电子商务网站用户分群

1 项目背景

随着计算机技术与通信技术的日益成熟和广泛应用,互联网自 90 年代初开始得到迅猛 的发展,至今虽只有十几年时间,但已发展成为信息时代的人类生活中不可或缺的部分,成 为全球范围内信息传播的主要渠道,与此同时,连入互联网的用户越来越多。目前,互联网 的用户规模已不容小视,互联网市场潜力巨大,各大网站运营商都在采取积极的措施,分析 用户的行为特征,对不同客户群提供差异化的服务,以达到精准营销的目的。比如,有些网 站根据用户的注册资料,性别、年龄、区域、职业等信息对用户进行分群,但这种分群方式 是"粗犷"的,未能考虑到用户的行为特征和兴趣偏好,分群结果难以为精准营销提供决策 的支持。

网站运营商想要在浩如烟海的互联网用户中找到目标客户存在很大困难,具体表现为: 运营商不知道目标用户浏览网站时有什么样的行为特征和习惯?运营商急需快速有效的方 法解决用户分群问题。在这一背景下,本案例采用数据挖掘的手段分析用户过去的浏览行为, 在此基础上建立用户自动分群模型。

本案例的研究对象是广州泰迪智能科技有限公司旗下的泰迪杯竞赛网站,泰迪智能科技 是一家专门从事大数据挖掘研发、咨询和培训服务的高科技企业。泰迪杯竞赛网致力于为用 户提供丰富的泰迪杯竞赛信息、数据挖掘培训咨询,并提供往届优秀作品作为参考,以及面 向高校的丰富教学资源,如:案例教程、教学视频、教学书籍、建模工具等。访问用户的增 加使得网站越发难以掌握用户的需求。为更好满足用户的需求,本案例依据用户的历史浏览 记录,研究用户的兴趣偏好,分析需求并发现用户的兴趣点,从而将用户分成不同群体。公 司后续可以针对不同群体提供差异化的服务,提高用户的使用体验。

2 项目目标

(1) 依据用户的历史浏览记录,分析用户的行为特征和兴趣偏好。

(2) 根据用户的行为特征和兴趣偏好将用户划分成不同的群体,分析各群体的属性特征。

3 项目步骤

3.1 工程前期准备

3.1.1 导入数据

(1) 介绍数据

用户访问网站时,系统会自动记录用户访问网站的日志。该网站被访问的数据记录(部 分字段),如表 3-1 所示。

id	page_path	Userid	seeeionid	ір	data_time
1	/zytj/index.j html	NA	DE80E709835F8AB1A381 96185B05FDBC	218.28.23.137	2016/7/14 18:33
2	/zytj/index.j html	NA	ED095CA37DB28D14041 24B4988CAFB9F	218.28.23.137	2016/7/14 18:33
3	/xtxm/index. jhtml	8180	773F9B491EF1027B76698 C489DEB9DB9	188.165.225.224	2016/7/14 18:34
4	/notice/614.j html	NA	E32144406C1DEAB298F E4677846A449D	180.153.214.152	2016/7/14 18:35
5	/stpj/626.jht ml	8181	FBD4EB0F3E6390A49399 7B22B0DE51AD	180.153.206.20	2016/7/14 18:35
6	/thirdtipdm/i ndex.jhtml	NA	0430EF0B7E5CD8A3831 E78290DD2CED3	111.206.36.19	2016/7/14 18:35

表 3-1 用户访问记录表

表 3-1 记录了访问序号、内容 id、访问内容、用户 id、ip 等多项属性的记录,并针对 其中各个属性进行了说明,如表 3-2 所示。

表 3-2 访问记录属性表

属性名称	属性说明	属性名称	属性说明
id	访问序号	browser_type	浏览器类型

content_id	内容 id	browser_version	浏览器版本
page_path	网址	platform_type	平台类型
username	用户名称	platform_series	平台系列
userid	用户 id	platform_version	平台版本
sessionid	一次浏览标识	data_time	访问时间
ip	ip 地址	mobile_type	手机类型
country	国家	agent	代理商
area	区域	uniqueVisitorId	唯一浏览 ID

(2) 上传数据到 Python 数据挖掘建模平台

在新增数据源上,选择本地上传数据,如图1所示。

F	rthon数据挖掘建模平台 ♀ ● ♀	Ø
▲ 首页	我的数据要 共享数据源	
数据源	+ 新環数環境 ▲ 透路状态 ▼ 透路術館前间 0 2	寮
「上程	資源来源于文件 创建人 数据来源 同步状态 创建时间 操作	
● 个人组件	2 数据未源于数据库	
₩		
留	智无数据	

图 1 本地上传数据源

在本地路径上选择文件,填写在平台新建的目标表名,如图 2 所示。

	新建数据》	泉		
1 文件属性	2 预放数据	8		3 字段设置
上传文件	选择文件 Jser_dat.csv 等待上传 删除			
* 新建目标表名	user_dat			
列分隔符	逗号 (,) 🚽	文件编码	UTF-8 👻	
存储有效期 (天)	180 – +	预览设置	分页显示	
				重置 下一歩

图 2 本地选择文件上传

根据文件的数据,可以修改文件的字段名和类型,如图 3 所示。

				新	建数据	調					
1 文件属性					2 预览数	据					3 字段设置
注意:【字段名】只能 原字段	能是以字母开头,由小写英文 字段名	字母、数字、下划线组成 类型		ŧ	度			精度		备注	
id	id	字符	-	255	-	+	0		+		
content_id	content_id	字符	-	255	-	+	0		+		
page_path	page_path	字符	-	255	-	+	0		+		
usemame	username	字符	•	255	-	+	0		+		
	userid	字符	•	255	-	+	0		+		
	sessionid	字符	-	255	-	+	0		+		
	sessionid	字符	•	255		+	0		+		上一步

图 3 字段设置

上传成功,可以在平台的数据源上查看数据,单击数据源操作的查看按钮如图 4 所示, 数据预览如图 5 所示。

我的数据源 共享数据源						
+ 新増数据源 ▼				请输入表名	选择状态 🔻	请选择创建时间
表名	创建人	数据来源	同步状态	创建时间	操作	
user_dat	xinyou	结构化文件	同步完成	2019-05-27 13:59:09	۵	<

图 4 单击预览数据按钮

			预步	危数据 (分页加载)				×
id	content_id	page_path	username	userid	sessionid	ip	country	area
47918		1			D572D21423916 10AED74F0BC75 7CDD58	60.191.123.80	浙江省杭州市	电信
47919		1			058B9C5E9EAB3 02601C4AD4F99 D5AC64	64.89.234.161	美国	ग्रद्धम
		共 38334 务	€ 100 条/页 ▼	(12345	6 … 384 > 前往	1页		

图 5 数据预览

3.1.2 新建空白工程

右击我的工程,新建一个空白的工程,如图 6 所示。

	Python数据挖掘建构	律平台	4 2 6 0 2
▼ 倉页	工程 〇	📩 Q Q 120% % 🖺 O	工程信息
●●● 数据源	 ▶ #104A-T-58 ● 新建工程 ▲ 导入工程 		未选择工程
工程	之 导出工程 ■ 添加文件夹		
₽ 个人组件	组件		
₩	输入内容进行过渡 ▶ 系统组件		
留	▶ 个人组件		
	模型		

图 6 新建工程

填写工程的信息,包括工程名称和工程描述,如图 7 所示。

	创建工程	×
* 工程名称	竞赛网站用户分群	
工程描述	针对泰迪智能有限公司的泰迪杯竞赛网站,为更好地满足用户需求,依据用户的历史浏览记录,研究用户的兴趣偏好,分析用户的需求和行为,发现用户的兴趣点,从而将用户分成不同的群体。	//
工程位置	▼我的工程	
	重置 确定	

图 7 填写工程信息

3.2 数据预处理

读取 user_dat 数据,步骤如图 8 所示。

- (1)选择工程。
- (2) 选择输入源组件。
- (3) 拖入输入源组件。
- (4) 填写数据表名。

(5) 单击更新按钮,更新出数据。

工程 〇	🛓 Q Q 🚺 🗞 🖹 🖸	◇ 字段属性	
▼我的工程 ▲ 竞赛网站用户	₩ 46人源	数据表 user_dat 字段信息	0
组件		字段	类型
输入内容进行过滤		id	字符
▼ 系统组件 ▼ 输入/输出		content_i d	字符
■ 输入源 ■ 输出源		page_pat h	字符
▶ 预处理		username	字符
▶ 统计分析			
▶ 聚类			
▶ 时序模型			
▶ 关联规则			
個型		> 组件描述	

图 8 输入源组件

3.2.1 属性规约

接下来进行属性规约。步骤如图 9 所示。

(1) 找到预处理→数据筛选组件。

(2) 拖入数据筛选组件,并将数据源和数据筛选组件连接。

(3) 单击更新按钮,勾选 page_path、username、userid、sessionid、ip、date_time、 uniquevisitorid 字段作为输出字段。

(4) 对数据筛选组件右键,选择运行该节点。

	📩 Q Q 120% % 🖺 🖸	◇ 字段属性
我的工程		● 特征
竟赛网站用户		
	11日本 11日本 11日本 11日本 11日本 11日本 11日本 11日本	300 tarrier 2011 - 1 dan to 40
		AND TRUE AS THIT
	展 属性规约	字段
。 件		ld
W tārz		content_id
■ 340万 ■ 数据筛选		Page_path
■ 分组聚合		Username
■ 修改列名		
X Python脚本		
X 缺失值处理		
■ 数学类函数		
X 特征构造		
國 数据离散化		
■ 标准化数据还原		
X 数据标准化		
國 数据编码化		> 参数设置
		> 细性描述

图 9 属性规约组件

(5)运行完成后,对全表统计组件右键,选择查看数据。如图 10 所示。

age_path	username	userid	sessionid	ip	date_time	uniquevisitorid
			D572D2142391610AED7 4F0BC757CDD58	60.191.123.80	2016-8-12 0:00	
			05BB9C5E9EAB302601C 4AD4F99D5AC64	64.89.234.161	2016-8-12 0:08	
mgj/575.jhtml			EB8FBC1627FD5B564B A2E06722DBE707	66.249.65.162	2016-8-12 0:09	
			2B9A9334275D6574C6C 9E0FADEC4389B	60.191.123.80	2016-8-12 0:09	
			7CE918EFE6BE41E05F7 9BE46BC80CC99	207.46.13.157	2016-8-12 0:10	
sjfxs/705.jhtml			0D71BB01AA9A61C07B BCAE3DAD3DFFEE	42.156.254.4	2016-8-12 0:15	
			578D4974B43E044279F			

图 10 属性规约结果

(6)运行完成后,对数据筛选组件右键,重命名为属性规约。

3.2.2 数据变换

接下来进行数据变换,步骤如图 11 所示。

- (1) 找到预处理→Python 脚本组件。
- (2) 拖入 Python 脚本组件,并将属性规约和 Python 脚本组件连接。
- (3) 选择字段属性,在脚本处填入数据变换代码,如表 3-3 所示。
- (4) 对 Python 脚本组件右键,选择运行该节点。

工程 0	📩 Q Q 120% % 🖹 O	◇ 字段属性
▼我的工程		输入
▲ 克赛网络用户		input1 from 10007281_1_1
	mart Hiller	input2 from
		input3 from
组件	数据变换	input4 from
X 排序		
■ 数据筛选		脚本
算分组聚合 算修改列名		<pre>1 data_in - db_utils.query(conn, 'select * from '+ inputs['input1']) 2 import warnings 3 warnings.filterwarnings("ignore")</pre>
翼 Python脚本		4 import pandas as pd 5 5 年 3 1 年回一介enceinged田区回的i_ 转化时回一个
X 缺失值处理		7 # 提取運性 in sessionid并将它们唯一化
與 数字关因数 製 结衍构造		<pre>8 ip_sessionid = data_in[["ip","sessionid"]] .drop_duplicates()</pre>
■ 故据高散化		9 10 #按sessionid统计对应的ip个数
■ 标准化数据还原		<pre>11 sessionid_count = ip_sessionid['sessionid'] .value_counts()</pre>
翼数据标准化		U
■ 数据编码化		
模型		> 组件描述

图 11 数据变换组件

表 3-3 数据变换代码

data_in = db_utils.query(conn, 'select * from ' + inputs['input1'])
import warnings
warnings.filterwarnings("ignore")
import pandas as pd

```
#2.1 将同一个 sessionid 中不同的 ip 转化成同一个 ip
# 提取属性 ip、sessionid 并将它们唯一化
ip_sessionid = data_in[["ip","sessionid"]].drop_duplicates()
# 按 sessionid 统计对应的 ip 个数
sessionid_count = ip_sessionid['sessionid'].value_counts()
# 提取计数大于 1 对应的 sessionid 号
rept_sessionid = sessionid_count[sessionid_count > 1].index
# 计算满足条件的 sessionid 个数
len = rept sessionid.size
# 同一个 sessionid 不同的 ip 用 sessionid 对应的第一个 ip 替换
for i in range(len-1):
    index = data_in[data_in['sessionid'] == rept_sessionid[i]].index
    data_in.iloc[index,]['ip'] = data_in.iloc[index[0]]['ip']
del ip_sessionid,sessionid_count,rept_sessionid,len,i
#2.2 将一次点击中有不同的 userid 换成同一个 userid
import numpy as np
# 提取属性 userid、sessionid 并将它们唯一化
userid_sessionid = data_in[["userid","sessionid"]].drop_duplicates()
# 按 sessionid 统计对应的 ip 个数
sessionid_count_1 = userid_sessionid['sessionid'].value_counts()
# 提取计数大于1 对应的 sessionid 号
rept_sessionid_1 = sessionid_count_1[sessionid_count_1 > 1].index
# 计算满足条件的 sessionid 个数
len_1 = rept_sessionid_1.size
# 将同一个 sessionid 且 userid 为 NA 的 userid 用该 sessionid 不为 NA 的 userid 替换
for i in range(len_1-1):
    index = data_in[data_in['sessionid'] == rept_sessionid_1[i]].index
    rept_dat = data_in.iloc[index]['userid']
    data_in.iloc[index,]['userid'] = rept_dat[rept_dat.notnull()].head(1)
```

del userid_sessionid_count_1,rept_sessionid_1,len_1,i,rept_dat

data_out = data_in

return(data_out)

(5)运行完成后,对 Python 脚本组件右键,选择查看数据,如图 12 所示。

			预览数据			
page_path	username	userid	sessionid	ip	date_time	uniquevisitorio
I			D572D2142391610AED7 4F0BC757CDD58	60.191.123.80	2016-8-12 0:00	
1			05BB9C5E9EAB302601C 4AD4F99D5AC64	64.89.234.161	2016-8-12 0:08	
/jmgj/575.jhtml			EB8FBC1627FD5B564B A2E06722DBE707	66.249.65.162	2016-8-12 0:09	
1			2B9A9334275D6574C6C 9E0FADEC4389B	60.191.123.80	2016-8-12 0:09	
1			7CE918EFE6BE41E05F7 9BE46BC80CC99	207.46.13.157	2016-8-12 0:10	
/sjfxs/705.jhtml			0D71BB01AA9A61C07B BCAE3DAD3DFFEE	42.156.254.4	2016-8-12 0:15	
			578D4974B43E044279F			

图 12 数据变换结果

(6)运行完成后,对 Python 脚本组件右键,重命名为数据变换。

3.2.3 用户识别

接下来进行用户识别,步骤如图 13 所示。

(1) 找到预处理→Python 脚本组件。

(2) 拖入 Python 脚本组件,并将数据变换和 Python 脚本组件连接。

(3) 选择字段属性, 在脚本处填入数据变换代码, 如表 3-4 所示。

(4) 对 Python 脚本组件右键,选择运行该节点。

工程 0	🛓 Q Q 120% % 🖺 D	◇ 字段属性
 ▼我的工程 ▲ 竞赛网站用户 		输入
		input1 from 10007282_1_1
	属性规约	input2 from
组件	数据变换	input3 from
■ 排序 ■ 数据筛选 ■ 分组聚合	Python脚本	脚本 1 data_in - db_wtlls.guery(com, 'select ' from '* inputs['input]])
X 修改列名 X Python脚本 X 缺失值处理		 approve and set of the set of
▲ 50.7 天田50 ■ 特征构造 ■ 数据离散化		8 nong_userid = data_in[data_in['userid'] .notnull()] 9 # 将userid将为学符用 11 nong_userid['userid'] =
國 标准化数据还原 國 数据标准化 國 数据编码化		nona_userid['userid'].apply(str)
模型		> 组件描述

图 13 用户识别组件

表 3-4 用户识别代码

```
data_in = db_utils.query(conn, 'select * from ' + inputs['input1'])
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
# 按 userid 为 nan 与非 nan 进行拆分
na_userid = data_in[data_in['userid'].isnull()]
nona_userid = data_in[data_in['userid'].notnull()]
# 将 userid 转为字符串
nona_userid['userid'] = nona_userid['userid'].apply(str)
# 对 nona.userid 数据集中的"userid"重新赋值
# 如果 unique Visitor Id 不为空则用它作为 userid, 若它为空则用 ip 作为 userid
#将 na.userid 按 unique Visitor Id 是否为空区分
na_uniqueVisitiorId = na_userid[na_userid['uniquevisitorid'].isnull()]
nona_uniqueVisitiorId = na_userid[na_userid['uniquevisitorid'].notnull()]
# 替换
na_uniqueVisitiorId['userid'] = na_uniqueVisitiorId['ip']
nona_uniqueVisitiorId['userid'] = nona_uniqueVisitiorId['uniquevisitorid']
# 数据整合
user_data = pd.concat([nona_userid,na_uniqueVisitiorId,nona_uniqueVisitiorId])
# 构造字段 reallid
userid = user_data['userid'].drop_duplicates()
userid = pd.DataFrame({'userid' : userid,
                          'reallid' : range(1,userid.size+1)})
user_data = user_data.merge(userid)
del data_in,na_userid,nona_userid,na_uniqueVisitiorId,nona_uniqueVisitiorId,userid
```

data_out = pd.DataFrame(user_data)

return(data_out)

(5) 运行完成后,对 Python 脚本组件右键,选择查看数据,如图 14 所示。

			预览数据			
page_path	username	userid	sessionid	ip	date_time	uniquevisitorid
/index.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:24	c2a8a06c-abf5- d9428151f293
/zytj/index.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:25	c2a8a06c-abf5- d9428151f293
/ts/747.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:25	c2a8a06c-abf5 d9428151f293
1	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:26	c2a8a06c-abf5 d9428151f293
/wjxq/725.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:26	c2a8a06c-abf5 d9428151f293
/ts/747.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:27	c2a8a06c-abf5 d9428151f293
			7CF464AF4678EF8F50F			33188ba0-7880

图 14 用户识别结果

(6) 运行完成后,对 Python 脚本组件右键,重命名为用户识别。

3.2.4 行为分析

接下来进行行为分析,步骤如图 15 所示。

(1) 找到预处理→Python 脚本组件。

(2) 拖入 Python 脚本组件,并将用户识别和 Python 脚本组件连接。

(3)选择字段属性,在脚本处填入数据变换代码,如表 3-5 所示。

(4) 对 Python 脚本组件右键,选择运行该节点。

工程 〇	초 Q Q 120% % 🖹 O	◇ 字段屬性
▼我的工程		▲ 輸入
▲ 竞赛网站用户		input1 from 10007283_1_1
	网络 网	input2 from
		input3 from
组件	数据变换	input4 from
24 排序		
■ 数据筛选	▼ 用户识别	脚本
X 分组聚合		<pre>1 data_in - db_utils.query(conn, 'select *</pre>
■ 修改列名	行为分析	2 import warnings 3 warnings, filterwarnings("ignore")
X Python脚本		4 import pandas as pd
🗙 缺失值处理		6 # 提取reallid、sessionid并将它们唯一化 7 peallid sessionid = dta is[['reallid'
X 数学类函数		,'sessionid']].drop_duplicates()
X 特征构造		9 # reallid进行统计
■ 数据离散化		.value_counts()
🕱 标准化数据还原		11 12 # 提取只登录了一次的reallid
翼 数据标准化		
■ 数据编码化		
1070		> 组件描述

图 15 行为分析组件

表 3-5 行为分析代码

data_in = db_utils.query(conn, 'select * from ' + inputs['input1'])
import warnings
warnings.filterwarnings("ignore")
import pandas as pd

提取 reallid、sessionid 并将它们唯一化 reallid_sessionid = data_in[['reallid','sessionid']].drop_duplicates()

reallid 进行统计 reallid_count = reallid_sessionid['reallid'].value_counts()

提取只登录了一次的 reallid click_con_user = reallid_count[reallid_count == 1].index

提取登录一次用户的原始点击数据 click_con_data = data_in[data_in['reallid'].apply(lambda x:x in click_con_user)]

按 click.one.data 中的 realIID 进行统计 reallid_count_1 = click_con_data['reallid'].value_counts()

提取只登录了一次且只点击了一个网页的用户 one_click_user = reallid_count_1[reallid_count_1 == 1].index

从原始数据中去掉闲逛人员的数据
提取用户编号并唯一化
user = data_in['reallid'].drop_duplicates()

```
# 提取非"闲逛人员"的用户编号
user_1 = pd.DataFrame(list(set(user).difference(one_click_user)))
user_1.columns = ['reallid']
```

提取非"闲逛人员"的原始数据 new_data = pd.merge(data_in,user_1,how='inner',on='reallid')

del data_in,reallid_sessionid,reallid_count,click_con_user,\ click_con_data,reallid_count_1,one_click_user,user,user_1

data_out = pd.DataFrame(new_data)

return(data_out)

(5) 运行完成后,对 Python 脚本组件右键,选择查看数据,如图 16 所示。

bage_path	username	userid	sessionid	ip	date_time	uniquevisitorid
index.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:24	c2a8a06c-abf5 d9428151f293
zytj/index.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:25	c2a8a06c-abf d9428151f293
ts/747.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:25	c2a8a06c-abf d9428151f293
	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:26	c2a8a06c-abf d9428151f293
wjxq/725.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:26	c2a8a06c-abf d9428151f293
ts/747.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:27	c2a8a06c-abf d9428151f293
			7CF464AF4678EF8F50F			33188ba0-788

图 16 行为分析结果

(6) 运行完成后,对 Python 脚本组件右键,重命名为行为分析。

3.2.5 网址分析

接下来进行网址分析,步骤如图 17 所示。

(1) 找到预处理→Python 脚本组件。

(2) 拖入 Python 脚本组件,并将行为分析和 Python 脚本组件连接。

(3)选择字段属性,在脚本处填入数据变换代码,如表 3-6 所示。

(4) 对 Python 脚本组件右键,选择运行该节点。

工程 0	📩 Q Q 120% % 🖺 🔉	◇ 字段属性
▼我的工程 ▲ 竞赛网站用户	(二) 输入源	输入
		input1 from 10007286_1_1
		input2 from
组件	A SUBSCIP	input4 from
24 排序		
X 数据筛选	X 用户识别	脚本
■ 分组聚合 ■ 修改列名	E Fithout	<pre>1 data_in = db_utils.query(con, 'select *</pre>
X 缺失值处理	2 网址分析	5 6 # 新增一类序号 7 data_in['number_id'] = range(1,len(data_in)+1)
2 特征构造 2 教授国際部化		8 9 # 提取.jhtml结尾的数据 10 jhtml_data - data_in[data_in['page_path'] appl(Jambda xix.endsulth('.jhtml'))]
I 标准化数据还原		11 12 # 提取非.jhtml站尾的数据
翼 数据标准化		L
■ 数据编码化		
模型		> 组件描述

图 17 网址分析组件

表 3-6 网址分析代码

data_in = db_utils.query(conn, 'select * from ' + inputs['input1'])
import warnings
warnings.filterwarnings("ignore")
import pandas as pd

新增一类序号 data_in['number_id'] = range(1,len(data_in)+1)

提取.jhtml 结尾的数据 jhtml_data = data_in[data_in['page_path'].apply(lambda x:x.endswith('.jhtml'))]

提取非.jhtml 结尾的数据 #unjhtml_data = data_in[-data_in['number_id'].isin(jhtml_data['number_id'])]

统计各非 jhtml 网址的个数 #unjhtml_count = unjhtml_data['page_path'].value_counts()

data_out = pd.DataFrame(jhtml_data)

return(data_out)

(5)运行完成后,对 Python 脚本组件右键,选择查看数据,如图 18 所示。

			预览数据			
page_path	username	userid	sessionid	ip	date_time	uniquevisitorid
/index.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:24	c2a8a06c-abf5-0df d9428151f293
/zytj/index.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:25	c2a8a06c-abf5-0df d9428151f293
/ts/747.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:25	c2a8a06c-abf5-0d€ d9428151f293
/wjxq/725.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:26	c2a8a06c-abf5-0df d9428151f293
/ts/747.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:27	c2a8a06c-abf5-0d€ d9428151f293
/zytj/index.jhtml	gangzhi21cn	9384	7CF464AF4678EF8F50F 2AC7FBDC93852	10.130.142.131	2016-8-12 8:48	33188ba0-7880-0c 3-4dc196d459f6
			7CF464AF4678EF8F50F			33188ba0-7880-0c

图 18 网址分析结果

(6)运行完成后,对 Python 脚本组件右键,重命名为网址分析。

3.2.6 数据清洗

接下来进行数据清洗,步骤如图 19 所示。

- (1) 找到预处理→Python 脚本组件。
- (2) 拖入 Python 脚本组件,并将网址分析和 Python 组件连接。
- (3) 选择字段属性, 在脚本处填入数据变换代码, 如表 3-7 所示。
- (4) 对 Python 脚本组件右键,选择运行该节点。

 ● 読的工程 ▲ 完美网络用户… ● 読入選 	工程 O	🛓 Q Q 120% % 🖺 O	◇ 字段属性
▲ 究奈列以田户 第四4 第二 日本 第二 「日本 「日本 「日本	▼ 我的工程		输入
日井 「「「」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」	▲ 竞赛网站用户	≓ 输入源 ■	input1 from 10007287_1_1 input2 from
第時件 ※ 依然時語: 第一年現期 ※ 公司務合 第一年現期 ※ 修改身合 ※ 行为分析 ※ 外の内閣本 ※ 行为分析 ※ 於外面的 ※ 行为分析 ※ 於外面的 ※ 引entile(ispace)(cone, 'select from 'singletispace ※ 対応の関本 ※ 引entile(ispace) ※ 対応の関本 ※ 対応の関本 ※ 対応の関本 ※ 対応の構本 ※ 対応の構成 ※ 対応の構成 ※ 対応の構成 ※ 対応の構成	组件	风 数据变换	input3 from input4 from
K 数据前选 X 数据前选 X 如果合 X 分组聚合 X 分组聚合 X 价为分析 X 们 可 X ① 1 ① 1 ① 1 ① 1 ① 1 ① 1 ① 1 ① 1 ① 1 ①	24 排序		
X 分相聚合 X 分相聚合 X 分相聚合 X 分相聚合 X 分为分析 X 分析出版 X 分析出版 X 和 和 和 和 和 和 和 和 和 和 和 和 和 和 和 和 和	■ 数据筛选	X 用户识别	脚本
其他改列名 其一行为分析 2 isport surrings 其代内的内理本: 其 1 1 1 其於共倫处理 其一行为分析 2 1 1 其於共倫处理 其一行方分析 1 1 1 其於学派命故 其一行方分析 1 1 1 其於行为分析 第 7 1 1 其於行为分析 第 7 1 1 其於於行用力的反点击量 1 1 1 1 其於於形面 第 1 1 1 其於於影面高能化 其一式 1 1 1 其於認識而能化 其一式 1 1 1 其然前面電子 1 1 1 1 其的子 1 1 1 1 其的子 1 1 1 1 其的子 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 <td>X分组聚合</td> <td></td> <td><pre>1 data_in - db_utils.query(conn, 'select * from ' * inputs['input1'])</pre></td>	X 分组聚合		<pre>1 data_in - db_utils.query(conn, 'select * from ' * inputs['input1'])</pre>
M Python際本 4 isport pandas as pd M Byt-Gb/mg 5 # FireILDE THE GHT Philippe.gd.mg M Byt-Gb/mg FireILDE THE GHT Philippe.gd.	2 修改列名	(文) 行为分析	2 import warnings 3 warnings.filterwarnings("ignore")
X 結決儀处理 X 就学業品数 X 就学業品数 X 新石构造 X 所並公証 R 新石构造 X 所並公証 R 新石构造 X 新石构造 X 新石构造 X 新石构造 X 新石构造 X 新石构造 X 新石和合 X 和合 X 和合	X Python脚本		4 import pandas as pd 5
試数学奖曲数 7 total_clik - data_in[realid'] 試特征构造 7 total_clik - data_in[realid'] 減損益期防化 10 more40 user - total_click(total_click) 減損益加防症 11 data_in[-data_in['realid'] 減損損防止化 11 data_in[-data_in['realid'] 減損損防化 11 data_in - data_in['realid']	🗙 缺失值处理		6 W 对reallID进行统计 ,统计结果即为每位用户的历史点击量
M 特征物造 M 数据案件化 M 标准化数据还示 M 标准化数据环境化	■ 数学类函数	Mitta:salit	<pre>7 total_click - data_in['reallid'] .value_counts()</pre>
X 数据素数化 X 数据素数 10 more@user - total_click[total_click > index index index idex idex idex idex idex idex idex i	X 特征构造		8 9 # 去除点击量大于40用户的数据
其标准化发展还须。 其标准化发展还须。 其 新闻编码化化	■ 数据离散化	数据清洗	<pre>10 more40_user = total_click[total_click > 40] .index</pre>
其 数据标准化	X 标准化数据还原		<pre>11 data_in = data_in[-data_in['reallid'].isin</pre>
繁 教理编码化	翼 数据标准化		
	■ 数据编码化		

图 19 数据清洗组件

表 3-7 数据清洗代码

data_in = db_utils.query(conn, 'select * from ' + inputs['input1'])
import warnings
warnings.filterwarnings("ignore")
import pandas as pd

对 reallID 进行统计,统计结果即为每位用户的历史点击量 total_click = data_in['reallid'].value_counts()

去除点击量大于 40 用户的数据 more40_user = total_click[total_click > 40].index data_in = data_in[-data_in['reallid'].isin(more40_user)]

del total_click,more40_user

data_out = pd.DataFrame(data_in)

return(data_out)

(5)运行完成后,对 Python 脚本组件右键,选择查看数据,如图 20 所示。

			预览数据			
page_path	username	userid	sessionid	ip	date_time	uniquevisitorid
/index.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:24	c2a8a06c-abf5-0df d9428151f293
/zytj/index.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:25	c2a8a06c-abf5-0d6 d9428151f293
/ts/747.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:25	c2a8a06c-abf5-0d6 d9428151f293
/wjxq/725.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:26	c2a8a06c-abf5-0df d9428151f293
/ts/747.jhtml	yyang891010	9383	8F63C7A49D56BDA9706 68D57B43D1DF6	27.38.32.12	2016-8-12 1:27	c2a8a06c-abf5-0d6 d9428151f293
/zytj/index.jhtml	gangzhi21cn	9384	7CF464AF4678EF8F50F 2AC7FBDC93852	10.130.142.131	2016-8-12 8:48	33188ba0-7880-0c 3-4dc196d459f6
			7CF464AF4678EF8F50F			33188ba0-7880-0c
		共 16673 条 25 条/页	 ✓ ✓ 1 2 3 4 5 6 	** 667 〉 前往 1 页		

图 20 数据清洗结果

(6) 运行完成后,对 Python 脚本组件右键,重命名为数据清洗。

3.2.7 网页分析

接下来进行数据清洗,步骤如图 21 所示。

(1) 找到预处理→Python 脚本组件。

(2) 拖入 Python 脚本组件,并将数据清洗和 Python 组件连接。

(3) 选择字段属性, 在脚本处填入数据变换代码, 如表 3-8 所示。

(4) 对 Python 脚本组件右键,选择运行该节点。

I程 O	🛓 Q Q 🚺 🗞 🖹 O	✓ 字段属性
▼我的工程 ▲ 竞赛网站用户		输入 @
	図 開性現的	input2 from
组件	展中研制	input4 from
國 排序 國 数据筛选 國 分组聚合	日本 行为分析	脚本 1 data_in - db_utils.query(conn, 'select * from ' + inputs['input1'])
■ 修改列名 ■ Python脚本	网站分析	2 import warnings 3 warnings.flitewarnings("ignore") 4 import pandas as pd 5 import re
X 缺失值处理 X 数学关函数 X 特征构造	(文) 数据清先	7 # 提取page_path,reallID 8 mode_data - data_in[['reallid','page_path']] 9 # 删除含有吗?的记录
■ 数据离散化 ■ 标准化数据还原	R HERON	<pre>11 mode_data = mode_data[mode_data["sage_path"]apply(lambda x:len(re.findal1("%",x)) == 0)] 12 anno multitude</pre>
■ 数据标准化 ■ 数据编码化		
模型		> 组件描述

图 21 网页分析组件

表 3-8 网页分析代码

data_in = db_utils.query(conn, 'select * from ' + inputs['input1'])
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
import re
提取 page_path,realIID
mode_data = data_in[['reallid','page_path']]
删除含有 "%"的记录
mode_data = mode_data[mode_data['page_path'].apply(lambda x:len(re.findall('%',x)) == 0)]
提取网址标签
mode_data['page_path'] = mode_data['page_path'].apply(lambda x:x.replace('.jhtml',''))

```
mode_data['page_path'] = mode_data['page_path'].apply(lambda x:re.sub('\d',",x))
    mode_data['page_path'] = mode_data['page_path'].apply(lambda x:x.replace('index',"))
    mode_data['page_path'] = mode_data['page_path'].apply(lambda x:x.replace('/',''))
    mode_data['page_path'] = mode_data['page_path'].apply(lambda x:x.replace('_',"))
    # 删除空数据 主页/index
    mode_data = mode_data[mode_data['page_path'] != "]
    # 统计网页内容符号数, 删除一些异常的网页内容符号如: sdfasf、cook…
    url_conent = mode_data['page_path'].value_counts()
    url = url_conent.index
    error url = ['dsjkf','sjfxs','jingsa','sdfasf','cookie','jiao','jao','asdf','sjsdf','jmg']
    url = url[~url.isin(error_url)]
    ## 网页汇总
    mode_data = mode_data[mode_data['page_path'].isin(url)]
    mode_data['page_path'] = mode_data['page_path'].apply(lambda x:x.replace('firsttipdm','yxzp'))
    mode_data['page_path']
                                            =
                                                              mode_data['page_path'].apply(lambda
x:x.replace('secondtipdm','yxzp'))
    mode_data['page_path'] = mode_data['page_path'].apply(lambda x:x.replace('thirdtipdm','yxzp'))
    mode_data['page_path'] = mode_data['page_path'].apply(lambda x:x.replace('fourthtipdm','yxzp'))
    mode_data['page_path'] = mode_data['page_path'].apply(lambda x:x.replace('sm','jszz'))
    mode_data['page_path'] = mode_data['page_path'].apply(lambda x:x.replace('td','jszz'))
```

mode_data = mode_data.groupby(['reallid','page_path']).size().unstack().fillna(0).reset_index()

del url_conent,url,error_url

data_out = pd.DataFrame(mode_data)

return(data_out)

(5)运行完成后,对 Python 脚本组件右键,选择查看数据,如图 22 所示。

			预览数据			×
reallid	cgal	information	jmgj	jszz	jxsp	kjxm
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	1	0	0	5	0	0
6	0	0	0	0	0	0
7	0	0	0	0	0	0
8	0	0	0	0	0	0
		共 2826 条 25 条/页 🔻	< 1 2 3 4 5 6 -	** 114 〉 前往 1 页		

图 22 网页分析结果

(6) 运行完成后,对 Python 脚本组件右键,重命名为网页分析。

3.2.8 属性构造

接下来进行数据清洗,步骤如图 23 所示。

(1) 找到预处理→Python 脚本组件。

(2) 拖入 Python 脚本组件,并将网页分析和 Python 组件连接。

(3) 选择字段属性, 在脚本处填入数据变换代码, 如表 3-9 所示。

(4) 对 Python 脚本组件右键,选择运行该节点。

工程 〇	🛓 Q Q 🚺 🗞 🖹 🗘	◇ 字段罵性
▼我的工程 ▲ 党赛网站用户		● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
		input2 from input3 from
组件 区排序	国行行王王	input4 from
■ 数据筛选 ■ 分组聚合	【 行为分析	脚本 1 data_in = db_utils.query(conn, 'select * from '+ inputs['input']) 2 inport wardings
X Python脚本 X 缺失值处理	Ritt:Set	3 warnings.filterwarnings("ignore") 4 import pandas as pd 5 # 一级栏目汇总 7 data_in["贪夷可许奖'] = data_in[['stpj','qk'
■ 数学类函数 ■ 特征构造 ■ 教祝密散化	又 网页公析	<pre>, 'notice']].sum(axis-1) 8 data_in[新闻内苏] - data_in[['s]' ,'notices','neus']].sum(axis-1) 9 data_in['教子页语'] - data_in[[']xsp','ts' ,'jmgi', 'information','zytj']].sum(axis</pre>
國 标准化数据还原 國 数据标准化	1000 一個人的	-1)
A SUBMUTER		> 担件描述

图 23 属性构造组件

表 3-9 属性构造代码

data_in = db_utils.query(conn, 'select * from ' + inputs['input1'])
import warnings
warnings.filterwarnings("ignore")
import pandas as pd
一级栏目汇总
data_in['竞赛与评奖'] = data_in[['stpj','qk','notice']].sum(axis=1)
data_in['新闻动态'] = data_in[['sip','notices','news']].sum(axis=1)
data_in['教学资源'] = data_in[['jxsp','ts','jmgj','information','zytj']].sum(axis=1)
data_in['项目与招聘'] = data_in[['xtxm','wjxq','zxns']].sum(axis=1)
data_in['创新与合作'] = data_in[['yxzp']].sum(axis=1)
data_in['完赛组织'] = data_in[['jxz']].sum(axis=1)

提取建模数据,即:建模样本表

data_in = data_in[['reallid','竞赛与评奖','新闻动态','教学资源','项目与招聘','创新与合作','优秀 作品','竞赛组织']]

data_in.rename(columns={'reallid':'用户编号'},inplace = True)

data_out = pd.DataFrame(data_in)

return(data_out)

(5)运行完成后,对 Python 脚本组件右键,选择查看数据,如图 24 所示。

			预览数据			
用户编号	竟赛与评奖	新闻动态	教学资源	项目与招聘	创新与合作	优秀作品
1	0	0	3	1	0	0
2	0	0	3	0	0	0
3	0	0	2	0	0	2
4	0	0	2	0	0	0
5	9	18	1	1	1	0
6	0	0	1	0	0	0
7	0	0	1	0	0	0
8	0	0	1	0	0	0
		共 2826 条 25 条/页	 ✓ 1 2 3 4 	5 6 … 114 〉 前往 1	页	

图 24 属性构造结果

(6)运行完成后,对 Python 脚本组件右键,重命名为属性构造。

3.3 模型构建

3.3.1 K-Means 聚类算法

选择 K-Means 聚类算法模型,步骤如如图 25、图 26 所示。

- (1) 找到聚类→K-Means 组件。
- (2) 拖入 K-Means 组件,将属性构造和 K-Means 组件连接。
- (3) 选择字段属性,单击更新数据,勾选除了用户编号外的全部字段。
- (4) 选择参数设置,设置聚类数(n_clusters)的值为5,其他的参数都设置为默认值。

		◆ 子校周生	
大的上程	(二) 输入源	特征	•
▲ 完積明始用广			
	网 属性规约	添加字段过滤字符串	
	■ 数据变换	字段 类型 取值范围	l I
组件		用户编号 数值 -	
· ····	「用户识别」	✓ 竞赛与评奖 数值 0-6	
▶ 分类	CHANE	✓ 新闻动态 数值 0-4	
	TPCCCLE M	✔ 教学资源 数值 0-7	
	网址分析	921 E3 1-47200 White 0.0	
● 高斯混合模型			
♥ DBSCAN密度	SUSTAINT.		
♥K-中心点繁美 ♥K-Means	网页分析		
▶ 时序模型		> 基础参数	
▶ 关联规则	■性協造		
▶ 模型评估		7 million \$2	

图 25 M-Means 聚类组件_字段属性

工程 〇	📩 Q Q <mark>120%</mark> % 🖺 O	> 字段屬性
▼我的工程 ▲ 党赛网站用户		✓ 基础伊政 預決政 5
60.04	数据安换	最大迭代次数
HIT+ ▶ 统计分析		
 ▶ 分类 ▶ 回归 ▼ 服美 	文 行为分析	
 ● 层次聚类 ● 高斯混合模型 		
♥ DBSCAN密度 ♥ K-中心点聚类 ♥ K-Means	网页分析	
 ▶ 时序-模型 ▶ 关联规则 	属性物語	N WRAN
●模型评估	CH K-Means	> 四%%%%X

图 26 K-Means 组件_参数设置

(5) 运行完成后,对 K-Means 组件右键,选择查看数据,K-Means 的输出表结果如图 27 所示。选择查看报告,K-Means 的报告如图 28 所示。

			预览数据			×
新闻动态	教学资源	项目与招聘	创新与合作	优秀作品	竟賽组织	cluster_id
0	3	1	0	0	0	1
0	3	0	0	0	0	1
0	2	0	0	2	0	2
0	2	0	0	0	0	2
18	1	1	1	0	5	4
0	1	0	0	0	0	2
0	1	0	0	0	0	2
0	1	0	0	0	0	2
		共 2826 条 25 条/页 🔍	<pre>1 2 3 4 5 6</pre>	114 〉 前往 1 页		

图 27 K-Means 聚类算法的结果

					算法运行报告				
				ŀ	(-Means算法	信果			
(模型参数				
				输出香	國的參数以及參考	如的取值。			
参数名称	参数值								
聚类个数	5								
最大迭代次数	100								
(聚类中心:				
cluster_id	竞赛与评奖	新闻动态	教学资源	项目与招聘	创新与合作	优秀作品	竞赛组织		

图 28 K-Means 聚类算法的报告