2016 湖南省研究生数学建模竞赛参赛承诺书

我们仔细阅读了湖南省研究生数学建模竞赛的竞赛规则.

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的，如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们将受到严肃处理。

我们授权湖南省研究生数学建模竞赛组委会，可将我们的论文以任何形式进行公开展示（包括进行网上公示，在书籍、期刊和其他媒体进行正式或非正式发表等）。

我们参赛选择的题号是（从组委会提供的试题中选择一项填写）：1

我们的参赛报名号为（如果组委会设置报名号的话）：

所属学校（请填写完整的全名）：

参赛队员（打印并签名）：1.

2.

3.

指导教师或指导教师组负责人(打印并签名)：

日期：　　年　　月　　日

评阅编号（由组委会评阅前进行编号）：

2016 湖南省研究生数学建模竞赛

编 号 专 用 页

评阅编号（由组委会评阅前进行编号）：

评阅记录（可供评阅时使用）：

评阅人										
评分										
备注										

湖南省第二届研究生数学建模竞赛

题 目　法律咨询平台日志的数据分析与建模

摘　　要：

在依法治国已经上升为国家战略的背景下，人们的法律意识不断增强，但由于传统的法律行业资源存在分布不均匀，人均资源匮乏的问题，web 法律咨询平台越来越受人重视。本文通过对法律快车网站日志数据的分析，为该网站提供指导性的建议。针对两大方面的问题，我们进行了建模和实验设计。

问题一：第一小问是对日志数据中用户关注领域，访问路径，客户体验等方面进行评价。我们首先对法律快车网站 2015 年三个月的日志数据 log.csv 进行了会话的划分，根据是否通过搜索引擎进入网页将其划分成 entr.csv 和 noentr.csv 两个会话数据集，并通过描述性统计的方法，针对不同类型的属性具体分析，得到如下结论：用户的关注领域较为分散，但在“量罪定刑”、“妇女权益”、“故意伤害”、“劳动合同纠纷”等领域具有较高的关注度；我们将会话长度作为访问路径的度量，发现会话长度服从重尾分布，entr.csv 的会话长度均值为 1.358，noentr.csv 的会话均值为 4.512，说明通过搜索引擎咨询更加高效；客户体验和网站的成功率和效率有正相关关系，需要进一步讨论。第二小问是对法律快车网站的成功率和效率的评价，我们从会话的长度和会话属性中的语义相似度两个方面设计了描述 entr.csv 与 noentr.csv 中会话的成功率，效率的综合指标 σ^~和 δ^~，通过计算得到了 σ^~=0.3787 和 δ^~=0.6790 两个得分,进一步地计算出了不同搜索引擎的 σ^~值，发现百度（0.6201），搜狗（0.6060）和神马（0.5495）的搜索成功率和效率较高，用户在这三个引擎上的客户体验也是最好的。

问题二：第一小问是对法律快车网站推荐系统的设计。我们通过合并同 ip 的会话用户以及会话中的 79 种页面标题类型构造评估矩阵R_{n*m}，依次通过 SVD 低秩近似，K-means 用户聚类，评估预测，排名推荐的过程给出了推荐算法；对于上述推荐算法，我们设计了推荐系统的评估算法，通过二分图对应的评估矩阵R_{n*m}进行随机扰动得到$\tilde{R}_{n*m}$，利用之前的算法进行推荐并计算召回率。第二小问是页面组织结构的优化，我们结合了问题一中的数据分析结论，从 web 链接技术的分析，网络缓存资源的配置，搜索引擎推荐和访问链路这四个方面给出了优化的建议。

总的来说在对法律快车网站日志挖掘的过程中，我们的创新点比较明显：1.对于同一数据集，根据不同的特点分开处理，设计了评估网站搜索成功率和效率的新指标，可以指导网站的优化 2.在推荐系统方面推陈出新，在协同过滤的基础上加入了 SVD 和用户聚类两个处理，取得了不错的效果。

关键词：法律咨询，搜索引擎， 描述性统计， 推荐系统， 页面组织优化

法律咨询平台日志的数据分析与建模

1、背景与相关问题

1.1 背景的分析

随着中国社会经济的快速发展，利益关系也越来越复杂，在依法治国已经成为国家战略的大背景下，人们通过法律手段解决问题的意识不断加强，对法律服务的需求不断增长，但是传统的法律行业资源存在严重的匮乏，以律师人数为例，目前全国执业律师约 27 万，北京万人律师比最高，约为 12.6，多个省份万人律师比不足 1。这些现实情况都限制了法律知识的传播和合理利用。

在这种需求的驱动下，法律咨询网站应运而生，利用互联网技术进行信息传播的高效性和海量性，可以让人们足不出户便知天下事，轻轻松松解决烦心事。但是海量法律资讯背后的冗余性，多样性，复杂性和专业性也给传统的信息检索提出了新的挑战。虽然 web2.0 提倡的是一种交互式的信息服务，但是现在的法律网站的交互性明显不足，很大程度上依然依赖于信息的科学检索。并且绝大多数用户缺乏专业法律知识，不了解以往类似案例，一般不能准确判断问题所适用的法律、法规条文，甚至不能确定法律类型，也很难使用恰当的关键词进行有效地搜索。因此，很多用户难以通过网站导航、关键词搜索等传统方式，高效、快捷地独立找到有针对性的法律知识、相关案例或专业律师，影响用户体验，限制网站发挥作用。

因此，对资讯网站历史日志的分析显得极为重要，一方面可以挖掘影响互联网咨询服务的效率因素，为优化页面组织结构提供参考，另一方面可以根据用户的访问特征设置个性化服务，提升用户体验。

1.2 问题的描述

在上述背景下，我们将对两个方面的问题开展研究：1.通过对法律快车网站提供的 2015 年三个月的日志数据进行挖掘，分析用户关注领域，访问路径，用户体验等方面的特征或规律，进一步地评估针对具体问题续展相关页面或专业律师的成功率和效率，总体来说这是关于历史规律总结和网站服务效率评价的问题 2.针对提供的日志数据和相关补充数据，尝试通过优化网页组织结构，增加新功能，建立新模式等等来进行网站服务效率的提升，总体来说这是在做优化的方案设计，以及个性化服务。我们将针对这两个大问题进行建模，尽可能地使模型符合事实需求，具有普适性。

1.3 问题的分析

两大问题的解决都依赖对 log 数据集（csv 格式）和用户访问网页记录数据集中的潜在信息，所以需要对 log 数据集进行预处理，这也是问题一和问题二的基

础。

总体来看，log 数据集的行代表一次访问记录，列代表不同属性，其中的大多数属性是标称类型的数据，例如地区编号，用户浏览器类型，页面类型等等，都不具有序关系；不同的属性信息存在很强的冗余性，例如“浏览器代理和用户浏览器类型”，“用户 id，客户端 id 和真实 ip”，“页面标题名字和页面标题类型”等等，它们都存在很强的信息冗余，几乎可以视为同一属性；同一 ip 的访问次数存在差异，访问的标准时间存在时序关系，可以通过这种关系进行访问记录的合并构成不同的会话，合并成会话是为了简化搜索意图，会话的长度反映了搜索感兴趣话题过程的复杂度；访问的入口存在缺失，搜索关键字存在缺失，这是因为用户不是通过搜索引擎进入的，所以需要针对这种不能填充的数据单独处理，另外一部分通过搜索引擎进入的存在搜索关键字，反映了用户的意图，所以可以用来和会话最终的页面标题进行相似度匹配以及进一步的评价。

用户访问网页记录（xls 格式）数据集是对 log.csv 数据集的整理，不过缺失了用户的真实 ip 属性，由于日志的用户 id 和客户端 id 无法很好的区分用户的唯一性，所以很难还原用户的搜索轨迹，不过它提供了处理得到的 session count，time on site 和 perv.pag 属性，可以提供额外的信息。

变量名：	**类型**	**描述**	**使用方法**
Log.csv 文件中的变量描述（括号内为在访问记录中的属性名）			
真实 ip	名义变量	用以区分用户的唯一标识	判别用户
地区编号（areacode）	名义变量	区分用户所在地区的编号	转换为类别代码，用 1、2、3 • ……代替
浏览器代理	名义变量	对用户使用的浏览器和系统环境的描述	舍弃
用户浏览器	名义变量	对用户的操作系统的描述	转换为类别代码，适当合并
用户 ID（userID）	名义变量		舍弃
时间戳	连续变量	以毫秒为单位的标准时间	将一个用户短时间内的访问合并成一个会话
标准时间（time）	连续变量	以年月日时分为单位的标准时间	
页面路径	名义变量	访问的页面在网站中的地址	舍弃
年月日	连续变量	访问的年月日	舍弃
访问网页	名义变量	访问的具体网址	抓取网址的关键字，构造为类别代码
页面类型（url_ids）	名义变量	页面类型的内部描述	转换为类别代码，用 1、2、3 • ……代替
主机名	名义变量	访问时通过的主机	舍弃
页面标题（title）	名义变量	页面的标题全称	分词后形成词向量，与搜索关键字对比。
页面标题类型	名义变量	页面标题名称的标号	作为推荐系统的推荐内容
页面标题名称 (title_categories)	名义变量	页面标题的主题分类	舍弃

标题关键字（title_wds）	名义变量	页面标题的关键字	舍弃
入口	名义变量	进入该网站的方式	转换为类别代码，用 1、2、3 • ……代替
入口网页	名义变量	进入该网站的前一网页	舍弃
搜索关键字(organicKeyword)	名义变量	在搜索引擎中的搜索内容	分词后形成词向量，与页面标题对比
来源	名义变量	与入口重复	舍弃
用户访问记录中的独有变量描述			
sessionCount	连续变量	对通过的模块计数	
timeOnSite	连续变量	在网站中的时间	
fullReferrer	名义变量	引用来源	
perv.page	名义变量	上一页	

表 1：变量描述

附件 2.3.4.5 是对附件 1 中的 log.csv 数据集和用户访问网页数据集的补充，尤其是对于数字串编码的语义解释。

对于问题一的求解分析：预处理阶段，我们可以在 log.csv 数据中对同 ip 记录进行会话的划分（对时间相近的访问记录合并构成一次会话），这样可以得到对于分析有益且数量较少的会话记录，进一步将会话长度作为会话的额外特征。通过对整理后的会话记录数据根据是否通过搜索引擎进入 web 法律咨询网站将其划分成两部分数据集 entr.csv 和 noentr.csv 单独分析。进一步地，对不同属性进行描述性统计分析可以看出用户在不同属性上的分布比例，例如用户关注领域，访问路径，用户体验等。我们在评估找到相关页面或律师的成功率时，可以将会话长度，比例以及会话搜索前后语义相似度考虑在内，设计新的指标进行系统评价。

对于问题二的求解分析：可以结合 log.csv 数据和用户访问网页数据集，整体分析网页的组织结构对部分信息进行统计，融合问题一中的数据分析结论，例如用户关注领域，访问路径分布等等给出页面组织结构优化的建议，在个性化推荐方面，利用协同过滤的思想设计推荐系统，在处理过程中需要评估矩阵的稀疏性给出低秩的表示。

2、模型的假设与符号说明

2.1 模型的假设

1.假设大部分用户在访问网站时存在特定意图，想要搜索特定信息
2.对于短期内同一 ip 用户出现大量的访问记录可以视为无明显意图访问或恶意访问，比如说网络爬虫，这些数据可以用来做异常点挖掘，进一步的优化网络结构
3. 真实 ip 属性可以对用户进行有效区分，日志的用户 id 和客户端 id 无法很好的区分用户的唯一性

2.2 符号说明

符号	含义
LOG.CSV	原网站日志
ENTR.CSV	通过搜索引擎进入的会话记录
NOENTR.CSV	直接通过法律快车进入的会话记录
CONV.CSV	总的会话记录
$\boldsymbol{\sigma}^{\sim}$	entr.csv 中的平均搜索成功率和效率综合指标
$\boldsymbol{\delta}^{\sim}$	noentr.csv 中的平均搜索成功率和效率综合指标
S	语义相似度（word2vec 余弦夹角）
$\mathbf{R_{n*m}}$	推荐系统中的评估矩阵，其中 n: user 数目 m: item 数模
$\boldsymbol{C_{n*k}}$	潜在因子评估矩阵
$\mathbf{f_i}$	聚类结果得到的指示向量
$\otimes$	Kronecker 积
$\mathbf{v_i^{\rightarrow}}$	类平均得分
Correct _ Total	推荐还原的链路数目
Set _ Items	推荐评估时随机移除的链路集合
$\mathbf{R_{n*m}^{\sim}}$	推荐系统扰动后的评估矩阵
SVD	矩阵奇异值分解

3、数据预处理

本文主要对日志的原数据 log.csv 进行信息挖掘，log.csv 数据集收集了基于 web 服务的海量网页点击流和用户访问轨迹，时间等信息，反映了用户浏览的习惯和兴趣等等，但是海量数据中实用信息通常较少，只有少部分是对用户有用的。

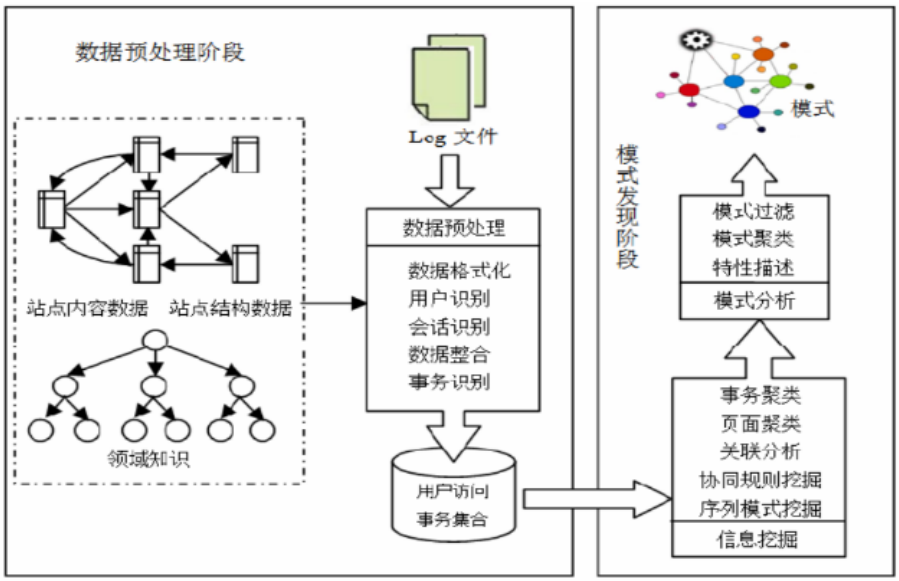

图 1：模式挖掘思路图 [1]

log.csv 数据集总共有 837451 行访问记录，21 列属性信息，其中绝大部分数据是标称类型的，只反映某个属性中的一个标签。

3.1web 用户（user）识别

由于在日志中同一用户可能使用不同的客户端 id 或用户 id 进行相关问题的搜索，如果不将其身份进行唯一标识会造成两大问题：1.同一用户在一个时期内会有相似的搜索模式，不同的身份标识会导致推荐信息重复，而且在进行推荐系统设计时需要用户具有唯一的标识 2.在进行记录的某些属性统计时会将导致具有与该用户相关的统计特征脱离现实。

log.csv 数据集中我们选择真实 ip 作为用户的唯一身份标识，在日志数据中用户 id 和客户端 id 可能无法很好的区分用户的唯一性。

3.2 会话划分算法

会话（session）定义为同一用户在一定时期内一系列的访问网页记录，在 log.csv 数据集中，会话即是同一真实 ip 用户在一定时期内具有时序的网页访问记录。对同一真实 ip 用户的访问进路进行会话分割是考虑到不同时期用户搜索的目标信息可能存在很大的差异，将访问记录划分成若干个单独的会话过程可以更好地挖掘不同时期内用户的意图和搜索结果。我们提出的会话划分算法如

下：

Step1. 对 log. csv 数据根据真实 ip 浏览记录按照访问时间先后进行排序，得到同一真实 ip 用户的访问记录集合$Set_i = \{l_{i1}, l_{i2}, l_{i3}, .., l_{im}\}$

Step2. 依次计算两条访问记录的时间间隔$\{t_{i1}, t_{i2}, .., t_{i(m-1)}\}$

Step3. 设定一个时间阈值α*，设置一个指标*$temp$*，temp=0。从第一个时间间隔*t_{i1}*开始累加，每累加一次 temp+1。*

Step4. 当累加的和超过α*时，将*$\{l_{i1}, l_{i2}, l_{i3}, .., l_{i(temp-1)}\}$*合并为一个会话，判断*$t_{i(temp)}$*与*$\alpha$*的大小关系，若*$t_{i(temp)} \geq \alpha$*，则使得 temp+1。*

Step5. 从$t_{i(temp)}$*开始重新累加，重复 Step4 直至将*$t_{i(m-1)}$*。*

我们在处理数据时设置的阈值为$\alpha = 30\ \text{min}$

3.3 数据清理

Step1. 进行数据的清理,包括空数据的处理,还有就是将一些冗余数据剔除(比如时间戳,日期,时间等等)

Step2. 将数据进行分为使用搜索引擎进入页面 entr. csv 的和直接进入 nonentr. csv 两种数据

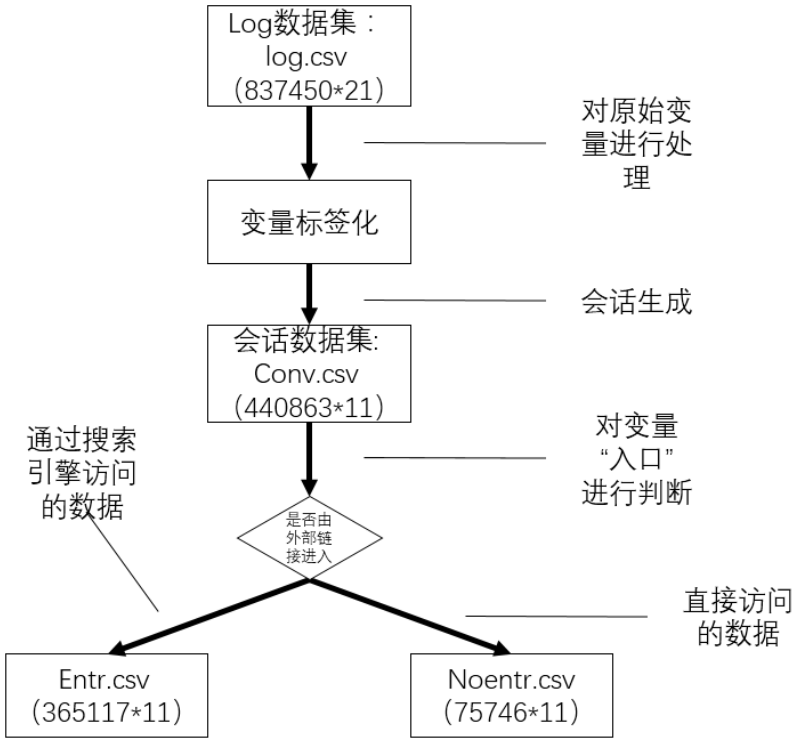

图 2：数据处理及生成的文件（括号内为文件大小行*列）

我们在会话数据集中加入了两个新的属性，即会话长度和会话开始时的搜索关键词。最终，通过上述的处理，我们得到了两部分数据集 entr.csv 和 noentr.csv。

4、问题一的建模与求解

4.1log.csv 数据集的描述性统计

对于 log.csv 数据集分析首先要描述不同的属性的类型和具体含义。

通过对上表的描述，我们清楚了各个变量所代表的意义，以及我们的处理方式，接下来对部分关键变量进行描述性的统计分析。

首先是对我们定义的会话的长度进行的统计分析：

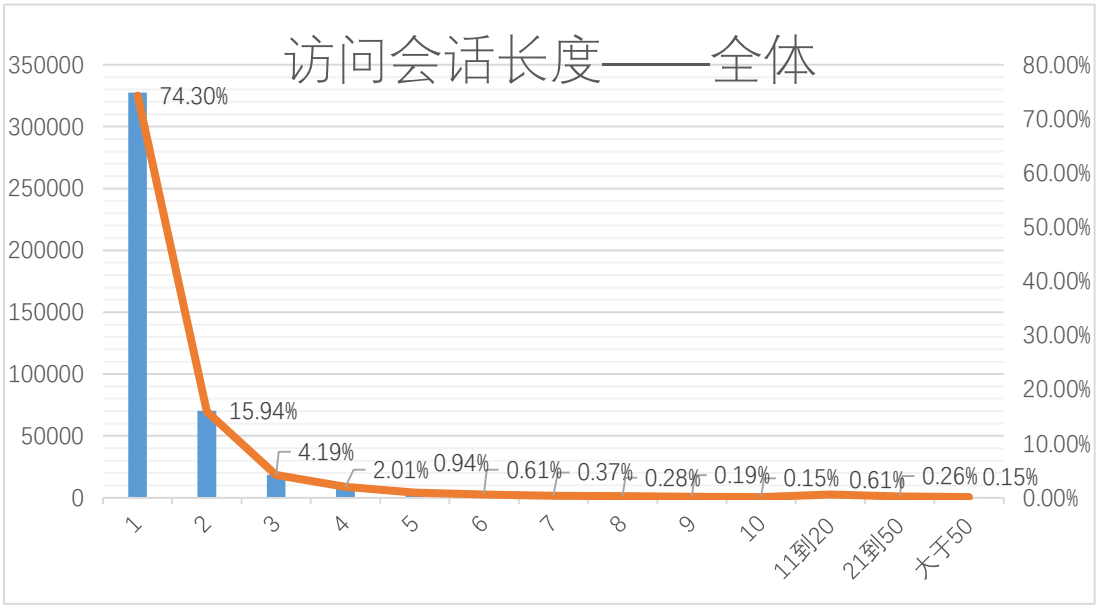

图 3：会话长度统计

上图中蓝色柱状图为实际数值的图像，在主坐标轴标明。而红色的折线则代表占比，在次坐标轴标明。对全部的用户而言，他们的平均会话长度为 1.900，通过上图也不难发现有超过 90%的用户的会话是在两次页面浏览内完成的。会话最长的达到了 2192，即在一个会话中该用户访问了 2192 个页面（可能有重复页面），长度超过 50 的会话总数为 641，占总比的 0.15%。我们不难发现，从总体上而言，用户在该网站的访问次数比较短。

我们知道，通过搜索引擎搜索的网页结果一般都是直接访问的，因此我们对通过网址输入的方式进行浏览和通过外源的搜索引擎访问这两者不同的情况进行对比，即在 entr.csv 和 noentr.csv 中分别统计：

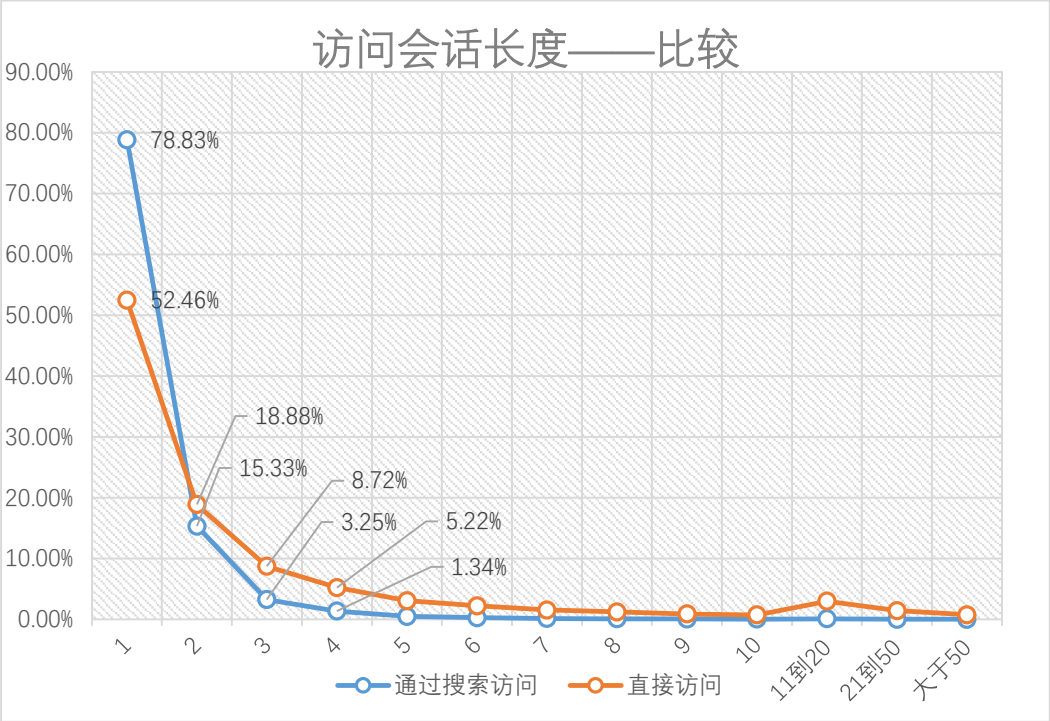

图 4：不同入口的会话长度对比

访问方式	数量（总比）	均值	极大值	超过 50 的占比
通过搜索访问	365117（82.82%）	1.358	273	0.02%
直接访问	75746（17.78%）	4.512	2192	0.74%

表 2：不同入口的会话长度对比

上图中红色代表直接访问各个会话长度的比值，而蓝色代表通过搜索引擎进行访问的会话长度。我们可以明显发现：通过搜索引擎来访问的会话更多，但是其长度普遍意义上要比直接访问的长度要低。而且，通过搜索引擎来访问该网站的会话中，超长会话（超过 50 个页面）的比值远远小于在该网站直接访问的。这直接说明了搜索引擎是用户访问该网站的主要途径，而且搜索的准确度较高。

接下来，我们来考虑下用户的关注领域:

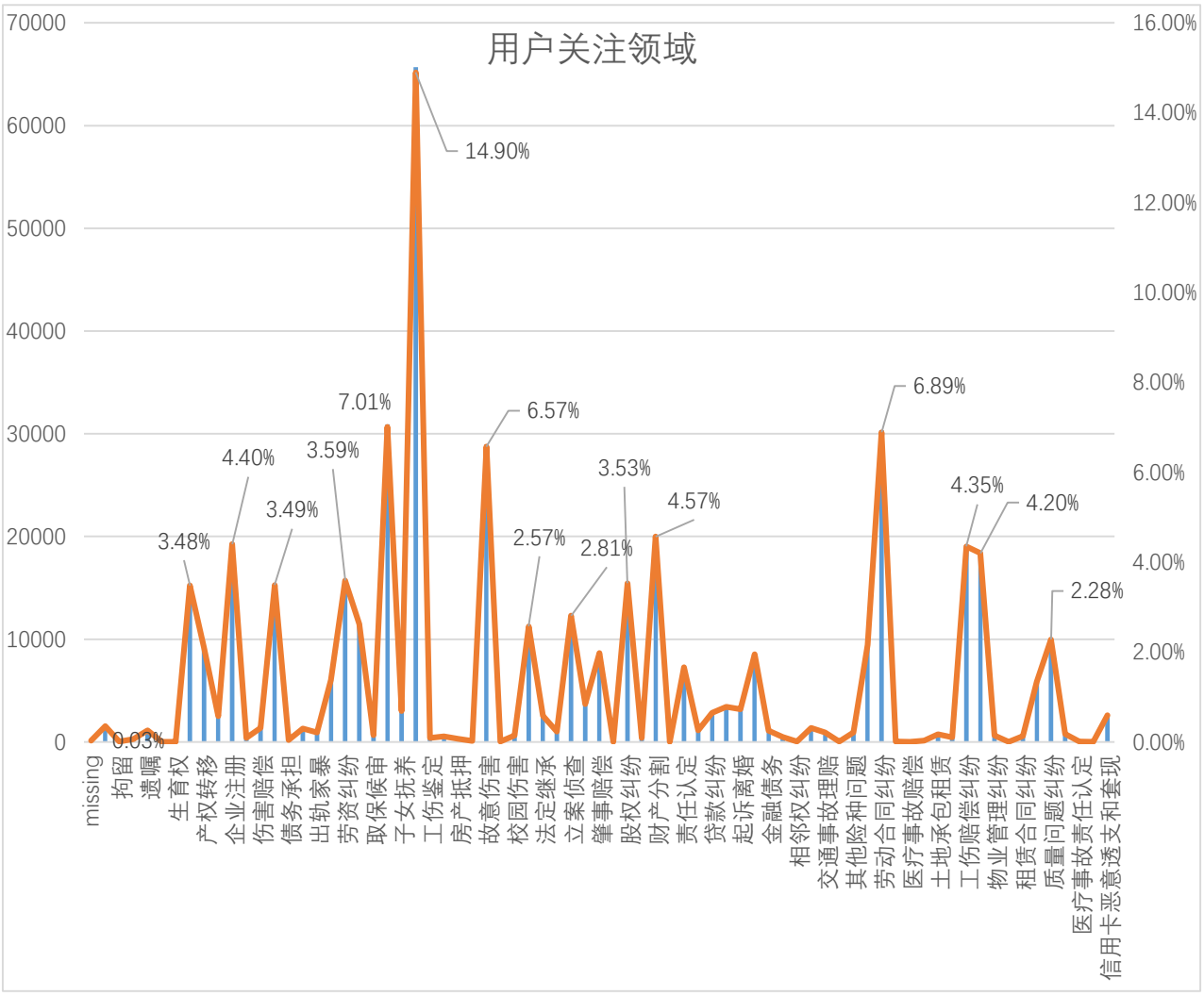

图 5：用户关注领域直方图

综合 entr.csv 和 noentr.csv 的数据，我们发现用户在“量罪定刑”、“妇女权益”、“故意伤害”、“劳动合同纠纷”等领域具有较高的关注度。我们依据各个关键词的出现频率，制作了一个词云图，其中字的大小与其出现的频率成正相关关系。关于领域的分析，在推荐系统的构建和用户分类的过程中还将有更为详细的叙述。

图 6：关键词词云

对用户的浏览器类型，我们进行如下分类：将所有的 WINDOWS 系统分为一类，IOS 系统分为一类，MAC OS 系统分为一类，ANDRIOD 系统分为一类，其余的各种操作系统合为一类，这样我们构建出了下图：

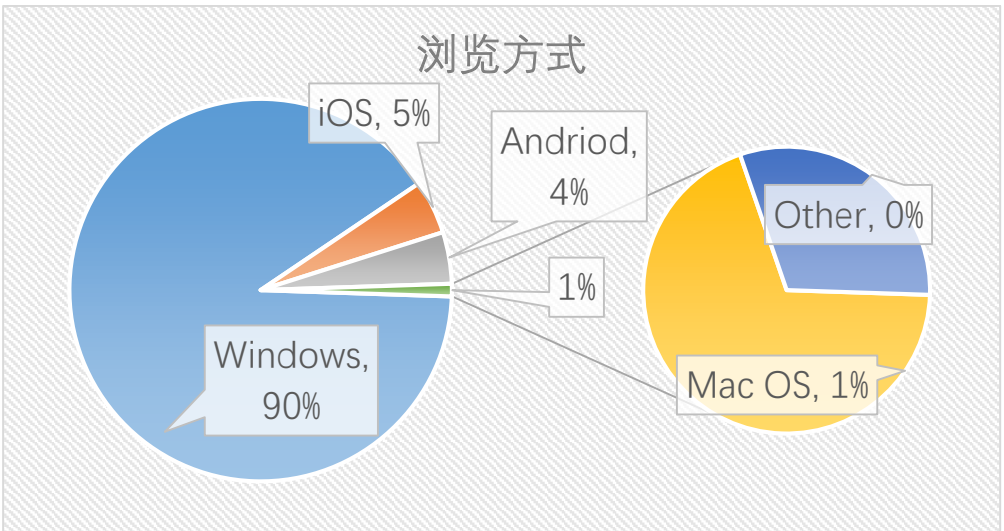

图 7：用户各类操作系统占比

我们不难发现，访问该网站的用户主要是通过 Windows 系统的电脑进行访问的，这与我国的计算机市场的情况相一致。

对通过搜索引擎进行访问的会话，我们将其进入方式也进行了分类：通过百度、百度百科、手机版百度等方式进入的统一分为一类，直接访问的分为一类，通过搜狗、手机版搜狗的分为一类，通过必应搜索（微软）的分为一类，通过神马搜索的分为一类，这样我们构造出下图：

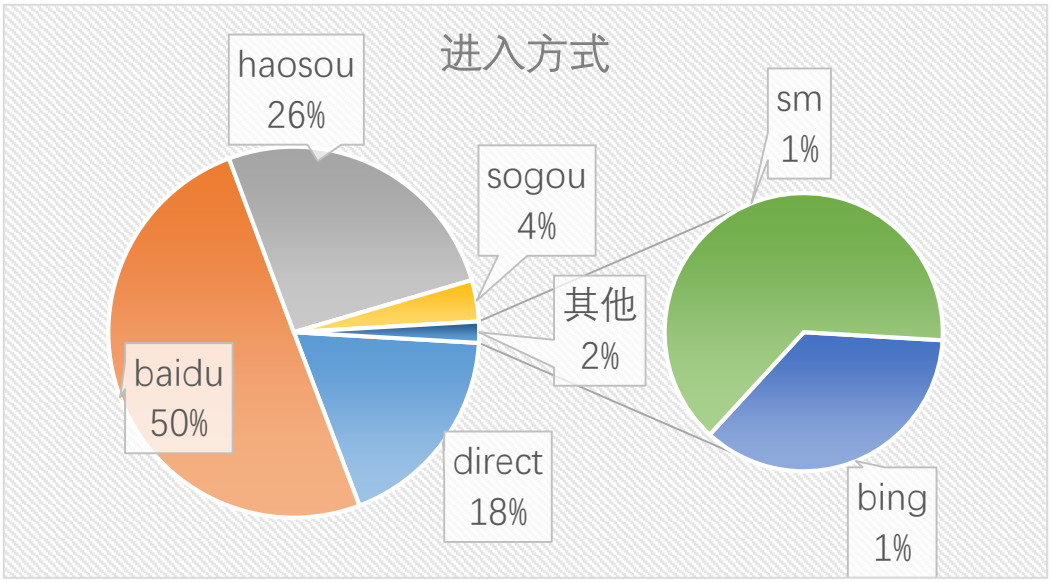

图 8：用户访问网站方式

除此以外，我们对会话的最终访问网址也进行了挖掘，通过分词比较的方式，我们提取出了除去地名以外的以下 11 个关键词：

通过该信息的挖掘，我们可以发现用户浏览该网站的主要目的是为了解决法律疑问或者查询指定专题的法律信息，而真正关注律师、事务所、法规等信息的会话并不多。

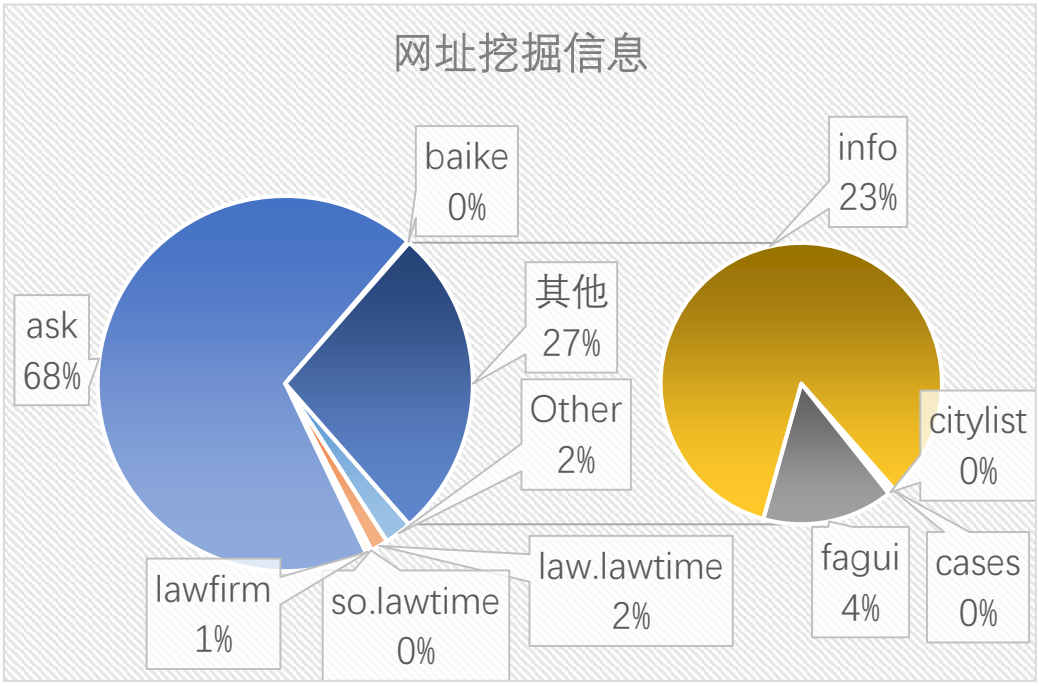

图 9：用户访问网址挖掘

4.2 成功率和效率的评估与用户体验的刻画

在对 log.csv 数据集进行会话合并后分成两个会话数据集 entr.csv 和 noentr.csv，前者是对通过搜索引擎进入 web 法律咨询网站的会话进行汇总，后者是对非通过搜索引擎进入 web 法律咨询网站的会话汇总。

在刻画成功率，效率和用户体验的时候需要进行用户行为模式的假设：1.大部分用户都是带有特殊意图进入法律咨询平台的，有其特定的兴趣所在，通常一个会话关注的内容不会很多，在搜索并浏览完感兴趣的内容后结束此次会话 2.会话长度很长的用户可以认为是异常访问，如网络爬虫，或者该用户是带有各种法律科普的无意图访问，这一类通过总的会话数据集可以看出（见图 3），超过 50 长度的会话比例只有 0.26%,所以这些异常访问数据对我们评价 web 法律咨询网站的效率和计算用户体验没有很大的参考价值，可以直接忽略。

进一步地分析，对于数据集 noentr.csv 来说，由于不是通过搜索引擎进入的，无法获取用户进入网站时键入的搜索关键词，只有会话结束时的网页及关键词，无法反映其意图；对于数据集 entr.csv 来说，通过搜索引擎进入，并键入了搜索关键词，所以可以通过比对会话结束前网页的关键词与会话开始时键入的搜索内容作语义相似度的计算，进而推断用户是否搜索到了想要的内容（由于没有给出在会话结束时网页的逗留时间，缺失了部分参考信息）。

在此给出会话数据集 entr 中语义相似度的计算方法:

利用会话数据集中的两个属性：”搜索关键字“（这个是一个会话中的第一条访问记录中的”搜索关键字”,反映了用户最初想要搜索的内容）和“页面标题”（这个是会话中最后一条访问记录中的“页面”标题）。

Step1. 从数据集 entr 中提取”搜索关键字“和“页面标题”属性提取的代码是 getaccu.py（见附录）

Step2. 利用开源的 NLPIR 汉语分词系统（ICTCLAS2016）[2]*对”搜索关键字“和“页面标题”进行分词处理*

Step3. 使用余弦定理计算分词向量的距离作为语义相似度s，结果保存在文件 cf. csv

我们在利用数据集 entr.csv 评估 web 法律咨询网站效率和成功率时，提出了如下依据：1.用户在某个会话中的会话长度越长，说明其找到自己感兴趣的内容的链路越长，过程越曲折（这里依赖于假设 2），那么网站的效率和成功率就越低 2.用户在某个会话中，会话开始在搜索引擎中键入的搜索内容和用户结束该会话时停留网页的关键词语义相似度越低，说明其搜到的内容并不能匹配他的意图，那么网站的效率和成功率也越低。

基于这两个依据，我们提出了 web 法律咨询网站的成功率或效率评估指标如下：

$$\sigma_i(s_i, l_i): (s_i, l_i) \to R^+$$

其中i是数据集 entr 中的第i次会话，s_i是会话结束前网页的关键词与会话开始时键入的搜索内容进行语义相似度的计算值，l_i是这次会话的长度，$\sigma_i(s,l)$是关于s_i的递增函数，关于l_i的递减函数，所以不妨令

$$\sigma_i(s_i, l_i) = \frac{s_i}{l_i}$$

计算均值$\tilde{\sigma} = \frac{\sum_{i=1..N}\sigma_i(s_i,l_i)}{N}$　　(N 为 entr 中的会话总个数)

我们在利用数据集 noentr 评估 web 法律咨询网站效率时注意到没有会话最初的用户搜索语句，我们无从分析用户的最初意图，所以我们提出针对该数据集的评估指标如下：

$$\delta_j(l_j): l_j \to R^+$$

其中j是数据集 noentr 中的第j次会话，l_j是这次会话的长度，$\delta_j(l_j)$是关于l_j的递减函数，所以不妨令

$$\delta_j = \frac{1}{l_j}$$

计算均值$\tilde{\delta} = \frac{\Sigma_{j=1..M}\delta_j}{M}$　　(M 为 nonentr 中的会话总长度)

综合对搜索引擎进入的会话数据数据集 entr 和非搜索引擎进入的会话数据集 noentr 的两个指标，我们可以对整个法律咨询平台的成功率和效率进行评价，见表 3。

数据表	数据表含义	评分方式	平均评分
Entr.csv	通过搜索引擎访问	$\sigma_i(s_i, l_i) = s_i/l_i$	0.3787
Noentr.csv	直接访问	$\delta_j = 1/l_j$	0.6790

表 3：成功率评分

当$\tilde{\sigma}$和$\tilde{\delta}$都比较大的时候，我们可以认为平台的成功率和效率都比较高，此时用户的体验指数也会随之增加，这两个独立的指标可以用来评估系统调整前后的成功率和效率变化，为优化提供指导。

进一步地，我们给出通过不同搜索引擎进入的评价值σ_i，见表 4 和表 5

平台（搜索引擎）	样本数量	平均评分
百度	220794	0.5814
好搜	115311	0.0026
搜狗	15897	0.4666
必应	2921	0.0044
神马	5217	0.3433
其他	4977	0.0741

表 4：不同搜索引擎评分

去除搜索关键字为 0 的数据后，平均得分为：

平台（搜索引擎）	样本数量	平均评分
总体	225556	0.6130
百度	207027	0.6201
好搜	1651	0.1823
搜狗	12240	0.6060
必应	68	0.1877
神马	3259	0.5495
其他	1311	0.2813

表 5：去除 missing 的数据后，不同搜索引擎评分

至于用户体验，我们可以从评价指标中给出反映，评价指标越高，用户体验越好，如表 4，5 给出了不同的搜索引擎上的评价值，也反映了对应的用户体验。

5、问题二的建模与求解

互联网的出现为信息的传播的存储提供了一个平台，网上的信息量飞速增长，但真正对相关用户实用的信息较少，大部分网络用户在浏览网页时只对一小部分信息感兴趣，，这一点在法律快车的日志信息中就有明显的体现，一次会话中，用户可能连续访问了很多不相关的网站之后才找到想要的信息，并在最终的网站上逗留较久的时间，这也导致了法律资讯平台的用户体验的下降。主要的改进体现在两个方面：1.个性化推荐系统的构建，传统的 web 法律咨询服务的效率明显不足，很难满足带有特定目标的用户需求，这就对挖掘客户感兴趣话题提出了要求，这是对用户行为模式的挖掘利用，如果能根据用户 web 日志中的冲浪痕迹提前预知用户的兴趣，或者为大部分感兴趣的话题提供较多的缓存，就可以提升 web 法律资讯平台的效率和用户体验；2.页面组织结构的优化，通过分析 web 日志的规律，利用统计信息，分析和评价网页质量，指导链接设计，优化信息流传播链路结构，提高检索效率，同时考虑信息流的时效性动态调整页面。（另外一个要优化的就是搜索引擎，从表 2 可以看出 82.82%是通过搜索引擎直接进行访问的）

5.1 个性化推荐系统的算法设计

在做信息推荐的过程中，应注意做到针对性，完整性和智能性，当然推荐也有时效性，不同时期的推荐内容要根据客户的兴趣变化而改变。推荐是指用户（user）与物品（item）产生的关系，可以通过二分图表示，如图 12。对应于本文的研究就是真实 ip 用户与页面标题类型之间的推荐。

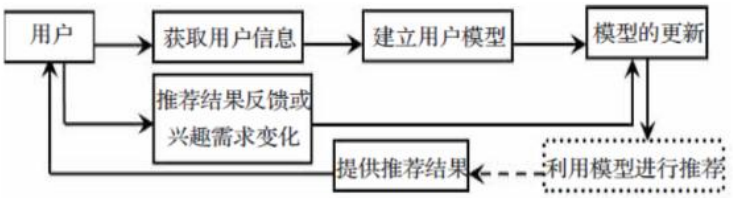

图 10：推荐过程 [3]

目前推荐系统算法主要有两种：一是基于内容的推荐（Content-based Recommendation），二是协同过滤（Collaborative Filtering）。基于内容的推荐是根据用户（user）已经选择了物品（item），在潜在推荐物品中选择与该物品相似的物品作为推荐结果；协同过滤通过分析用户的历史数据，寻找与该用户相似的用户的数据作为未知可能感兴趣的信息作为推荐，它同时考虑到了用户与用户和用户与感兴趣物品之间的相关性，对历史数据整体把握，所以推荐性能较好，这也是我们在解决法律咨询推荐系统设计的思想根源。

考虑到数据本身的特点，我们设计了融合了 SVD，低秩近似，聚类平均三个特色的推荐系统，每种处理都给出了相应的原理解释，具体分成五个步骤,见流程图 11

5.1.1Web 法律咨询评价矩阵R_{n*m}的构造（Rating Matrix）

在协同过滤中，通常需要建立评价矩阵，也就是不同用户对不同物品的评价或者说打分。

我们选择总数为 230149 的真实 ip 作为用户标识，总类型为 79 的页面标题类型作为物品的推荐标识，构造评价矩阵R_{n*m}

其中n代表真实 ip 用户的个数，m代表推荐的物品个数（n=230149,m=79），矩阵中的项r_{ij}代表真实 ip 用户i对页面类型j的访问情况

矩阵写成向量的形式

$$R_{n*m}=\begin{pmatrix} r_1^T \\ r_2^T \\ \dots \\ r_{n-1}^T \\ r_n^T \\ \end{pmatrix}$$

5.1.2 评价矩阵的奇异值分解（Singular Value Decomposition）

对于原始的评价矩阵R_{n*m}存在一些特点：高维度下的稀疏性，一个用户访问的 XX 是有限的，即感兴趣的话题很少，而且用户 ip 很多，这就导致矩阵的维度很高，但是评价矩阵元素大部分为 0，同一用户很难找到与其兴趣相似的用户。产生这种情况的原因分析如下：协同过滤的潜在假设是类似的用户具有类似的物品偏好，类似的物品存在类似的潜在感兴趣用户，也就是说评价矩阵R_{n*m}的行向量之间和列向量之间的相关性较高，评价矩阵具有低秩性（low rank）的特点。[4]基于这种考虑，我们将对R_{n*m}进行 SVD 的处理，在给出低秩表示的同时，降低矩阵表示维度，给出潜在因子的表示。

评估矩阵 SVD 的过程为：

$$R_{n*m}={U_{n*n}^{\sim}}^T*\begin{pmatrix}\Gamma^{\sim} & 0\\ 0 & 0\end{pmatrix}_{n*m}*V_{m*m}^{\sim}$$

其中Γ~是对角元（我们称为奇异值）由大到小排列且非 0 的对角阵，易知$\mathrm{rank}(\Gamma^{\sim})=rank(R_{m*n})$

进行k低秩近似的过程为：

$$R_{n*m}\approx U_{n*k}^T*\Gamma_{k*k}*V_{k*m}=U_{n*k}^T*\Gamma_{k*k}^{\frac{1}{2}}*\Gamma_{k*k}^{\frac{1}{2}}*V_{k*m}$$

其中U_{n*k}^T为${U_{n*n}^{\sim}}^T$取前k列，Γ_{k*k}为Γ~取前k阶方阵，V_{k*m}为$V_{m*m}^{\sim}$取前k行，$k<n$。

其中k的设置是计算Γ~中前几个奇异值之和与总奇异值之和的比值达到预定阈值（贡献率）α时的最少奇异值个数。

可以证明通过这种低秩分解可以得到在秩为k的矩阵中对R_{n*m}近似最好的矩

阵（Frobenius 范数下）。

通过奇异值分解得到新的客户对潜在因子的评估矩阵

$$C_{n*k} = U^T_{n*k} * \Gamma^{\frac{1}{2}}_{\mathrm{k*k}} = \begin{pmatrix} u_1^T \\ u_2^T \\ \cdot\cdot \\ u_n^T \end{pmatrix}$$

5.1.3 对用户的聚类

通过 SVD 可以给出用户对潜在因子的评价矩阵C_{n*k}，这反映了用户的偏好，协同过滤需要寻找偏好类似的用户为进一步推荐作准备，所以需要对用户进行聚类处理。

利用 k-means 算法将n个用户根据其偏好聚成s类，得到指示向量（indicator vector）如下

$$\mathrm{f_i} = (\mathrm{f_{i1}}, f_{i2}, f_{i3}, .., f_{in})$$
$$\mathrm{i} = 1..\mathrm{s}$$

其中$\mathrm{f_{ij}} = \begin{cases} 1, if\ \ j \in Cluster\ \ i \\ 0, if\ \ j \notin Cluster\ \ i \end{cases}$

通过s个指示向量可以反映属于不同类中的用户标识，这为我们下一步的推荐提供了依据。

5.1.4 评估矩阵$\mathrm{R_{n*m}}$的重构

对评估矩阵$\mathrm{R_{n*m}}$重构的过程也就是根据用户和用户偏好信息与用户和物品偏好信息进行推荐的过程，推荐的潜在对象是矩阵中所有的0元单位。

我们分析如下：为了克服不同物品所占比重的干扰，我们逐一计算原$\mathrm{R_{n*m}}$中每列中不同类的物品在该列得分和所占的比重，我们定义为类平均比例得分（ratio score），用它作为对该列该类中原本为 0 的项的推荐得分。

具体分两步如下

Step1. 由于不同类型物品（也就是R_{n*m}中的不同的列向量）所在的度量尺度不同，所以我们需要进行规范化，将每一列中的缺失值（0 元）都通过如下方式进行推荐得分，而且得分一定处于[0,1]*中，这种纵向的归一化是为了横向的比较做准备，即：*

计算向量$b = \left(\frac{1}{(1^T * R_{n*m})_1}, \frac{1}{(1^T * R_{n*m})_2}, \frac{1}{(1^T * R_{n*m})_3}, \cdots, \frac{1}{(1^T * R_{n*m})_m}\right)^T$

计算向量$a_i = f_i^T * R_{n*m}$

$$i = 1..s(\text{用户总共聚成了}s\text{类})$$

计算 Kronecker 积向量$\vec{v_i} = a_i \otimes b$

$$i = 1..s$$

说明：这里的$1^T = (1,1,..,1)^T, (1^T * R_{n*m})_k$*为向量中的第*$k$*项，这里的 Kronecker 积就是向量对应的项作乘积，即：*

$$v_i^{\rightarrow} = \left(\frac{(f_i^T * R_{n*m})_1}{(1^T * R_{n*m})_1}, \frac{(f_i^T * R_{n*m})_2}{(1^T * R_{n*m})_2}, \frac{(f_i^T * R_{n*m})_3}{(1^T * R_{n*m})_3}, \ldots, \frac{(f_i^T * R_{n*m})_m}{(1^T * R_{n*m})_m}\right)^T$$

这就得到了所有物品在第i类中的比例得分向量$v_i^{\rightarrow}$。

Step2. 对于原评估矩阵R_{n*m}中的每一行r_p^T，根据指示向量判断所属的聚类标识，例如$p \in Cluster\ i$,则通过$v_i^{\rightarrow T}$中的元素对r_p^T中的 0 元进行同位置元素的打分补全得到$r^{\sim T}_p$。

可以看出，这些比例得分向量$\{v_i^{\rightarrow} | i = 1 \ldots s\}$反映了属于不同类的用户在不同的物品上的偏好或兴趣，我们可以用比例得分给出用户在得分空白物品上的预期评价估计,但是需要注意的是我们估计 0 元的比例得分和矩阵其他非 0 元的数值意义完全不同，这里只是为了方便下一步进行横向的 0 元推荐先后的比较而进行的列规范化。

5.1.5 排名（Ranking）推荐过程

对R_{n*m}重构的每一行$r^{\sim T}_p$中，新的元素打分值(之前矩阵中的 0 元)进行由大到小的排名，取前J个推荐给用户p即可。

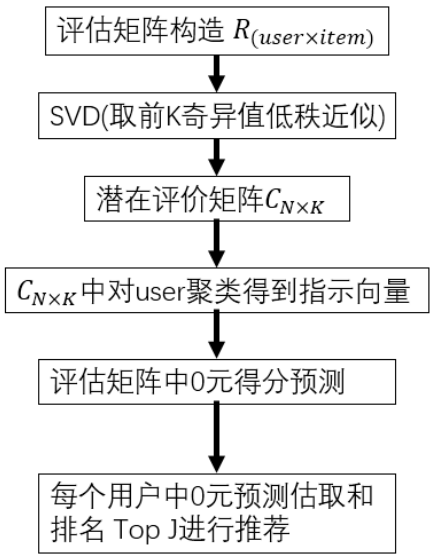

图 11：推荐系统设计

实验的设计与结果分析：

首先我们对 entr.csv 和 noentr.csv 进行合并得到会话数据集，对会话数据集中的“同一真实 ip”的会话记录进行合并作为推荐对象（user），并且以“页面标题类型”属性作为推荐物品（item），其中“真实 ip”用户共 230149 个，“页面标题类型”共 79 种，构造了 230149*79 的评估矩阵，同一真实 ip 用户的所有会话中出

现的页面标题类型进行计数作为评估矩阵的元素。

由于评估矩阵存在大量 0 元，我们进行 SVD 降维低秩矩阵近似的时候，取奇异值贡献率阈值$\alpha = 0.9$，通过 matlab 程序计算得到只需要前 40 个中间矩阵的奇异值即可达到预定效果。

5.2 推荐系统的评价

我们在此进行推荐系统进行评价算法的设计：

Step1. 对于原评估矩阵R_{n*m}*随机选择一定比例的用户集合 Set_Users，随机移除一部分有历史评分的物品（使得评分为 0），记为 Set_Items，通过这种操作可以得到扰动后的评估矩阵*$\tilde{R_{n*m}}$

Step2. 对上述的扰动评估矩阵$\tilde{R_{n*m}}$*使用 5.1 中的推荐算法，对上述 Set_Users 中的用户进行推荐（对于 Set_Users 中的每个用户进行推荐的物品数和之前移除的物品数目相同）*

Step3. 计算 Set_Users 中的每个用户中推荐的物品与之前移除的物品相同的个数，累计之后记为$Correct\_Total$

Step4. 计算召回率（Average Recall Ratio）为

$$Recall = (Correct\ \_\ Total)/|Set\ \_\ Items|$$

其中$|Set\ \_\ Items|$*为集合*$Set\ \_\ Items$*的势，即元素的个数*

上述方法可以多次重复随机操作，计算平均召回率来提升评价的鲁棒性。

图 12 反映了我们在设计评估算法时的思想，如图所示，左图是历史数据存在的推荐，我们随机选择链路进行移除得到右图，其中虚线部分是移除的链路，我们对右图的用户 2，3 通过 5.1 进行推荐，计算还原比例。

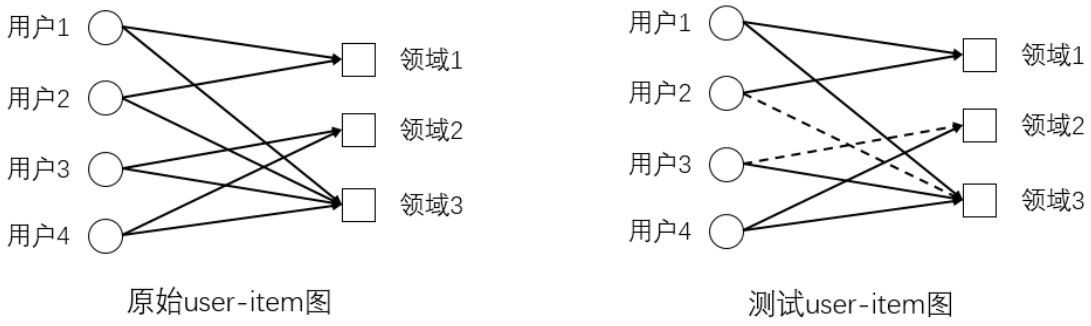

图 12：评价示意

5.3 页面组织结构的优化

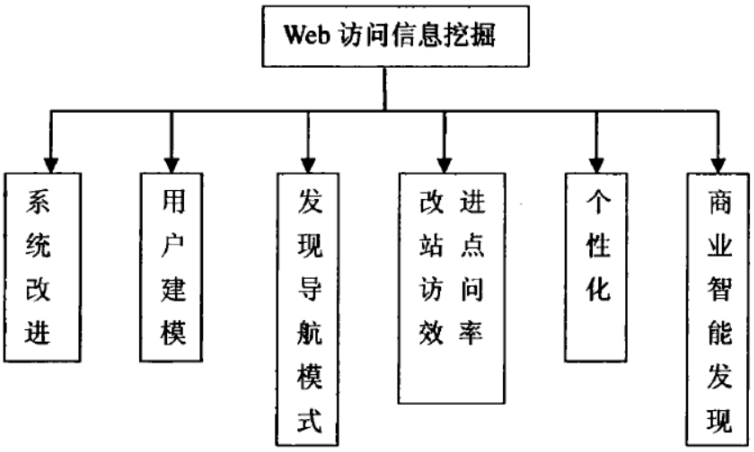

图 13：组织结构优化方式［5］

网页组织结构的优化主要体现在 web 链接技术的分析，网络缓存资源的配置，搜索引擎的推荐和访问链路四个方面，具体来说：

1.web 链路技术的好坏可以直接影响到信息检索的成功率和效率，利用之前对会话数据集的描述性统计结果表 2 可以看出通过网站直接进行会话的比例只有 17.78%，但是平均会话长度却达到了 4.512，远比通过搜索引擎进行会话的平均长度 1.358 要高，这说明真正对特定用户有用的链接并不是那么容易获取，所以我们提出可以通过挖掘网页链路的频繁模式集，寻找较短的会话模式，通过推荐系统给类似的用户个性化展示超链接，减少中间环节。尤其是问题页面和信息页面的分配和推荐需要更加合理。（见图 14）另外一个可以优化的是这种通过挖掘较短的频繁会话模式进行个性化推荐服务是一个时序的过程，网页需要根据不同时期的数据流进行适时调整。

2.网络缓存资源的配置主要体现在对一些用户经常访问的内容进行提前大量的缓存，提高访问效率，缓解服务器压力，主要是通过描述性统计中的图 5，图 6 进行资源按热度配置，这一点也是要适时调整的。

3.搜索引擎推荐的优化，通过之前的描述性统计分析可以看出 82.82%的会话都是通过搜索引擎开始的，透过这一点可以看出搜索引擎在互联网咨询服务中的重要意义，我们通过对 entr.csv 中的不同搜索引擎开展会话的比例（见图 8）和表 4，表 5 中计算得到的$\sigma^{\sim}$和$\delta^{\sim}$值给出分析，对于会话比例较多且$\delta^{\sim}$值较高的搜索引擎如百度（0.6201），搜狗（0.6060）和神马（0.5495）可以投入更多的广告经费。

4.对于一些长会话的个案，我们进行了额外的分析。通过挖掘，我们发现了如下图（图 14）所示的一个特征。有大量用户通过搜索引擎进入到该网站的某一页面（可能是问题，也可能是某个信息）然后这个用户就在同一类页面中不断游走。但是很少出现跳转到其他类型的页面的情况。然而，我们可以发现同一个用户如果有多个长会话，那么他有可能不同次会话在不同的页面类型中游走。所以，我们认为，如果能够在网页推荐上将不同类型的页面按照主题推送的话，就能有效减少用户的会话次数，同时在问题页面对相关信息构建超链接，这样能够更好的帮助用户尽快的找到相关页面并带来更好的体验。

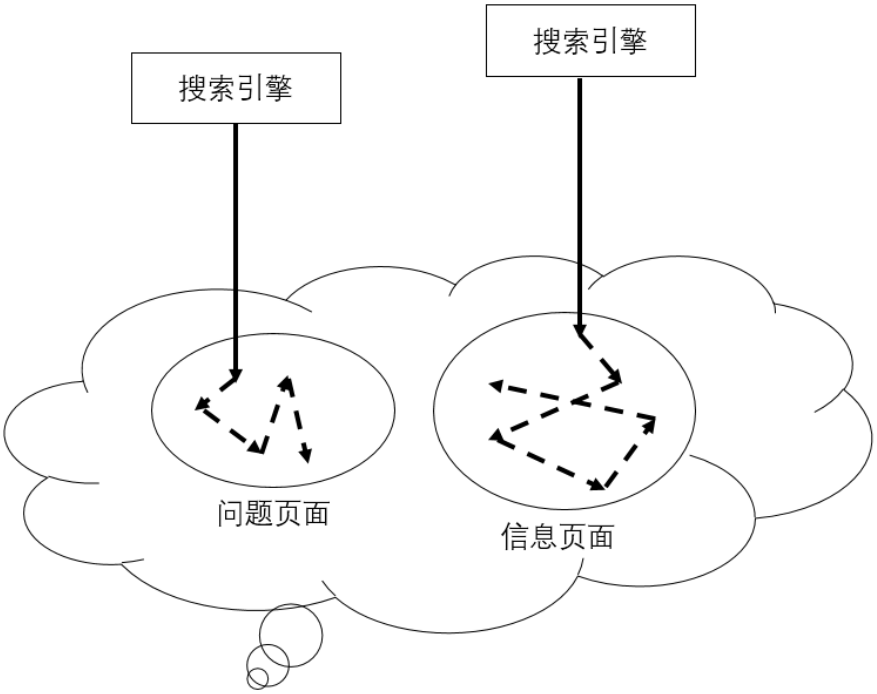

图 14：典型浏览模式示意

6 、模型的评价与推广

6.1 模型的评价

本文主要是对法律快车网站日志数据的分析，模型的好坏很大程度上依赖于数据的预处理 ，通过定义会话（以 30min 作为阈值划分）可以很好地压缩访问记录，忽略对我们分析帮助较少的中间环节，进一步地抽取出会话的初始访问搜索信息和会话长度两个属性，并对信息冗余的属性进行了剔除，整理出了由搜索引擎进入的 entr.csv 和由网站直接进入的 noentr.csv 两个数据集，这些科学的处理方法都为后续的模型求解奠定了基础。

针对会话长度的分布趋势通过直方图给出了描述，关注领域，浏览器类型，访问入口，访问意图等通过饼图给出了直观的感受，对于法律关键词，我们给出了词云的表示方法，通过对这些属性的描述性统计方法，我们可以对数据有初步的认识。在评估用户搜索的成功率和有效率方面，我们提出了两种指标$\tilde{\sigma}$和$\tilde{\delta}$，并通过历史数据 entr.csv 和 noentr.csv 给出了计算，这两个指标可以指导网络的优化，这也直接可以定义为用户的体验指数。在网站的信息推荐方面，我们设计新的推荐系统，以合并同一真实 ip 的信息构造评估矩阵，利用 SVD 降维和 k-means 用户聚类，并启发式地给出潜在推荐对象的预估排名，对于系统推荐的有效性，我们通过对评估矩阵随机扰动再推荐的方法得到推荐对象，计算召回率，这种方法可以很好地模拟真实的推荐情景。

总的来说，我们的文章体现了一定的创新型：1.对于不同属性给出不同的描述性统计 2.提出了评估网站搜索成功率和效率的新指标 3.在推荐系统方面推陈出新，独立设计，并给出了评价算法，总体具有一定的启发性。

6.2 模型的不足和展望

但是由于时间有限，我们的工作也存在一些不足，比如没有做不同属性组合的频繁模式挖掘，web 文本频繁模式的挖掘其实是很有意义的，可以很好地优化信息推送链路，另外一方面没有更深度地给出页面组织结构的分析，这一点也是值得进一步研究的问题。

参考文献

[1] 史玉龙, “Web 访问对象轨迹聚类方法研究,” *工学硕士学位论文*, pp. 7-7, 3 2013.

[2] 张华平, “NLPIR 汉语分词系统,” [联机]. Available: http://ictclas.nlpir.org. [访问日期: 17 4 2016].

[3] 王国霞，刘贺平, “个性化推荐系统综述,” *计算机工程与应用*, 卷 48(7), pp. 67-67, 2012.

[4] C.-Y. L. &. S.-D. Lin, “Matching Users and Items Across Domains to Improve the Recommendation Quality,” 出处 *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining.*, 2014.

[5] 余铁军, “WEB 访问信息挖掘若干关键技术的研究,” *博士学位论文*, pp. 12-12, 4 2006.

附录(只添加了 Python 代码，matlab 打包发送)

getaccu.py(提取使用搜索引擎进入的会话的两个关键字)

```
# coding: utf-8
import matplotlib.pyplot as plt
import re
import scipy;
import csv;
from numpy import *;
import    numpy as np
import pandas as pd

f1 = pd.read_csv('new_entr.csv',encoding='utf-8')
#f2 = pd.read_csv('newdata.csv',encoding='utf-8')


fdata1=f1.values;
f2 = open('dic.csv','r');
lines = [0 for i in range(835476)];
count=0;
for line in f2.readlines():
      t = line.split(',')
      if len(t) >=3:
            resultline=""
            for i in range(2):
                  resultline=resultline+t[i]+','
            lastword=""
            for i in range(2,len(t)):
                  lastword=lastword+t[i].strip("\n");
            resultline=resultline+lastword
            #print resultline
            lines[count]=resultline
            count=count+1
print count
curp=0;
fmiss=open('miss.csv','w')
f=open('cfinput2.csv','w')
for i in range(109950,len(fdata1)-1):
      tag=0
      for j in range(curp,count-1):
            t = lines[j].split(',')
```

```
            if len(t)<3:
                print "error"
            else:
                if str(fdata1[i,2])==str(t[0]) and str(fdata1[i,4])==str(t[1]):
                    print  >>  f,str(fdata1[i,0])+';'+str(unicode(fdata1[i,11]).encode("utf-
8"))+';'+str(t[2])
                    curp=j
                    tag=1
                    break;
        if tag==0:
            print >> fmiss,i



#f1.fillna('missing');
#f2=f2.drop(['locnum','viewagent','view','userID','clientid','timestamp','time','directory'
,'ymd','urltype','hostname','titallabel','titlename','keywork','entrance','entrancepage','sea
rchword','source'],axis=1)
#f2.to_csv('dic.csv',index=False,encoding='utf-8');
#fdata1=f1.values;
#fdata2=f2.values;



#url=fdata2[:,10];
#df = pd.DataFrame(fdata1)
#df.to_csv('test.csv',index=False,encoding='utf-8');

#if str(fdata1[0,2])==str(fdata2[0,0]) and str(fdata1[0,4])==str(fdata2[0,10]):




#f=open('new_entr.csv','w')
#for line in file_object.readlines():
#    word = line.split(',')
#    resultline=""
#    for i in range(11):
#        resultline=resultline+word[i]+',';
    #print resultline
#    lastword=""
#    for i in range(11,len(word)):
#        lastword=lastword+word[i].strip("\n");
#    resultline= resultline+lastword;
```

```
#     print >>f,resultline

#input_df = pd.read_csv('entr.csv',encoding='utf-8');
#accurancy_df                                                                    =
input_df.drop(['Viewer','convlength','desturl','LocNum','UrlType','TitleLbel','Entrance',
'keyword'],axis=1)
#accurancy_df.to_csv('accurancy.csv',index=False,encoding='utf-8');
```

数据抽取.py

```
# coding: utf-8
import matplotlib.pyplot as plt
import re
import scipy;
import csv;
from numpy import *;
import    numpy as np
import pandas as pd
import math
def getconversation(df):
      tmp=df.values;
      usercount=0;
      usetime= [0 for i in range(len(tmp))];
      userID = [0 for i in range(len(tmp))];
      userIP = tmp[:,0];
      breakpoint = [0 for i in range(len(tmp))];
      allurl = [0 for i in range(len(tmp))];
      TitleName = [0 for i in range(len(tmp))];
      entr = [0 for i in range(len(tmp))];
      keyword = [0 for i in range(len(tmp))];
      viewer = [0 for i in range(len(tmp))];
      skeyword = [0 for i in range(len(tmp))];
      key=tmp[:,13];
      skey=tmp[:,16];
      entrance=tmp[:,14];
      tname=tmp[:,12];
      url=tmp[:,7];
      vi=tmp[:,2];
      breakpoint[0]=1;
      for i in range(1,len(tmp)):
            if(tmp[i,0]==tmp[i-1,0]):
                  usetime[i]=(tmp[i,3]-tmp[i-1,3])/60000
                  userID[i]=usercount
```

```
            if usetime[i]>30:
                breakpoint[i]=1
        else:
            breakpoint[i]=1
            usetime[i]=0
            usercount=usercount+1
            userID[i]=usercount

for i in range(len(url)):
        s=unicode(url[i])
        allurl[i]=s.encode('gb2312')
for i in range(len(tname)):
        s=unicode(tname[i])
        TitleName[i]=s.encode('utf-8')
for i in range(len(entrance)):
        s=unicode(entrance[i])
        entr[i]=s.encode('gb2312')
for i in range(len(key)):
        s=unicode(key[i])
        keyword[i]=s.encode('utf-8')
for i in range(len(skey)):
        s=unicode(skey[i])
        skeyword[i]=s.encode('utf-8')
for i in range(len(vi)):
        s=unicode(vi[i])
        viewer[i]=s.encode('gb2312')



user=[0 for i in range(len(tmp))];
convlength=[0 for i in range(len(tmp))];
desturl=[0 for i in range(len(tmp))];
TN=[0 for i in range(len(tmp))];
LocNum = [0 for i in range(len(tmp))];
TitleLable = [0 for i in range(len(tmp))];
Entrance = [0 for i in range(len(tmp))];
KeyWord = [0 for i in range(len(tmp))];
SeKeyWord = [0 for i in range(len(tmp))];
Viewer = [0 for i in range(len(tmp))];
UrlType = [0 for i in range(len(tmp))];
ptr=0;
convcount=0;
while (ptr<len(usetime)):
```

```
            convlen=1
            curptr=ptr+1
            if(curptr<len(usetime)):
                while (breakpoint[curptr]==0):
                    convlen=convlen+1
                    curptr=curptr+1
            user[convcount]=tmp[ptr+convlen-1,0]
            LocNum[convcount]=tmp[ptr+convlen-1,1]
            TitleLable[convcount]=tmp[ptr+convlen-1,11]
            desturl[convcount]=allurl[ptr+convlen-1]
            TN[convcount] = TitleName[ptr+convlen-1]
            KeyWord[convcount]=keyword[ptr+convlen-1]
            SeKeyWord[convcount]=skeyword[ptr]
            Viewer[convcount]=viewer[ptr]
            Entrance[convcount] = entr[ptr+convlen-1]
            UrlType[convcount]=tmp[ptr+convlen-1,8]
            convlength[convcount]=convlen
            convcount=convcount+1
            ptr=ptr+convlen

        pathlen=[0 for i in range(max(convlength)+1)]
        for i in range(convcount):
            pathlen[convlength[i]]=pathlen[convlength[i]]+1;
        return
convcount,user,convlength,desturl,LocNum,UrlType,TitleLable,TN,Entrance,KeyWor
d,SeKeyWord,Viewer,pathlen




input_df = pd.read_csv('log.csv',encoding='gb18030')

#t=input_df.sort(columns=u'真实 ip');
input_df.columns=['ip','locnum','viewagent','view','userID','clientid','timestamp','time','
directory','ymd','url','urltype','hostname','pagetitle','titallabel','titlename','keywork','entr
ance','entrancepage','searchword','source'];

#对数据进行排序
sort_df=input_df.sort(['ip','timestamp']);
#丢掉一部分数据
sort_df = sort_df.drop(['viewagent','userID','clientid'],axis=1);
#将搜索引擎为空的
temp=sort_df.fillna('missing');
```

```
print temp.columns;


entr_df=temp[temp.entrance=='missing']
#print entr_df
convcount,user,convlength,desturl,LocNum,UrlType,TitleLable,TN,Entrance,KeyWor
d,SeKeyWord,Viewer,pathlen=getconversation(entr_df)
f=open('noentr.csv','w')
print                                                                    >>
f,'convNumber,Viewer,userIP,convlength,desturl,LocNum,UrlType,TitleLable,TitleNa
me,Entrance,keyword,searchword'
for i in range(convcount):
    print                                                                >>
f,str(i)+','+str(Viewer[i])+','+str(user[i])+','+str(convlength[i])+','+str(desturl[i])+','+str
(LocNum[i])+','+str(UrlType[i])+','+str(TitleLable[i])+','+str(TN[i])+','+str(Entrance[i]
)+','+str(KeyWord[i])+','+str(SeKeyWord[i]);
f.close()

f2=open('noentrcifa.txt','w')
for i in range(convcount):
    print >> f2,str(i)+','+str(user[i])+','+str(KeyWord[i])+','+str(SeKeyWord[i]);
f2.close()


#print convcount

#f=open('conversation.csv','w')
#print                                                                   >>
f,'convNumber,Viewer,userIP,convlength,desturl,LocNum,UrlType,TitleLable,TitleNa
me,Entrance,keyword,searchword'
#for i in range(convcount):
#                                                          print         >>
f,str(i)+','+str(Viewer[i])+','+str(user[i])+','+str(convlength[i])+','+str(desturl[i])+','+str
(LocNum[i])+','+str(UrlType[i])+','+str(TitleLable[i])+','+str(TN[i])+','+str(Entrance[i]
)+','+str(KeyWord[i])+','+str(SeKeyWord[i]);
#f.close()

#f=open('cifainput.txt','w')
#for i in range(convcount):
#    print >> f,str(i)+','+str(user[i])+','+str(KeyWord[i])+','+str(SeKeyWord[i]);
#f.close()

#f=open('newdata1.csv','w')
#print                                                                   >>
```

```
f,'userIP,convlength,desturl,LocNum,UrlType,TitleLable,TitleName,Entrance,keywor
d,searchword'
#for i in range(convcount):
#                                                                   print >>
f,str(user[i])+','+str(convlength[i])+','+str(desturl[i])+','+str(LocNum[i])+','+str(UrlTyp
e[i])+','+str(TitleLable[i])+','+str(TN[i])+','+str(Entrance[i])+','+str(KeyWord[i])+','+st
r(SeKeyWord[i]);
#f.close()

#noentr_df=temp[temp.entrance=='missing']
#print entr_df
#convcount,user,convlength,desturl,LocNum,UrlType,TitleLable,TN,Entrance,KeyW
ord,SeKeyWord,pathlen=getconversation(noentr_df)

#f=open('newdata2.csv','w')
#print                                                                   >>
f,'userIP,convlength,desturl,LocNum,UrlType,TitleLable,TitleName,Entrance,keywor
d,searchword'
#for i in range(convcount):
#                                                                   print >>
f,str(user[i])+','+str(convlength[i])+','+str(desturl[i])+','+str(LocNum[i])+','+str(UrlTyp
e[i])+','+str(TitleLable[i])+','+str(TN[i])+','+str(Entrance[i])+','+str(KeyWord[i])+','+st
r(SeKeyWord[i]);
#f.close()



#x=[i for i in range(max(convlength)+1)];
#plt.bar(x[1:20],pathlen[1:20]);
#plt.show();

#noentr_df=temp[temp.entrance=='missing']
#print len(noentr_df)
#convcount2,user2,convlength2,desturl2,pathlen2=getconversation(noentr_df)
#print convcount2

#x=[i for i in range(max(convlength2)+1)];
#plt.bar(x[1:20],pathlen2[1:20]);
#plt.show();
#f=open('noentrout.txt','w')
#for i in range(convcount2):
#       print >>f,str(user2[i])+','+str(convlength2[i])+','+str(desturl2[i]);
#f.close()
```