挑战就是“信息过载”问题，即用户没有办法从大量的信息中获取到自己需要的信息的问题。为了解决这个问题，数据挖掘技术应运而生。

在本案例中，我们首先通过建立 ARIMA 模型来预测未来七天网站的访问量，根据访问量的多少我们可以制定相应的营销策略。接下来我们会通过基于物品的协同过滤算法建立推荐系统，通过分析网站之间的相似度，然后将类似网站推荐给用户。最后我们将根据关联规则算法，将用户经常一起访问的网站排列在一起，方便用户查找浏览。

2 数据准备

由于本案例给出的是 sql 文件，需要通过 Mysql 数据库执行后才能导出我们相应的数据，但是由于 sql 文件高达 177M，在本地 Mysql 中执行缓慢且中间有部分插入语句报错，会导致数据的丢失。因此通过 python 软件来进行处理，将 sql 语法替换为空格，通过清洗，最终得到 464243 条数据记录，每条记录含有 21 个属性特征，最终我们建立数据字典，并保存为 clear.csv 文件，接下去的分析将全部基于这个清洗好的数据。

3 使用 ARIMA 算法预测网站访问量

3.1 ARIMA 算法介绍

如果一个时间序列$\{Y_t\}$ 的 d 次差分$W_t = \nabla^d Y_t$ 是一个平稳的 ARIMA 过程，则称$\{Y_t\}$ 为自回归差分移动平均模型。如果$\{W_t\}$ 服从 ARMA (p,q) 模型，那么可以称$\{Y_t\}$ 是 ARIMA(p,d,q) 过程。在实际建模过程中，研究者们通常将 d 的值取 {0,1,2}。下面考虑 ARIMA(p,1,q) 过程，令$W_t = Y_t - Y_{t-1}$, 模型的具体形式为：$W_t = \varphi_1 W_1 + \varphi_2 W_2 + \cdots + \varphi_p W_{t-p} + e_t - \theta_1 e_{t-1} - \theta_2 e_{t-2} - \cdots - \theta_q e_{t-q}$

3.2 ARIMA 实证分析

3.2.1 数据来源及预处理

为了研究网站访问量的变化规律，我们需要获得 2016 年 9 月 -2017 年 3 月该网站的每天访问数据，故我们对清洗后的数据的 “data_time” 特征进行处理，统计网站每天的访问次数，然后基于该数据进行 ARIMA 建模。

```
#读入数据并进行预处理
data <- read.csv("clear.csv",encoding = "UTF-8")
date <- data$data_time
library(lubridate)
date <- ymd_hms(date)
date <- round_date(date,"day")
df <- table(date)
df <- data.frame(df)
head(df,3)
```

```
##         date Freq
## 1 2016-09-01 1232
## 2 2016-09-02 2203
## 3 2016-09-03 2248
```

3.2.2 平稳性检验

首先，本文为了初步探析 2016 年 9 月 -2017 年 3 月这 7 个月间该网站的访问量波动情况，故以天为单位，绘制了时序图，得到如下图 1-1 所示的结果。

```
par(mfrow=c(1,1))
t <- df$date #转化为日期
hits <- as.ts(df$Freq)   #转化为时间序列数据
library(zoo)        #调用时间序列包
t1.ts <- zoo(hits,t)
#访问量的时序图
plot(t1.ts,xlab = "Image 1-1:The timing diagram of hits",ylab = "hits")
```

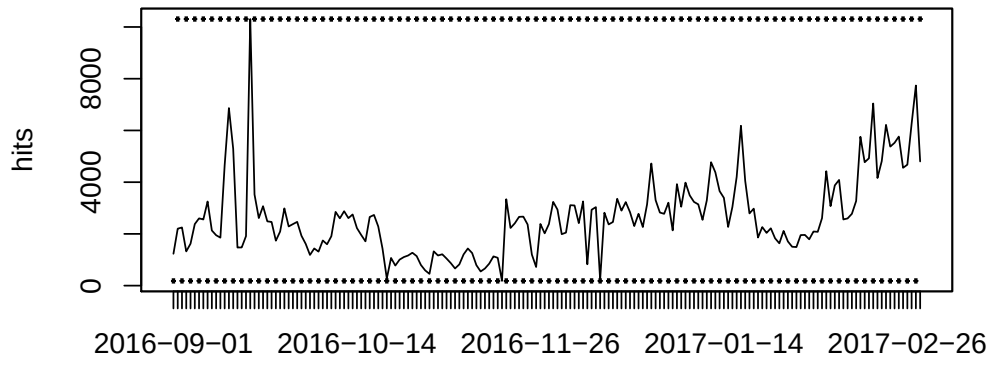

Image 1−1:The timing diagram of hits

从图 1-1 的时序图中可以看出，该网站的访问量并没有一个确定的规律，从总体上看，似乎存在一定的上升趋势。而且从时序图中也可以明显的看出，该网站的访问量数据并不平稳，为了进一步从统计角度验证访问量序列是否为非平稳序列，我们需要进行单位根检验，得到如下的结果：

```
# 单位根检验
library(tseries)
adf.test(hits)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  hits
## Dickey-Fuller = -1.8, Lag order = 5, p-value = 0.7
## alternative hypothesis: stationary
```

从结果中可知，单位根检验的 P 值为 0.6643，明显大于显著性水平 0.05，故我们不可以拒绝非平稳的原假设，认为访问量序列非平稳。因此如果我们要利用此数据进行时间序列建模时不合理的，所以在建模之前需要对其进行差分处理使之平稳化。对此本文对其进行一阶差分处理，得到了访问量一阶差分后的数据，并绘制了相应的时序图，得到如图 1-2 所示的结果：

```
# 进行差分
hits.diff <- diff(hits)
t2.ts <- zoo(hits.diff,t)
# 一阶差分后网站的访问量
plot(t2.ts,xlab = "Image 1-2:The timing diagram of diff(hits)",ylab = "
    hits")
```

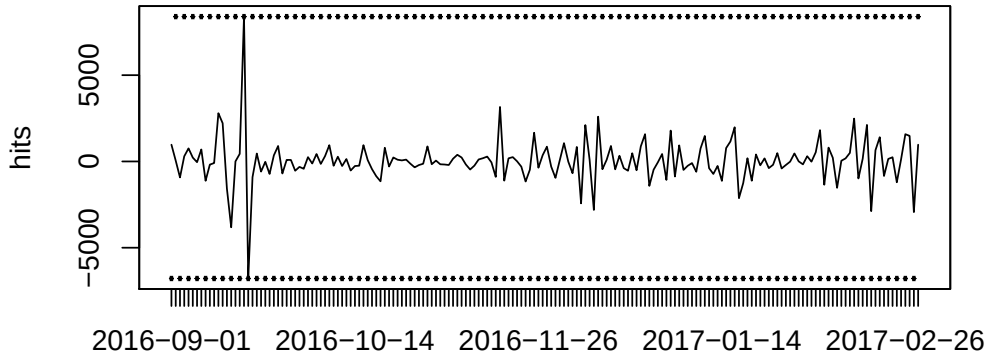

Image 1−2:The timing diagram of diff(hits)

从图 1-2 中可以发现，差分后的访问量序列不再具有明显的波动特征，而是在 0 值附近波动，不再具有显著的起伏，故可判断差分后的访问量数据具有平稳性。为了进一步确认差分后的访问量序列是平稳序列，故对差分后的序列进行单位根检验，得到如下的结果：

```r
# 一阶差分序列单位根检验
adf.test(hits.diff)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  hits.diff
## Dickey-Fuller = -8.1, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

根据检验结果可知，差分后的序列单位根检验的 P 值为 0.01，明显小于显著性水平 0.05，故拒绝非平稳的原假设，认为差分后的访问量序列为平稳序列，可以进行接下去的分析。

3.2.3 白噪声检验

接下来为了判断差分后的序列是否可以用于时间序列建模，还需进一步进行白噪声检验：

```r
# 纯随机性检验
Box.test(hits.diff, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  hits.diff
## X-squared = 15, df = 1, p-value = 0.0001
```

从结果中可知，白噪声检验的 P 值为 0.0001，明显小于显著性水平 0.05，故拒绝原假设，所以该序列为非随机序列，可以用于建模分析。

3.2.4 构建 ARIMA 模型

ARIMA 模型的实质是差分与 ARMA 模型的结合，对差分后平稳的序列进行自回归移动平均拟合。由上文可知，一阶差分后的网站访问量数据为平稳的非白噪声序列,，所以对差分后的序列建立 ARMA（p,q）模型。但在建立模型之前，我们首先得确定 p、q 的大小，故根据 AIC 和 BIC 准则进行模型定阶：

```r
library(forecast)
model <- auto.arima(hits)
summary(model)
```

```
## Series: hits
## ARIMA(2,1,2)
##
## Coefficients:
##          ar1    ar2     ma1    ma2
##       0.914  -0.41  -1.486  0.670
## s.e.  0.194   0.09   0.198  0.156
##
## sigma^2 estimated as 1185719:  log likelihood=-1471
## AIC=2951   AICc=2951   BIC=2967
##
```

```
## Training set error measures:
##                    ME RMSE   MAE    MPE  MAPE   MASE     ACF1
## Training set 62.48 1073 710.6 -18.89 40.15 0.9265 -0.01212
```

根据模型定阶结果，确定为 ARIMA(2,1,2)，得到模型得参数估计结果为：diff(hits)=0.9141AR(1)-0.41AR(2)-1.4855MA(1)+0.6696MA(2)

3.2.5 模型检验与评估

构建 ARIMA 模型后，我们还需要对模型的残差序列进行白噪声检验，只有当残差序列是白噪声序列时，才说明 ARIMA 模型提取完访问量序列中所含的信息，说明拟合的模型有效。接下来我们对残差序列绘制时序图和纯随机性检验。

```
# 残差的时序图与纯随机性检验
r <- model$residuals
rt <- zoo(r,t)
plot(rt,xlab = "Image 1-3:The timing diagram of residual",ylab = "hits")
```

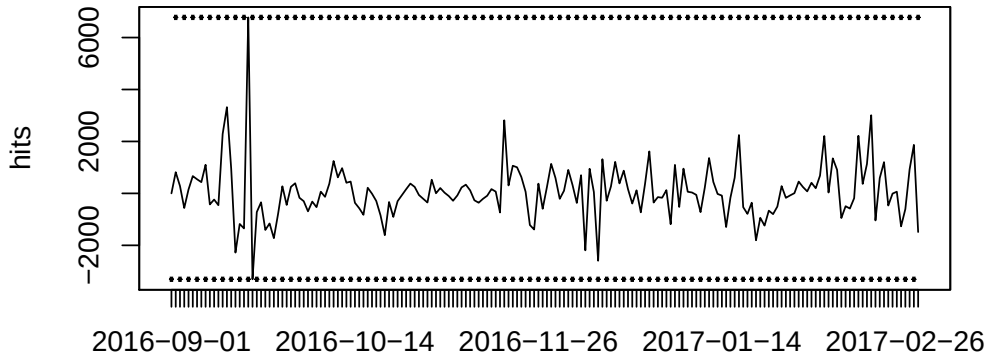

Image 1−3:The timing diagram of residual

```
Box.test(r,type="Ljung-Box",lag=1)
```

```
##
##  Box-Ljung test
##
## data:  r
## X-squared = 0.026, df = 1, p-value = 0.9
```

根据纯随机性检验的 P 值可知，其明显大于显著性水平 0.05，故不拒绝原假设，所以该序列为纯随机序列，ARIMA(2,1,2) 模型已提取完序列中的信息，模型是有效的。

3.2.6 模型预测

使用建立好的 ARIMA(2,1,2) 模型来对未来七天网站的访问量进行预测：

```r
pred <- predict(model,n.ahead = 7)
pred$pred
```

```
## Time Series:
## Start = 177
## End = 183
## Frequency = 1
## [1] 4971 5334 5596 5687 5663 5603 5559
```

根据预测结果，未来 7 天网站的访问量为 [4971,5334,5596,5687,5663,5603,5559]。

4 使用协同过滤算法实现网站的智能推荐

4.1 协同过滤算法介绍

“协同过滤” 算法最开始是在 90 年代中期开发推荐系统时被提出来的，它是一种基于领域的算法，它通过集体的智慧做出最优的推荐。常见的协同过滤算法有两种，一种是基于用户的协同过滤，另一种是基于物品的协同过滤。基于用户的协同过滤，首先计算用户之间的相似度，然后根据用户之间的相似度给用户推荐那些和他有共同兴趣爱好的用户所喜欢的物品。基于物品的协同过滤，基本思想是计算物品之间的相似度，然后根据物品的相似度和用户的历史行为给用户生成推荐列表。

4.2 相似度计算

4.2.1 夹角余弦

$$\mathrm{sim}_{lm} = \frac{\sum_{k=1}^{n} x_{k1} x_{km}}{\sqrt{\sum_{k=1}^{n} x_{k1}^2}}$$

夹角余弦相似度的取值范围为 [-1,1]，当余弦值的绝对值越接近于 1 时，表明相似度程度越高；越接近于 0 时，相似程度越低。

4.2.2 杰卡德相似系数

$$J\left(A_1, A_m\right) = \frac{\left|A_1 \cap A_m\right|}{\left|A_1 \cup A_m\right|}$$

杰卡德相似系数主要用于计算分类数据的相似度，其中$A_1 \cup A_m$ 表示喜欢物品 1 与喜欢物品 M 的用户总数，$A_1 \cap A_m$ 表示同时喜欢物品 1 和物品 M 的用户数，它的取值范围为 [0,1]。

4.2.3 相关系数

$$\mathrm{sim}_{lm} = \frac{\sum_{k=1}^{n} \left(x_{k1} - A_1\right)\left(x_{km} - A_m\right)}{\sqrt{\sum_{k=1}^{n} \left(x_{k1} - \overline{A_1}\right)^2} \sqrt{\sum_{k=1}^{n} \left(x_{km} - \overline{A_m}\right)^2}}$$

相关系数的取值也为 [-1,1]，相关系数的绝对值越大，表明相似度越高。

4.3 推荐系统建模的流程和步骤

1. 从业务系统中获得原始数据
2. 通过数据抽取技术获得用户访问日志
3. 对数据进行探索分析和预处理（数据去重、数据变换以及数据分类）
4. 将清洗好的数据进行建模，常用的推荐算法有协同过滤推荐、热点推荐以及随机推荐等
5. 模型建立完毕，进行模型评价，如果模型满足要求，则可以进行测试；如果模型不满足要求，则需要进行模型优化和重构

4.4 构建智能推荐系统

4.4.1 数据来源及预处理

为了实现网站的智能推荐服务，一般需要两个特征，一个特征是用来区分用户，一个特征用来区分网站，经过数据筛查，我们从 clear 数据集中选择 “IP” 作为区分用户的属性特征，作为用户的唯一标识，“page_path” 作为区分网站的属性特征，作为网站的唯一标识。

```
# 统计九月份的访问数据总量
count <- sum(df$Freq[1:30])
dr <- data[1:count,]
#根据推荐模型，一般为两个特征或者三个特征
dr <- dr[c("ip","page_path")]
head(dr)
```

```
##                  ip                   page_path
## 1  '183.63.102.193'             '/sm/736.jhtml'
## 2   '58.249.112.60'         '/notice/757.jhtml'
## 3    '125.208.4.78'             '/sj/638.jhtml'
## 4    '125.208.4.78'             '/sj/638.jhtml'
## 5   '157.55.39.254'  '/secondtipdm/index.jhtml'
## 6    '125.208.7.80'             '/sj/638.jhtml'
```

根据之前建立的 df 数据框，我们可以知道 2016 年 9 月的总访问量数据，根据这个数据，我们可以得到包含 85432 条记录的数据集，但是数据集中还存在缺失值、异常值和重复值，故需要进行处理。

```
#直接删除存在缺失值的样本
newdr <- na.omit(dr)
#删除重复的样本
newdr <- unique(newdr)
# 得到存在异常值的样本的，然后将其删除index
strange <- newdr$page_path[8]
index1 <- newdr$page_path==strange
newdata <- newdr[-which(index1),]
dim(newdata)
```

```
## [1] 37916     2
```

经过删除缺失值、重复值和异常值的操作后，原本 85432 条数据还剩 37916 条，接下来我们用这 37916 条数据进行推荐系统建模。

4.4.2 数据结构转换

在使用推荐系统建模时，我们需要的二元数据矩阵，故进行数据转换：

```
library(plyr)
library(recommenderlab)
# 将数据转换为二元型数据，即模型的输入数据集0-1
info <- as(newdata, "binaryRatingMatrix")
```

通过分析可知，转换后的数据变成了 6130 名用户对 856 个网站的二元矩阵，接下来我们将根据这个二元矩阵建立推荐系统模型。

4.4.3 建立推荐系统模型

由于本数据集用户的数量远大于网站的数量，故建立基于物品的协同过滤模型。

```
# 采用基于物品的协同过滤算法构建模型，相似度采用杰卡德相似系数
info.re <- Recommender(info, method = "IBCF")
```

4.4.4 模型预测

通过基于物品的的协同过滤算法构建的推荐系统，我们可以预测得到所有用户的前五个感兴趣的网站，并展示前两个用户的推荐结果：

```
# 利用模型对原始数据集进行预测并获得推荐长度为的结果5
info.p <- predict(info.re, info, n = 5)
print(as(info.p, "list")[1:2])
```

```
## $‘ ’1.12.55.67’‘
## [1] ” ’/ts/index.jhtml’”   ” ’/yxzp/index.jhtml’” ” ’/jszz/index.
    jhtml’”
## [4] ” ’/stpj/index.jhtml’” ” ’/ts/654.jhtml’”
##
## $‘ ’1.160.114.183’‘
## [1] ” ’/zytj/index.jhtml’” ” ’/ts/661.jhtml’”     ” ’/yxzp/index.
    jhtml’”
## [4] ” ’/ts/index.jhtml’”   ” ’/jszz/index.jhtml’”
```

5 使用 Aprior 算法实现网站的关联分析

5.1 APrior 算法介绍

Aprior 算法是应用最广泛的关联规则算法，其主要思想是找出存在于事务数据集中最大的频繁项集，利用最大频繁项集与预先设定的最小置信度阈值生成强关联规则。Aprior 算法包含两个过程：根据最小支持度阈值找出事务数据库中所有的频繁项集；由频繁项集和最小支持度产生强关联规则，最后根据算法结果输出关联规则。在 Aprior 算法中有几个常用的术语，分别是置信度、支持度和提升

度。支持度指的是 A 和 B 同时发生的概率，置信度是指 A 发生条件下 B 发生的概率，提升度表示 A 发生的条件下，B 发生的概率与 B 总体发生的概率之比，当提升度小于 1 时，说明规则负相关，若提升度大于 1 时，则说明规则正相关。

5.2 构建关联规则模型

5.2.1 数据来源及预处理

由于关联规则分析所需的数据和推荐系统使用的数据内容和数据格式都相同，故我们只需要使用推荐系统中的 “newdata” 数据即可。

5.2.2 数据结构转换

```
library(arules)  # 导入所需库包
# 数据形式转换
dataList <- list()
for (i in unique(newdata$ip)) {
  dataList[[i]] <- newdata[which(newdata$ip == i), 2]
}
# 将数据转换为关联规则所需要的数据类型
TransRep <- as(dataList, "transactions")
# 查看转换后数据的前行数据2
inspect(TransRep[1:2])
```

```
##     items                                    transactionID
## [1] { '/sm/736.jhtml'}                        '183.63.102.193'
## [2] { '/index.jhtml', '/notice/757.jhtml'}  '58.249.112.60'
```

从转换后的数据可以看出，IP 为'183.63.102.193' 的用户只浏览过一个网站，而 IP 为'58.249.112.60' 的用户浏览过两个网站，接下来我们将使用这种形式的数据进行关联规则分析。

5.2.3 热门网站分析

```
click <- data.frame(table(newdata$page_path))
click["percent(%)"] <- click$Freq/sum(click$Freq)*100
click['hot'] <- (click$Freq-min(click$Freq))/(max(click$Freq)-min(click$
    Freq))
head(click[order(click$Freq,decreasing = TRUE),],10)
```

```
##                      Var1 Freq percent(%)    hot
## 839         '/index.jhtml' 1413      3.727 1.0000
## 2598  '/zytj/index.jhtml' 1141      3.009 0.8075
## 2215      '/ts/661.jhtml'  980      2.585 0.6936
## 1867  '/stpj/index.jhtml'  927      2.445 0.6561
## 2524  '/yxzp/index.jhtml'  902      2.379 0.6384
## 1064  '/jszz/index.jhtml'  682      1.799 0.4827
## 1325  '/notice/757.jhtml'  668      1.762 0.4728
## 2182      '/ts/578.jhtml'  578      1.524 0.4091
## 1525      '/qk/729.jhtml'  540      1.424 0.3822
## 2204      '/ts/654.jhtml'  535      1.411 0.3786
```

从分析结果中可以看出，‘/index.jhtml’ 网站的访问用户最多，达到了 1413 名，占比为 3.73%；其次是’/zytj/index.jhtml’，该网站有 1141 个用户访问，占比 3% 。

5.2.4 生成关联规则

```
# 生成关联规则
rules <- apriori(TransRep, parameter = list(support = 0.01, confidence =
    0.5))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support
   minlen
##         0.5    0.1    1 none FALSE            TRUE       5    0.01
      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 61
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[856 item(s), 6130 transaction(s)] done [0.02s].
## sorting and recoding items ... [146 item(s)] done [0.00s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 4 5 6 7 8 done [0.01s].
## writing ... [1838 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
summary(rules)
```

```
## set of 1838 rules
##
## rule length distribution (lhs + rhs):sizes
##   2   3   4   5   6   7   8
##  78 621 580 327 168  56   8
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00    3.00    4.00    4.05    5.00    8.00
##
## summary of quality measures:
##     support         confidence          lift            count
##  Min.   :0.0101   Min.   :0.500   Min.   : 2.17   Min.   : 62.0
##  1st Qu.:0.0114   1st Qu.:0.661   1st Qu.: 3.94   1st Qu.: 70.0
##  Median :0.0131   Median :0.793   Median : 5.26   Median : 80.0
##  Mean   :0.0148   Mean   :0.774   Mean   : 6.56   Mean   : 90.6
##  3rd Qu.:0.0160   3rd Qu.:0.890   3rd Qu.: 6.95   3rd Qu.: 98.0
##  Max.   :0.1106   Max.   :1.000   Max.   :28.87   Max.   :678.0
```

```
##
## mining info:
##       data ntransactions support confidence
##  TransRep          6130    0.01        0.5
```

```
# 查看提升度排名前二的规则
inspect(sort(rules, by = list('lift'))[1:2])
```

```
##     lhs                          rhs                          support
    confidence  lift count
## [1] { '/yxzp/index.jhtml',
##       '/yxzp/index_3.jhtml'} => { '/yxzp/index_2.jhtml'} 0.01468
    0.9184 28.87    90
## [2] { '/yxzp/index.jhtml',
##       '/yxzp/index_2.jhtml'} => { '/yxzp/index_3.jhtml'} 0.01468
    0.5625 25.54    90
```

通过关联规则可以得出，在第一条规则中，用户同时访问 '/yxzp/index.jhtml','/yxzp/index_3.jhtml' 和'/yxzp/index_2.jhtml' 这三个网站的概率为 1.468%，但是在用户浏览完'/yxzp/index.jhtml' 和'/yxzp/index_3.jhtml' 网站之后，浏览'/yxzp/index_2.jhtml' 网站的概率达到了 91.84%，提升度达到了 28.87，说明该规则有效。同理对应第二条规则，用户同时访问规则 2 中的三个网站的概率为 1.468%，但是在浏览完前两个网站之后，浏览第三个网站的概率为 56.25%，提升度为 25.54。所以通过上述的规则信息，我们可以在网站设计的时候将这些存在明显有效规则的网站放在相邻位置，以便于用户更加方便快捷的访问。