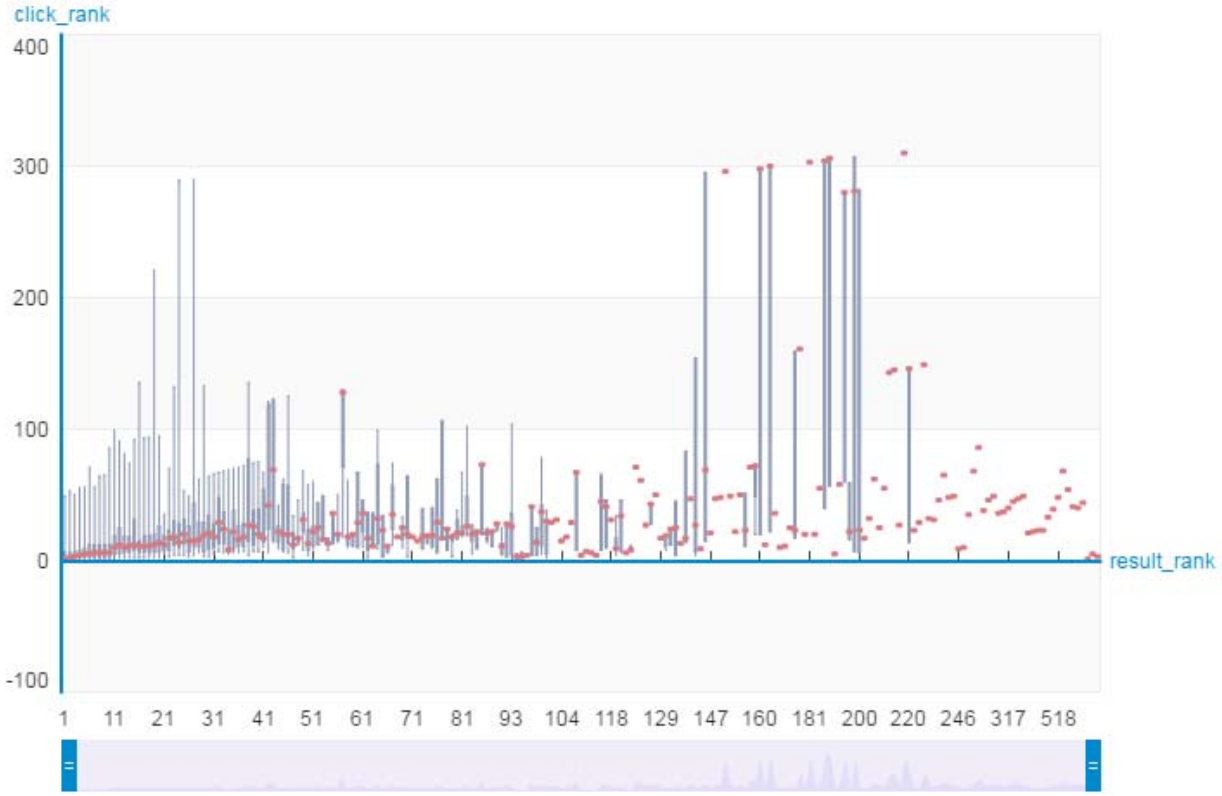
基于查询日志的查询意图识别研究

1. 讲在前面

1.1 研究背景

互联网（World Wide Web）是世界上最大的信息资源库，是人们快速获取信息的重要途径之一，极大的改变了人们的生活方式。据2016年1月中国互联网络信息中心（CNNIC）发布的第37次《中国互联网络发展状况统计报告》显示，截至2015年12月，中国网民已达6.88亿，互联网普及率高达50.3%。截至2015年12月，中国网站数量为423万个，年增长26.3%，中国网页的数量为2123亿个，年增长11.8%，如图所示。

由于用户提交的查询往往较短，自然语言存在模糊性，无法清晰的表达用户的意图。所以包含查询关键字的查询结果，有可能不足以满足用户的需求；搜索引擎返回的数以千计的文档也造成严重的信息过载。有研究显示，在查询日志中，有至少16%的歧义查询，有超过75%的查询具有更复杂的信息需求，难以被简单的答案或者特定的网址满足。



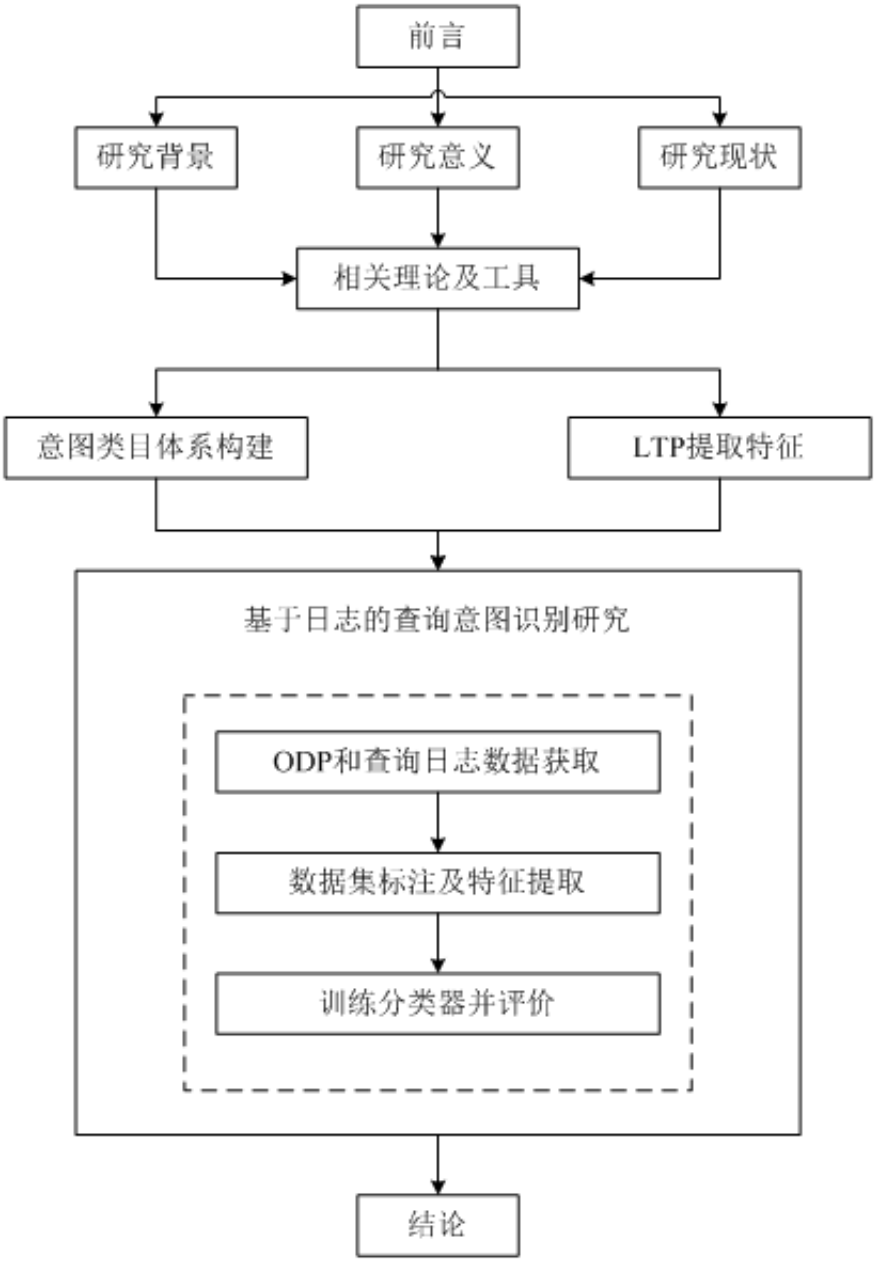
1.2 研究意义

- 增加用户对搜索结果的满意度：正确对用户意图进行识别，可以有效解决“信息过载”等问题，提高搜索效率；
- 提高广告推荐的精度：向搜索引擎用户提供和用户当前搜索相关的广告，提高商业价值；
- 帮助搜索引擎组织和检索信息：可以根据意图整理网络上的信息资源，建立更高效的检索系统。

1.3 国内外研究现状

- 查询意图类目体系构建
- 查询意图特征提取
- 查询意图的识别方法研究
- 数据集与评价方法

1.4 论文框架



2. 相关理论介绍

3. 实验

导入要用到的python程序库，并且设置数据展示在notebook中。

In [1]:

```
import graphlab
import re
import pandas as pd
import string
import jieba.posseg as pseg
import jieba
from collections import OrderedDict
from collections import Counter
```

In [2]:

```
graphlab.canvas.set_target('ipynb')
```

3.1 搜集数据

3.1.1 查询日志数据（subset）

In [3]:

```
data = graphlab.SFrame.read_csv('SogouM.txt', delimiter='\t', header=True, column_type_hints=
```

```
This non-commercial license of GraphLab Create is assigned to guoxiuhe@nefu.
edu.cn and will expire on April 02, 2017. For commercial licensing options,
visit https://dato.com/buy/. (https://dato.com/buy/.)

2016-05-05 20:08:41,888 [INFO] graphlab.cython.cy_server, 176: GraphLab Crea
te v1.9 started. Logging: C:\Users\heguoxiu\AppData\Local\Temp\graphlab_serv
er_1462450115.log.0

Finished parsing file E:\Python\2Graduation-Project\SogouM.txt

Parsing completed. Parsed 10000 lines in 0.124801 secs.
```

In [4]:

```
data.head()
```

Out[4]:

time	user_id	query	result_click	u
00:00:00	2982199073774412	[360安全卫士]	8 3	download.it.c b/software/
00:00:00	07594220010824798	[哄抢救灾物资]	1 1	news.21cn.cc an/2008/05/2
00:00:00	5228056822071097	[75810部队]	14 5	www.greatoo.c n/list.asp?li
00:00:00	6140463203615646	[绳艺]	62 36	www.jd-cd.co 200607/7
00:00:00	8561366108033201	[汶川地震原因]	3 2	www.bi
00:00:00	23908140386148713	[莫衷一是的意思]	1 2	www.chinaba e/81/82/110/
00:00:00	1797943298449139	[星梦缘全集在线观看] ...	8 5	www.6wei.net ??\xa1\xe9|??
00:00:00	00717725924582846	[闪字吧]	1 2	www.shan
00:00:00	41416219018952116	[霍震霆与朱玲玲照片] ...	2 6	bbs.gouzai.cr 36.ht
00:00:00	9975666857142764	[电脑创业]	2 2	ks.cn.yahoo.c 130712020

[10 rows x 5 columns]

In [5]:

```
data['result_and_click'] = data['result_click'].apply(lambda x : re.split(r'\s+',x))
```

In [6]:

```
def getfirst(l):
    return int(l[0])
def getsecond(l):
    return int(l[1])

data['result_rank'] = data['result_and_click'].apply(lambda l: getfirst(l))
data['click_rank'] = data['result_and_click'].apply(lambda l: getsecond(l))
```

In [7]:

```
data['query'] = data['query'].apply(lambda x: str(x)).apply(lambda x: x.strip().lstrip('[').
```

In [8]:

```
def remove_tail(url):
    if url[-1] == '/':
        return url[:-1]
    else:
        return url[:]
```

In [9]:

```
data['url'] = data['url'].apply(remove_tail)
```

In [10]:

```
data.head()
```

Out[10]:

time	user_id	query	result_click	
00:00:00	2982199073774412	360安全卫士	8 3	download.it b/softwar
00:00:00	07594220010824798	哄抢救灾物资	1 1	news.21cn. an/2008/05
00:00:00	5228056822071097	75810部队	14 5	www.greatoc n/list.asp'
00:00:00	6140463203615646	绳艺	62 36	www.jd-cd.c 200607
00:00:00	8561366108033201	汶川地震原因	3 2	www.
00:00:00	23908140386148713	莫衷一是的意思	1 2	www.chinat e/81/82/11

00:00:00	1797943298449139	星梦缘全集在线观 ◆◆...	8 5	www.6wei.r ??\xa1\xe9|
00:00:00	00717725924582846	闪字吧	1 2	www.sh
00:00:00	41416219018952116	霍震霆与朱玲玲照 ◆◆...	2 6	bbs.gouzai. 36.
00:00:00	9975666857142764	电脑创业	2 2	ks.cn.yaho 13071202

click_rank
3
1
5
36

3.1.2 Open Directory Project(ODP) 体系

Open Directory Project			
主题	数量	主题	数量
休闲	529	新闻	552
体育	244	游戏	576
健康	750	社会	1557
儿童	240	科学	923
参考	2220	艺术	960
商业	5600	计算机	1639
家庭	114	购物	430

In [11]:

```
odp = graphlab.SFrame.read_csv('ODP.csv', delimiter=',', header=True)
```

```
Finished parsing file E:\Python\2Graduation-Project\ODP.csv

Parsing completed. Parsed 100 lines in 0.153802 secs.

------------------------------------------------------
Inferred types from first 100 line(s) of file as
column_type_hints=[str,str,str,str]
If parsing fails due to incorrect types, you can correct
the inferred type list above and pass it to read_csv in
the column_type_hints argument
------------------------------------------------------

Finished parsing file E:\Python\2Graduation-Project\ODP.csv

Parsing completed. Parsed 16248 lines in 0.068603 secs.
```

In [12]:

```
odp.head()
```

Out[12]:

url	name	label	url_new
http://news.jmu.edu.cn/	集美大学新闻网	大专院校	news.jmu.edu.cn/
http://jjxj.swufe.edu.cn/	经济学家	出版物	jjxj.swufe.edu.cn/
http://www.jsacd.gov.cn/	江苏省农业资源开发局 ...	江苏	www.jsacd.gov.cn/
http://www.yndaily.com/	云南日报网	地区	www.yndaily.com/
http://www.panda.org.cn/	成都大熊猫繁育研究基地 ...	熊猫	www.panda.org.cn/
http://www.fjinfo.gov.cn/	福建科技信息	福建	www.fjinfo.gov.cn/
http://www.klxuexi.com/	快乐学习教育科技团 ...	上海	www.klxuexi.com/
http://www.haier.com/cn/	海尔集团	消费电子产品	www.haier.com/cn/
http://www.jstvu.edu.cn/	江苏广播电视大学	江苏	www.jstvu.edu.cn/
http://www.gxtc.edu.cn/	广西师范学院	大专院校	www.gxtc.edu.cn/

[10 rows x 4 columns]

In [13]:

```
def remove_head(url):
    return string.replace(url, 'http://', '')
```

In [14]:

```
def remove_tail(url):
    if url[-1] == '/':
        return url[:-1]
    else:
        return url[:]
```

In [15]:

```
odp['url_new'] = odp['url'].apply(remove_head).apply(remove_tail)
```

In [16]:

```
odp.head()
```

Out[16]:

url	name	label	url_new
http://news.jmu.edu.cn/	集美大学新闻网	大专院校	news.jmu.edu.cn
http://jjxj.swufe.edu.cn/	经济学家	出版物	jjxj.swufe.edu.cn
http://www.jsacd.gov.cn/	江苏省农业资源开发局 ...	江苏	www.jsacd.gov.cn
http://www.yndaily.com/	云南日报网	地区	www.yndaily.com
http://www.panda.org.cn/	成都大熊猫繁育研究基地 ...	熊猫	www.panda.org.cn
http://www.fjinfo.gov.cn/	福建科技信息	福建	www.fjinfo.gov.cn
http://www.klxuexi.com/	快乐学习教育科技团 ...	上海	www.klxuexi.com
http://www.haier.com/cn/	海尔集团	消费电子产品	www.haier.com/cn
http://www.jstvu.edu.cn/	江苏广播电视大学	江苏	www.jstvu.edu.cn
http://www.gxtc.edu.cn/	广西师范学院	大专院校	www.gxtc.edu.cn

[10 rows x 4 columns]

3.2 构建新的类目体系以标注查询日志数据

3.2.1 将ODP主题类目体系映射到Rose类目体系

- Rose类目体系

层级	解释	例子
(N) 导航类 (Navigation)	用户为了获得一个明确的网址	公司、学校等的主页
(I) 信息 (Imformation)	用户为了获得数据或信息	某条法律条款的解释
(R) 资源类 (Resource)	用户为了获得有用的资源	购买物品、游戏等
(N.T) 事务类导航 (Navigation to Transactional)	用户用来处理事务的导航网址	match.com
(N.T) 信息类导航 (Navigation to Information)	用户用来获取信息的网址	Yahoo.com
(I.D) 有指导性的 (Directed)	用户为了获取某个特定问题的答案	哈尔滨市的邮编
(I.U) 无指导性的 (Undirected)	用户为了获取一个主题的所有信息	2016 年新出电视剧的信息
(I.F) 发现 (Find)	用户为了获得一个产品或者服务的具体位置	哈尔滨中央大街的位置
(I.L) 列表 (List)	用户为了获得一组可信的站点列表	电子商务网站有哪些
(I.A) 建议 (Advice)	用户为了获得某个主题的建议、观点和指南等	如何高效率的学习
(R.O) 获取 (Obtain)	用户为了获得一个明确的资源或项目	某首歌的歌词
(R.D) 下载 (Download)	用户为了把某个资源下载到本地	电影、音乐、小说和论文等的下载
(R.E) 娱乐 (Entertainment)	用户可以在网页上进行的娱乐活动	游戏、聊天等
(R.I) 交互 (Interact)	用户与网络上的程序或者资源进行交互	在淘宝网上购买商品
(I.D.C) 确定的 (Closed)	用户为了获得一个问题的无歧义回答	宪法的第 3 条
(I.D.O) 开放的 （Opened）	用户为了获得两个或更多个的信息	人类的免疫系统
(R.O.O) 在线的 (Online)	用户需要在线获取	火车票的余票信息
(R.O.F) 离线的 (Off-line)	用户可以离线获得资源	—

通过对Rose分类体系的分析，本文将ODP主题类目体系映射到Rose类目体系的三大类即信息类 、资源类 和导航类 中，主要是信息类 和导航类。然后对查询日志数据进行标注。

- 导航类：本文将日志数据中url仅能匹配到web服务器名称的标记为导航类。如：www.nefu.edu.cn
- 资源类：本文首先利用启发式的方法，把日志数据中url能匹配到download/game/music/movie/book等字符的标记为资源类。然后利用上述的分析，人工筛选出ODP中属于资源类的url，构建资源类url库*resource*，当日志数据中的url可以匹配到*resource*时，标记为资源类。例如：ODP中的购物类、游戏类等都属于资源类。

- 信息类：将其他不属于以上两类的标注为信息类。

In [17]:

```
r = ['二手货','交通工具','休闲','体育用品','健康饮食','健康器材','在线销售','化妆美容','出版物
     '婴幼儿用品','宠物','家具','家居与园艺','批发','日用商品','服装饰品','消费电子产品','图书
     '珠宝首饰','礼品','视觉艺术','计算机','食品','鲜花','精油香氛','分类','拍卖','目录','大学
    '乒乓球','渔具','飞镖','家具','文具','办公室服务','购物','批发与分销','烟草','机动车','珠宝
     '鞋帽','饰品','电子通讯','数字卡','虚拟物品交易','摄影','画','饮料','茶','葡萄酒',\
    '卡牌游戏','投币式游戏','棋类游戏','牌类游戏','电子游戏','电脑游戏','益智游戏','网络游戏',
        '中国象棋','军棋','围棋','国际象棋','连珠','黑白棋','组织','休闲','体育','冒险','动作'
        '角色扮演','赛车','音乐与舞蹈','射击','格斗','网页游戏','魔兽争霸','魔兽世界','大型多
      '网络泥巴','角色扮演','冒险岛','天龙八部','永恒之塔','魔兽世界','手持平台','游戏机平台'
        '世嘉','任天堂','微软','索尼','下载','下载','会议展览','作弊与攻略','家族与公会','开发
        '电子竞技','聊天与论坛','麻将','体育','彩票','赌场']
```

In [18]:

```
f = open('resource_show.csv', 'a')
```

In [19]:

```
for i in r:
    for url in odp[odp['label'] == i]['url_new']:
        f.writelines(url+'\n')
f.close()
```

In [20]:

```
resource = graphlab.SFrame.read_csv('resource_show.csv',header=False)
```

```
Finished parsing file E:\Python\2Graduation-Project\resource_show.csv

Parsing completed. Parsed 100 lines in 0.037602 secs.

------------------------------------------------------
Inferred types from first 100 line(s) of file as
column_type_hints=[str]
If parsing fails due to incorrect types, you can correct
the inferred type list above and pass it to read_csv in
the column_type_hints argument
------------------------------------------------------

Finished parsing file E:\Python\2Graduation-Project\resource_show.csv

Parsing completed. Parsed 4043 lines in 0.028002 secs.
```

In [21]:

```
resource.unique().save('resource_show.csv')
```

In [22]:

```
resource = graphlab.SFrame.read_csv('resource_show.csv',header=False)
```

```
Finished parsing file E:\Python\2Graduation-Project\resource_show.csv

Parsing completed. Parsed 100 lines in 0.028601 secs.

------------------------------------------------------
Inferred types from first 100 line(s) of file as
column_type_hints=[str]
If parsing fails due to incorrect types, you can correct
the inferred type list above and pass it to read_csv in
the column_type_hints argument
------------------------------------------------------

Finished parsing file E:\Python\2Graduation-Project\resource_show.csv

Parsing completed. Parsed 1603 lines in 0.015001 secs.
```

In [23]:

```
resource.head()
```

Out[23]:

X1
X1
wow.uuu9.com/immtc
www.bmw-motorsport.com.cn
zh.wikipedia.org/zh-cn/Xbox ...
www.zglyyx.com
www.csapa.org
app.hicloud.com
www.e800.com.cn
d3.178.com
sports.sohu.com/weiqi.shtml ...

[10 rows x 1 columns]

3.2.2 对查询日志数据进行标注

标注资源类数据的方法

In [24]:

```
def label_resource(row, resource):
    for url_i in resource['X1']:
        if row['url'].find(url_i) != -1:
            return True
    return False
```

对整个数据集进行标注的方法

In [25]:

```
def log_label(row, resource):
    if re.match(r'www(.*?)\b(com|cn|org|net|gov|xin|red|pub|ink|info|xyz|win|edu|mil|tv|TV|mo
                row['url']):
        return 'Navigation'
    elif label_resource(row, resource):
        return 'Resource'
    else:
        return 'Information'
```

In [26]:

```
data['label'] = data.apply(lambda x: log_label(x, resource))
```

In [27]:

```
data['label'].show()
```

3.3 利用NLP技术提取特征

本文主要利用NLP技术来提取Query中的特征，包括：分词、词性统计等特征。结合点击排序特征和结果排序特征共同作为查询意图识别的特征。

In [28]:

```
def cut(query):
    flag_cut = ''
    temp = pseg.cut(query)
    for word, flag in temp:
        flag_cut = flag_cut + ' ' + flag
    return flag_cut
```

In [29]:

```
data['flag_cut'] = data['query'].apply(cut)
```

```
Building prefix dict from the default dictionary ...
2016-05-05 20:13:07,861 [DEBUG] jieba, 111: Building prefix dict from the de
fault dictionary ...
Loading model from cache c:\users\heguoxiu\appdata\local\temp\jieba.cache
2016-05-05 20:13:07,864 [DEBUG] jieba, 131: Loading model from cache c:\user
s\heguoxiu\appdata\local\temp\jieba.cache
Loading model cost 1.353 seconds.
2016-05-05 20:13:09,216 [DEBUG] jieba, 163: Loading model cost 1.353 seconds
.
Prefix dict has been built succesfully.
2016-05-05 20:13:09,230 [DEBUG] jieba, 164: Prefix dict has been built succe
sfully.
```

In [30]:

```
def search_cut(query):
    query_search_cut = ''
    temp = jieba.cut_for_search(query)
    for word in temp:
        query_search_cut = query_search_cut + ' ' + word
    return query_search_cut
```

In [31]:

```
data['query_search_cut'] = data['query'].apply(search_cut)
```

In [32]:

```
data.head()
```

Out[32]:

time	user_id	query	result_click	
00:00:00	2982199073774412	360安全卫士	8 3	download.it b/softwar
00:00:00	07594220010824798	哄抢救灾物资	1 1	news.21cn. an/2008/05
00:00:00	5228056822071097	75810部队	14 5	www.greatoc n/list.asp'
00:00:00	6140463203615646	绳艺	62 36	www.jd-cd.c 200607
00:00:00	8561366108033201	汶川地震原因	3 2	www.
00:00:00	23908140386148713	莫衷	1 2	www.chinat

		一定的意思		e/81/82/11
00:00:00	1797943298449139	星梦缘全集在线观 ◆◆...	8 5	www.6wei.r ??\xa1\xe9|?
00:00:00	00717725924582846	闪字吧	1 2	www.sh
00:00:00	41416219018952116	霍震霆与朱玲玲照 ◆◆...	2 6	bbs.gouzai. 36.
00:00:00	9975666857142764	电脑创业	2 2	ks.cn.yahoo 13071202

click_rank	label	flag_cut	query_search_cut
3	Information	m nz	360 安全 卫士 安全卫士 ...
1	Information	v l	哄抢 救灾 物资 救灾物资 ...
5	Information	m n	75810 部队

3.4 比较Logistic Regression和Boost Tree对查询意图识别的效率

3.4.1 数据准备(train_data, validation_data, test_data)

In [33]:

```
data['flag_word_count'] = graphlab.text_analytics.count_words(data['flag_cut'])
```

In [34]:

```
data['query_search_word_count'] = graphlab.text_analytics.count_words(data['query_search_cut'
```

In [35]:

```
data['tfidf'] = graphlab.text_analytics.tf_idf(data['flag_word_count'])
```

In [36]:

```
data['search_tfidf'] = graphlab.text_analytics.tf_idf(data['query_search_word_count'])
```

In [37]:

```
train_data, test_data = data.random_split(.8, seed=0)
```

In [38]:

```
train_data, validation_data = train_data.random_split(0.75, seed=0)
```

3.4.2 Logistic Regression

- 分类器训练

In [39]:

```
logistic_model_1 = graphlab.logistic_classifier.create(train_data, target='label', \
                                                       features=['tfidf', 'search_tfidf', 're
                                                       validation_set=validation_data)
```

```
Logistic regression:
--------------------------------------------------------
Number of examples          : 6010
Number of classes           : 3
Number of feature columns   : 4
Number of unpacked features : 5053
Number of coefficients    : 10108
Starting L-BFGS
--------------------------------------------------------
+-----------+----------+-----------+--------------+-------------------+-----
---------------+
| Iteration | Passes   | Step size | Elapsed Time | Training-accuracy | Vali
dation-accuracy |
+-----------+----------+-----------+--------------+-------------------+-----
---------------+
| 1         | 3        | 0.000166  | 1.116007     | 0.911814          | 0.87
7349            |
| 2         | 5        | 1.000000  | 1.168808     | 0.931780          | 0.87
8338            |
| 3         | 6        | 1.000000  | 1.223608     | 0.941265          | 0.87
```

```
4382            |
| 4          | 7         | 1.000000   | 1.262610      | 0.941764           | 0.87
4382            |
| 5          | 8         | 1.000000   | 1.307613      | 0.946755           | 0.87
1414            |
| 6          | 9         | 1.000000   | 1.355616      | 0.948087           | 0.87
2404            |
+-----------+----------+-----------+-------------+-------------------+-----
----------------+

TERMINATED: Iteration limit reached.

This model may not be optimal. To improve it, consider increasing `max_itera
tions`.
```

In [40]:

```
logistic_model_2 = graphlab.logistic_classifier.create(train_data, target='label', \
                                                       features=['search_tfidf', 'result_rank'
                                                       validation_set=validation_data)
```

```
Logistic regression:
--------------------------------------------------------

Number of examples          : 6010

Number of classes           : 3

Number of feature columns   : 3

Number of unpacked features : 5007

Number of coefficients    : 10016

Starting L-BFGS
--------------------------------------------------------

+-----------+----------+-----------+-------------+-------------------+-----
----------------+

| Iteration | Passes   | Step size | Elapsed Time | Training-accuracy | Vali
dation-accuracy |

+-----------+----------+-----------+-------------+-------------------+-----
----------------+

| 1         | 3        | 0.000166  | 0.015600     | 0.923627          | 0.87
1414            |
| 2         | 5        | 1.000000  | 0.062400     | 0.934443          | 0.87
3887            |
| 3         | 6        | 1.000000  | 0.124003     | 0.940266          | 0.86
```

```
7953            |

| 4          | 7          | 1.000000   | 0.168006       | 0.942097            | 0.87
2898            |

| 5          | 8          | 1.000000   | 0.217008       | 0.946090            | 0.86
8447            |

| 6          | 9          | 1.000000   | 0.260011       | 0.948419            | 0.86
8447            |

+------------+----------+------------+--------------+-------------------+-----
----------------+

TERMINATED: Iteration limit reached.

This model may not be optimal. To improve it, consider increasing `max_itera
tions`.
```

- 分类器测试

In [41]:

```
logistic_model_1.evaluate(test_data)
```

Out[41]:

```
{'accuracy': 0.8689024390243902,
 'auc': 0.7889915995372349,
 'confusion_matrix': Columns:
        target_label    str
        predicted_label str
        count   int

 Rows: 9

 Data:
 +--------------+-----------------+-------+
 | target_label | predicted_label | count |
 +--------------+-----------------+-------+
 |  Navigation  |    Navigation   |  103  |
 | Information  |    Navigation   |   76  |
 | Information  |     Resource    |   23  |
 |  Navigation  |   Information   |  117  |
 |   Resource   |   Information   |   40  |
 | Information  |   Information   |  1600 |
 |  Navigation  |     Resource    |   1   |
 |   Resource   |     Resource    |   7   |
 |   Resource   |    Navigation   |   1   |
 +--------------+-----------------+-------+
 [9 rows x 3 columns],
 'f1_score': 0.5389522755075119,
 'log_loss': 0.46973028282117485,
 'precision': 0.5695572718513214,
 'recall': 0.5178757038047105,
 'roc_curve': Columns:
        threshold       float
```

```
        fpr     float
        tpr     float
        p       int
        n       int
        class   int

Rows: 300003

Data:
+-----------+-----+----------------+------+-----+-------+
| threshold | fpr |      tpr       |  p   |  n  | class |
+-----------+-----+----------------+------+-----+-------+
|    0.0    | 1.0 |      1.0       | 1699 | 269 |   0   |
|   1e-05   | 1.0 |      1.0       | 1699 | 269 |   0   |
|   2e-05   | 1.0 | 0.999411418481 | 1699 | 269 |   0   |
|   3e-05   | 1.0 | 0.999411418481 | 1699 | 269 |   0   |
|   4e-05   | 1.0 | 0.999411418481 | 1699 | 269 |   0   |
|   5e-05   | 1.0 | 0.999411418481 | 1699 | 269 |   0   |
|   6e-05   | 1.0 | 0.999411418481 | 1699 | 269 |   0   |
|   7e-05   | 1.0 | 0.999411418481 | 1699 | 269 |   0   |
|   8e-05   | 1.0 | 0.999411418481 | 1699 | 269 |   0   |
|   9e-05   | 1.0 | 0.999411418481 | 1699 | 269 |   0   |
+-----------+-----+----------------+------+-----+-------+
[300003 rows x 6 columns]
Note: Only the head of the SFrame is printed.
You can use print_rows(num_rows=m, num_columns=n) to print more rows and co
lumns.}
```

In [42]:

```
logistic_model_2.evaluate(test_data)
```

Out[42]:

```
{'accuracy': 0.8648373983739838,
 'auc': 0.7887784476518876,
 'confusion_matrix': Columns:
        target_label    str
        predicted_label str
        count   int

 Rows: 9

 Data:
+--------------+-----------------+-------+
| target_label | predicted_label | count |
+--------------+-----------------+-------+
| Information  |     Resource    |   20  |
| Information  |    Navigation   |   85  |
|  Navigation  |    Navigation   |  102  |
|  Navigation  |   Information   |  118  |
|   Resource   |   Information   |   41  |
| Information  |   Information   |  1594 |
|  Navigation  |     Resource    |   1   |
|   Resource   |     Resource    |   6   |
|   Resource   |    Navigation   |   1   |
+--------------+-----------------+-------+
[9 rows x 3 columns],
```

```
 'f1_score': 0.5274333672364083,
 'log_loss': 0.479065690164296,
 'precision': 0.5580245864682271,
 'recall': 0.5082458006972427,
 'roc_curve': Columns:
        threshold       float
        fpr     float
        tpr     float
        p       int
        n       int
        class   int

 Rows: 300003

 Data:
 +-----------+-----+-----+------+-----+-------+
 | threshold | fpr | tpr |  p   |  n  | class |
 +-----------+-----+-----+------+-----+-------+
 |    0.0    | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   1e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   2e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   3e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   4e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   5e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   6e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   7e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   8e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   9e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 +-----------+-----+-----+------+-----+-------+
 [300003 rows x 6 columns]
 Note: Only the head of the SFrame is printed.
 You can use print_rows(num_rows=m, num_columns=n) to print more rows and co
lumns.}
```

3.4.3 Boosted Tree

- 分类器训练

In [43]:

```
boost_model_1 = graphlab.boosted_trees_classifier.create(train_data, target='label', \
                                                         features=['tfidf', 'search_tfidf', '
                                                         validation_set=validation_data)
```

```
Boosted trees classifier:
--------------------------------------------------------

Number of examples          : 6010

Number of classes           : 3

Number of feature columns   : 4

Number of unpacked features : 5053

+-----------+--------------+-------------------+-------------------+--------
------------+---------------------+

| Iteration | Elapsed Time | Training-accuracy | Training-log_loss | Validat
ion-accuracy | Validation-log_loss |

+-----------+--------------+-------------------+-------------------+--------
------------+---------------------+

| 1         | 0.041002     | 0.873711          | 0.816677          | 0.86844
7           | 0.819736            |

| 2         | 0.071004     | 0.879867          | 0.654540          | 0.86894
2           | 0.660694            |

| 3         | 0.106006     | 0.876040          | 0.551717          | 0.87289
8           | 0.560675            |

| 4         | 0.141008     | 0.881364          | 0.481115          | 0.87339
3           | 0.493366            |

| 5         | 0.176010     | 0.881864          | 0.432191          | 0.87388
7           | 0.447579            |

| 6         | 0.212012     | 0.883694          | 0.398424          | 0.87388
7           | 0.416491            |

+-----------+--------------+-------------------+-------------------+--------
------------+---------------------+
```

In [44]:

```
boost_model_2 = graphlab.boosted_trees_classifier.create(train_data, target='label', \
                                                         features=['search_tfidf', 'result_ra
                                                         validation_set=validation_data)
```

```
Boosted trees classifier:
--------------------------------------------------------

Number of examples          : 6010

Number of classes           : 3

Number of feature columns   : 3

Number of unpacked features : 5007

+-----------+--------------+-------------------+-------------------+--------
------------+---------------------+

| Iteration | Elapsed Time | Training-accuracy | Training-log_loss | Validat
ion-accuracy | Validation-log_loss |

+-----------+--------------+-------------------+-------------------+--------
------------+---------------------+

| 1         | 0.040003     | 0.874043          | 0.832070          | 0.87289
8           | 0.833190            |

| 2         | 0.070004     | 0.874043          | 0.679494          | 0.87240
4           | 0.681089            |

| 3         | 0.100006     | 0.875874          | 0.582850          | 0.87487
6           | 0.585373            |

| 4         | 0.130008     | 0.875541          | 0.519184          | 0.87388
7           | 0.524795            |

| 5         | 0.163010     | 0.873711          | 0.476809          | 0.87289
8           | 0.483074            |

| 6         | 0.197012     | 0.873045          | 0.445507          | 0.87240
4           | 0.454434            |

+-----------+--------------+-------------------+-------------------+--------
------------+---------------------+
```

- 分类器测试

In [45]:

```
boost_model_1.evaluate(test_data)
```

Out[45]:

```
{'accuracy': 0.8785569105691057,
```

```
'auc': 0.746247017529306,
'confusion_matrix': Columns:
       target_label    str
       predicted_label str
       count   int

Rows: 7

Data:
+--------------+-----------------+-------+
| target_label | predicted_label | count |
+--------------+-----------------+-------+
|   Resource   |    Navigation   |   3   |
|  Navigation  |    Navigation   |   55  |
|  Navigation  |   Information   |  166  |
| Information  |    Navigation   |   26  |
|   Resource   |   Information   |   44  |
| Information  |   Information   |  1673 |
|   Resource   |     Resource    |   1   |
+--------------+-----------------+-------+
[7 rows x 3 columns],
'f1_score': 0.44519569459256186,
'log_loss': 0.36782414625815874,
'precision': 0.8477459137310438,
'recall': 0.41813299737727605,
'roc_curve': Columns:
       threshold       float
       fpr     float
       tpr     float
       p       int
       n       int
       class   int

Rows: 300003

Data:
+-----------+-----+-----+------+-----+-------+
| threshold | fpr | tpr |  p   |  n  | class |
+-----------+-----+-----+------+-----+-------+
|    0.0    | 1.0 | 1.0 | 1699 | 269 |   0   |
|   1e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
|   2e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
|   3e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
|   4e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
|   5e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
|   6e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
|   7e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
|   8e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
|   9e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
+-----------+-----+-----+------+-----+-------+
[300003 rows x 6 columns]
Note: Only the head of the SFrame is printed.
You can use print_rows(num_rows=m, num_columns=n) to print more rows and co
lumns.}
```

In [46]:

```
boost_model_2.evaluate(test_data)
```

```
boost_model_2.evaluate(test_data)
```

Out[46]:

```
{'accuracy': 0.8683943089430894,
 'auc': 0.7198899577124401,
 'confusion_matrix': Columns:
        target_label    str
        predicted_label str
        count   int

 Rows: 5

 Data:
 +--------------+-----------------+-------+
 | target_label | predicted_label | count |
 +--------------+-----------------+-------+
 |  Navigation  |    Navigation   |   19  |
 | Information  |    Navigation   |   9   |
 |  Navigation  |   Information   |  202  |
 |   Resource   |   Information   |   48  |
 | Information  |   Information   |  1690 |
 +--------------+-----------------+-------+
 [5 rows x 3 columns],
 'f1_score': 0.36047901416051675,
 'log_loss': 0.41459987487687805,
 'precision': 0.7748527245949927,
 'recall': 0.3602252056706234,
 'roc_curve': Columns:
        threshold       float
        fpr     float
        tpr     float
        p       int
        n       int
        class   int

 Rows: 300003

 Data:
 +-----------+-----+-----+------+-----+-------+
 | threshold | fpr | tpr |  p   |  n  | class |
 +-----------+-----+-----+------+-----+-------+
 |    0.0    | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   1e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   2e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   3e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   4e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   5e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   6e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   7e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   8e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 |   9e-05   | 1.0 | 1.0 | 1699 | 269 |   0   |
 +-----------+-----+-----+------+-----+-------+
 [300003 rows x 6 columns]
 Note: Only the head of the SFrame is printed.
 You can use print_rows(num_rows=m, num_columns=n) to print more rows and co
lumns.}
```

3.4.4 比较不同分类器和不同特征提取对查询意图识别的影响

呃，其实这部分还有待改进。毕竟虽然有影响，但是影响比较小。

- 不同分类器对查询意图识别会有**1**个百分点的影响；
- 不同特征提取对意图识别的影响仅仅有**0.1**个百分点。

4. 结论

由于论文没有完成，所以结论部分还有待分析。

- 在答辩前，要深入学习**LTP**工具，优化特征提取；
- 同时要调整分类器的参数，优化分类结果。

欢迎大家批评指正，谢谢！