

该项目为微博爬虫以及数据分析说明文档

一共为4个代码组成:

名称	修改日期	类型	大小
pycache	2023/11/20 11:16	文件夹	
data	2023/11/20 21:06	文件夹	
img	2023/11/20 21:20	文件夹	
chromedriver.exe	2023/10/30 11:38	应用程序	16,468 KB
data_analysis.py	2023/11/20 21:17	JetBrains PyChar...	8 KB
spider_Data.py	2023/11/20 12:28	JetBrains PyChar...	4 KB
spider_URL.py	2023/11/20 14:29	JetBrains PyChar...	2 KB
train_model.py	2023/11/20 21:19	JetBrains PyChar...	8 KB
说明文档.md	2023/10/7 14:33	Markdown File	8 KB

先来说爬虫部分

爬虫部分分为spider_URL和spider_Data

spider_URL就是去爬取URL的，就是通过输入相关的热词，然后去爬取实时的用户对应的url

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	URL																									
2	https://weibo.com/1398165410?refer_flag=1001030103																									
3	https://weibo.com/5476485501?refer_flag=1001030103																									
4	https://weibo.com/5476485501?refer_flag=1001030103																									
5	https://weibo.com/5476485501?refer_flag=1001030103																									
6	https://weibo.com/5167198927?refer_flag=1001030103																									
7	https://weibo.com/748426079?refer_flag=1001030103																									
8	https://weibo.com/1765941144?refer_flag=1001030103																									
9	https://weibo.com/7330409525?refer_flag=1001030103																									
10	https://weibo.com/2403484123?refer_flag=1001030103																									
11	https://weibo.com/2548236520?refer_flag=1001030103																									
12	https://weibo.com/2739646084?refer_flag=1001030103																									
13	https://weibo.com/5864695058?refer_flag=1001030103																									
14	https://weibo.com/7785454944?refer_flag=1001030103																									
15	https://weibo.com/u/1268207667																									
16	https://weibo.com/1458940234?refer_flag=1001030103																									
17	https://weibo.com/2407994580?refer_flag=1001030103																									
18	https://weibo.com/6051932290?refer_flag=1001030103																									
19	https://weibo.com/2697176354?refer_flag=1001030103																									
20	https://weibo.com/7734466995?refer_flag=1001030103																									
21	https://weibo.com/6863146289?refer_flag=1001030103																									
22	https://weibo.com/7018345375?refer_flag=1001030103																									
23	https://weibo.com/7018345377?refer_flag=1001030103																									
24	https://weibo.com/680825761?refer_flag=1001030103																									
25	https://weibo.com/6591637204?refer_flag=1001030103																									
26	https://weibo.com/7458410148?refer_flag=1001030103																									
27	https://weibo.com/6591638695?refer_flag=1001030103																									
28	https://weibo.com/6591239902?refer_flag=1001030103																									
29	https://weibo.com/6647780511?refer_flag=1001030103																									
30	https://weibo.com/2632969742?refer_flag=1001030103																									
31	https://weibo.com/7458410148?refer_flag=1001030103																									
32	https://weibo.com/773450841?refer_flag=1001030103																									
33	https://weibo.com/6334465793?refer_flag=1001030103																									
34	https://weibo.com/7732379156?refer_flag=1001030103																									
35	https://weibo.com/7732313926?refer_flag=1001030103																									
36	https://weibo.com/7731652494?refer_flag=1001030103																									
37	https://weibo.com/1678844014?refer_flag=1001030103																									
38	https://weibo.com/7732313786?refer_flag=1001030103																									
39	https://weibo.com/7734147546?refer_flag=1001030103																									
40	https://weibo.com/7735369792?refer_flag=1001030103																									
41	https://weibo.com/6251409056?refer_flag=1001030103																									
42	https://weibo.com/7730874860?refer_flag=1001030103																									

接着通过spider_Data去模拟登陆然后去爬取相关的用户信息数据

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	URL	昵称	简介	ip属地	粉丝量	关注量	博文数																			
2	https://we/Felimo	倒置家	IP属地: 河	59	299	全部微博 (111)																				
3	https://we/h我叫崔小广	广告 附	IP属地: 江	349	1124	全部微博 (1120)																				
4	https://we/不喝25斤江如意	IP属地: 河	274	80	全部微博 (90)																					
5	https://we/右手甩	want to fo	IP属地: 江	40	144	全部微博 (17)																				
6	https://we/李忠Geek	自媒体安利	IP属地: 安	1万	455	全部微博 (306)																				
7	https://we/安静的雨E	暂无简介	IP属地: 广	10	127	全部微博 (2407)																				
8	https://we/南岛心空	喜欢一切	IP属地: 上	938	1235	全部微博 (8072)																				
9	https://we/春情不抵柔情你过	分	IP属地: 上	105	971	全部微博 (7998)																				
10	https://we/不知道算/什么时候	-	IP属地: 上	158	936	全部微博 (6568)																				
11	https://we/篇上风也/如果哪一天	IP属地: 上	47	664	全部微博 (3898)																					
12	https://we/睡一个月多今晚月色	IP属地: 上	35	635	全部微博 (3632)																					
13	https://we/永远二十岁永远保持	IP属地: 上	31	624	全部微博 (3719)																					
14	https://we/昨日梦失书今晚月色	IP属地: 上	35	658	全部微博 (3682)																					
15	https://we/短发超暴	短发超暴	IP属地: 上	2356	408	全部微博 (1592)																				
16	https://we/前阵头像我发的头像	IP属地: 上	33	385	全部微博 (1370)																					
17	https://we/很酷也是	暂无简介	IP属地: 广	4	88	全部微博 (2191)																				
18	https://we/柠檬树上柠檬无简介	IP属地: 广	6	97	全部微博 (2522)																					
19	https://we/向往自由由	IP属地: 广	6	69	全部微博 (2349)																					
20	https://we/应大嘴下	暂无简介	IP属地: 广	7	86	全部微博 (2564)																				
21	https://we/想法或露	暂无简介	IP属地: 广	3	80	全部微博 (2520)																				
22	https://we/爱带带着	小	IP属地: 广	4	85	全部微博 (2746)																				
23	https://we/夜晚的喝	暂无简介	IP属地: 广	9	103	全部微博 (2597)																				
24	https://we/九三患者	颜值患者	IP属地: 上	108	700	全部微博 (4006)																				
25	https://we/抽后自带	暂无简介	IP属地: 广	7	91	全部微博 (2854)																				
26	https://we/发生了什	如果我一	IP属地: 上	144	550	全部微博 (3387)																				
27	https://we/用超像	暂无简介	IP属地: 广	7	92	全部微博 (2646)																				
28	https://we/记得日语	如果事与	IP属地: 上	41	463	全部微博 (2939)																				
29	https://we/送你小人	羊	IP属地: 广	7	91	全部微博 (2697)																				
30	https://we/你失眠夜	脾气暴躁	IP属地: 上	50	465	全部微博 (2901)																				
31	https://we/珠地	吃	IP属地: 广	3	97	全部微博 (2411)																				
32	https://we/暖灯下	的羊	IP属地: 广	9	97	全部微博 (2699)																				
33	https://we/枕边熊	关于小熊	IP属地: 福	473	909	全部微博 (1612)																				
34	https://we/吃不停的	小	IP属地: 广	10	59	全部微博 (39)																				
35	https://we/看舞	知识	IP属地: 广	10	60	全部微博 (81)																				
36	https://we/可东	爱加维	作品	中	IP属地: 广	2	61	全部微博 (48)																		
37	https://we/那我也是	暂无简介	IP属地: 浙	86	167	全部微博 (361)																				
38	https://we/脑红	的恩是	内心丰	IP属地: 山	471	389	全部微博 (3727)																			
39	https://we/我跑	路上	羊	IP属地: 广	6	85	全部微博 (2759)																			
40	https://we/重庆	张	无简介	IP属地: 广	5	89	全部微博 (2702)																			
41	https://we/回头再遇	无简介	IP属地: 广	7	85	全部微博 (2673)																				
42	https://we/王瑜	哈哈	记录小王	IP属地: 浙	52	22	全部微博 (68)																			

接着我们爬取相关信息之后，我们开始数据清洗工作以及数据建模工作

也就是train_model，我们先把相关的数据特征收集下来的，无用的则删除掉，这里根据提供的数据集进行训练

接着我们再去数据清洗工作，把数字的归为数字，文字的的进行筛选，去掉无用词以及机械压缩

```
#ip处理
def ip_chuli1(x):
    x1 = str(x).split(": ")
    return x1[-1]

#ip处理
def ip_chuli2(x):
    provinces = ['安徽', '澳门', '北京', '重庆', '福建', '甘肃', '广东', '广西', '贵州', '海南', '河北', '黑龙江', '河南', '湖北', '湖南', '江苏', '江西', '吉林', '辽宁', '内蒙古', '宁夏', '青海', '山东', '上海', '山西', '陕西', '四川', '台湾', '天津', '西藏', '香港', '新疆', '云南', '浙江']

    if x in provinces:
        #表示属于国内
        return 1
    else:
        #表示属于外国
        return 0

def jianjie_processing(x):
    x1 = str(x)
    if x1 == '暂无简介':
        #表示低活跃用户
        return 0
    else:
        #表示高活跃用户
        return 1

#数字处理
def number_data(x):
    try:
        x1 = str(x)
        number = re.findall(r'\d+', x1)
```

```

        return number[0]
    except:
        return 0

#粉丝处理
def fan_chuli(x):
    try:
        x1 = int(x)
        return x1
    except:
        if '万' in x:
            x1 = str(x).replace('万', '')
            x1 = float(x1) * 10000
            return x1

#停用词函数
stop_words = []
with open("./data/stopwords_cn.txt", 'r', encoding='utf-8') as f:
    lines = f.readlines()
    for line in lines:
        stop_words.append(line.strip())

#去掉标点符号，以及机械压缩
def preprocess_word(word):
    word1 = str(word)
    word1 = re.sub(r'转发微博', '', word1)
    word1 = re.sub(r'#\w+#', '', word1)
    word1 = re.sub(r'【.*?】', '', word1)
    word1 = re.sub(r'@[\w]+', '', word1)
    word1 = re.sub(r'[a-zA-Z]', '', word1)
    word1 = re.sub(r'\.\d+', '', word1)
    return word1

def emjio_tihuan(x):
    x1 = str(x)
    x2 = re.sub('([\.\*\?\])', '', x1)
    x3 = re.sub(r'@[\w\u2E80-\u9FFF]+:?\|[\w+\\]', '', x2)
    x4 = re.sub(r'\n', '', x3)
    return x4

def is_all_chinese(strs):
    for _char in strs:
        if not '\u4e00' <= _char <= '\u9fa5':
            return False
    return True

# 定义机械压缩函数
def yasuo(st):
    for i in range(1, int(len(st) / 2) + 1):
        for j in range(len(st)):
            if st[j:j + i] == st[j + i:j + 2 * i]:
                k = j + i

```

```

        while st[k:k + i] == st[k + i:k + 2 * i] and k < len(st):
            k = k + i
        st = st[:j] + st[k:]

    return st

def get_cut_words(content_series):
    # 读入停用词表
    # 分词
    word_num = jieba.lcut(content_series, cut_all=False)

    # 条件筛选
    word_num_selected = [i for i in word_num if i not in stop_words and len(i) >=
2 and is_all_chinese(i) == True]

    return ' '.join(word_num_selected)

def null_paichu(x):
    x1 = str(x)
    if len(x1) != 0:
        return x1
    else:
        return np.NaN

data = pd.read_csv('./data/data.csv')
data['ip属地'] = data['ip属地'].apply(ip_chuli1)
data['属地分类'] = data['ip属地'].apply(ip_chuli2)
data['简介分类'] = data['简介'].apply(jianjie_processing)
data['博文数'] = data['博文数'].apply(number_data)
data['粉丝量'] = data['粉丝量'].apply(fan_chuli)
data['关注量'] = data['关注量'].apply(fan_chuli)
data['关注量'] = data['关注量'].astype('int')
data['粉丝量'] = data['粉丝量'].astype('int')
data['博文数'] = data['博文数'].astype('int')
data['属地分类'] = data['属地分类'].astype('int')
data['简介分类'] = data['简介分类'].astype('int')

```

我们对处理好的数据集进行规范化处理，也就是z-score

把这些数据进行相加，生成影响力，根据影响力来进行判断用户的类别

```

data1 = data[['属地分类', '简介分类', '粉丝量', '关注量', '博文数']]

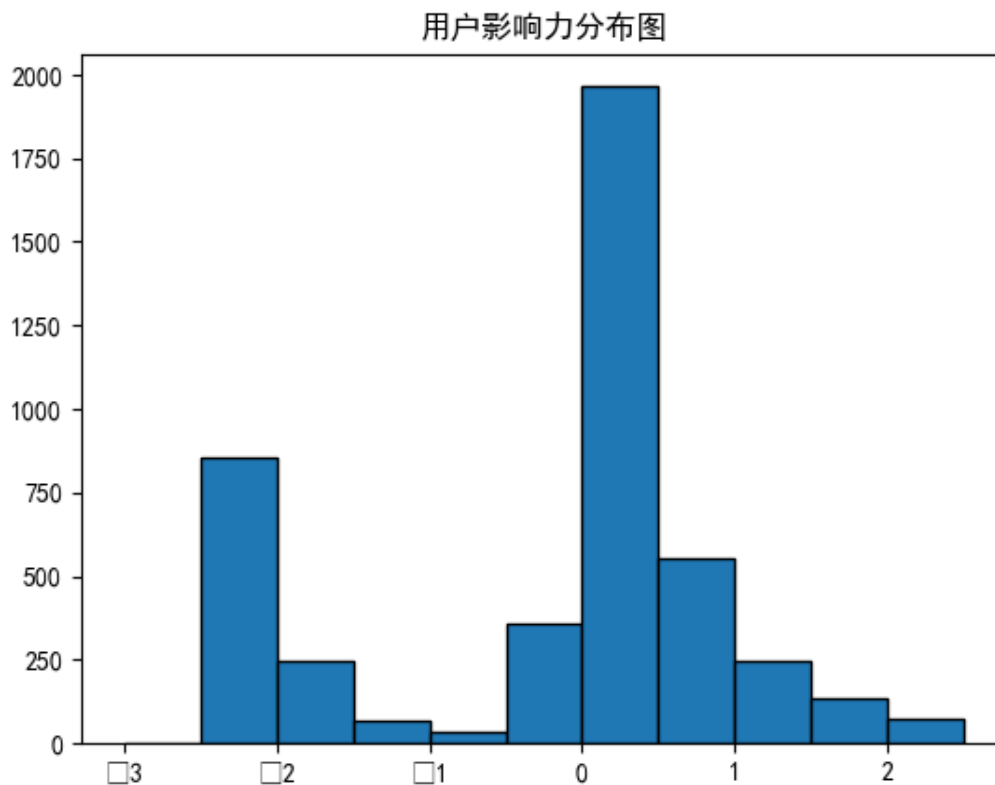
#将数据进行[0,1]规范化
scaled_x = preprocessing.scale(data1)

list_scaler = []
for m in scaled_x:
    number1 = m[0] + m[1] + m[2] + m[3] + m[4]
    list_scaler.append(number1)

data['影响力'] = list_scaler
plt.rcParams['font.sans-serif'] = ['SimHei']

```

```
plt.hist(list(data['影响力']), bins=np.arange(-3, 3, 0.5), edgecolor="black")
plt.title('用户影响力分布图')
plt.savefig('./img/用户影响力分布图.png')
plt.show()
```



根据该图，我们分四个类别，分类规则如下：

在影响力小于-2的时候，归为机器用户，
当影响力大于-2小于0的时候，归为低活跃用户
当影响力大于0小于1的时候，归为普通用户
当影响力大于1的时候，归为高活跃用户

代码如下：

```
def class_type(x):
    x1 = float(x)
    if x1 <= -2:
        return 0
    elif -2 < x1 <= 0:
        return 1
    elif 0 < x1 <= 1:
        return 2
    else:
        return 3
```

```
data['用户分类'] = data['影响力'].apply(class_type)
```

接着 我们做好用户分类之后，我们来做随机森林分类学习任务，代码如下：

```
# 分割训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(data1, list(data['用户分类']),
test_size=0.4, random_state=42)

# 利用网格搜索进行超参数调优
param_grid = {'n_estimators': [50, 100, 200],
              'max_features': ['sqrt', 'log2'],
              'max_depth' : [4, 5, 6, 7, 8],
              'criterion' :['gini', 'entropy']}

rfc = RandomForestClassifier(random_state=1)
CV_rfc = GridSearchCV(estimator=rfc, param_grid=param_grid, cv = 5)
CV_rfc.fit(X_train, y_train)

# 输出最佳参数
print(CV_rfc.best_params_)

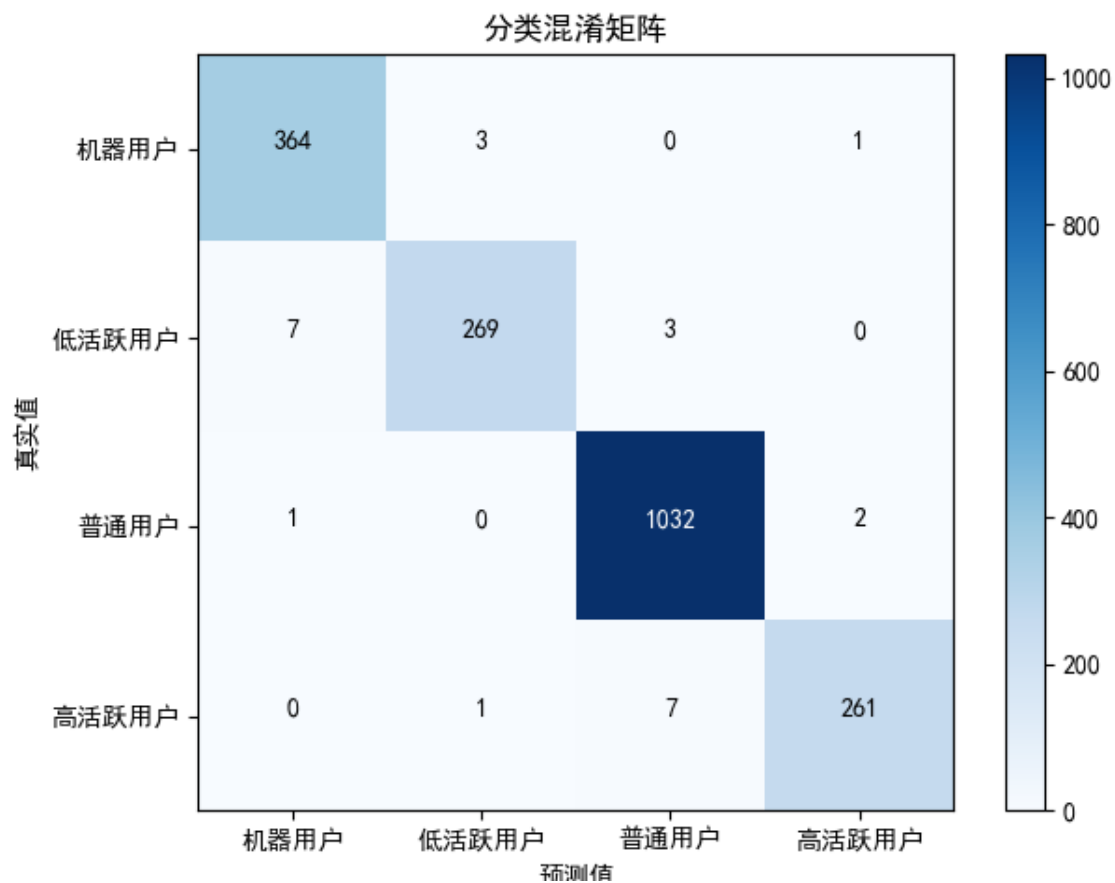
# 使用最佳参数重建模型
rfc1=RandomForestClassifier(random_state=1, n_estimators=100,
max_features='sqrt', max_depth=8, criterion='entropy')
rfc1.fit(X_train, y_train)

# 进行预测
pred = rfc1.predict(X_test)

# 输出模型评估结果
print('Accuracy score: ', accuracy_score(y_test,pred))
print('Confusion Matrix: \n', confusion_matrix(y_test,pred))
```

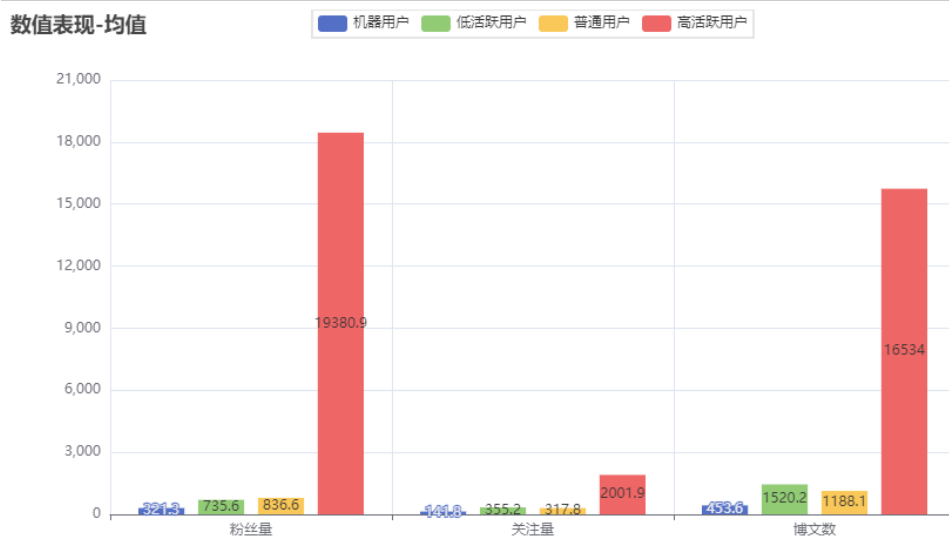
结果呈现：

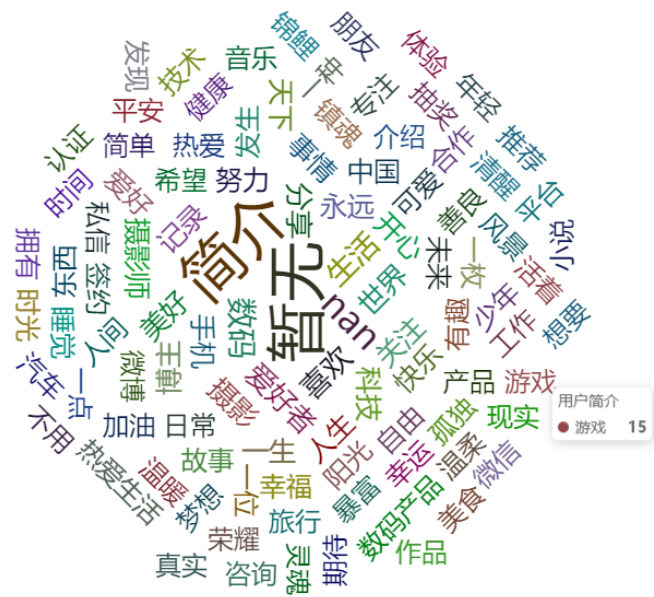
	precision	recall	f1-score	support
0	0.98	0.99	0.98	368
1	0.99	0.96	0.97	279
2	0.99	1.00	0.99	1035
3	0.99	0.97	0.98	269
accuracy			0.99	1951
macro avg	0.99	0.98	0.98	1951
weighted avg	0.99	0.99	0.99	1951



最后我们依次来介绍这些可视化的含义：

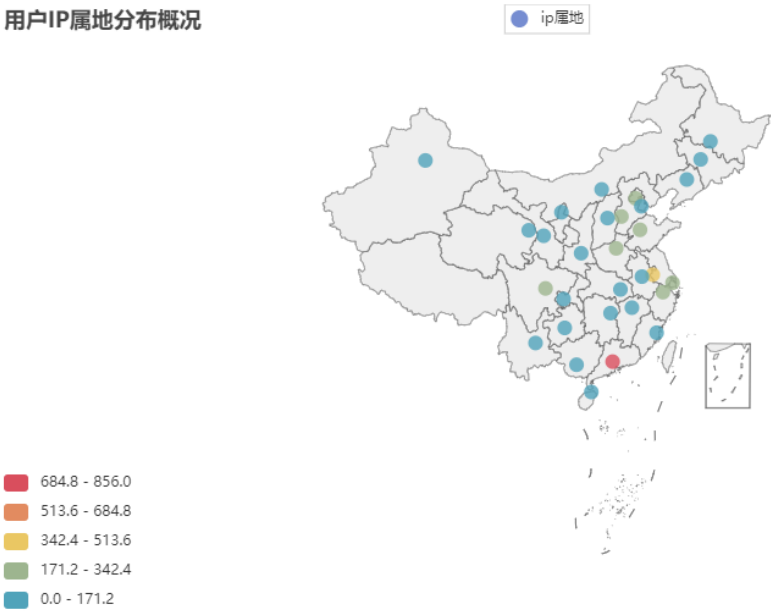
该图是代表不同用户，他们的粉丝量、关注量、博文量的平均数，是统计该类型的全部数值，然后求的平均数





这个是基于用户简介里面做的词云图，其目的是看看用户平时主要用哪些词语来组成简历，显而易见，大多数用户都是暂无简介的

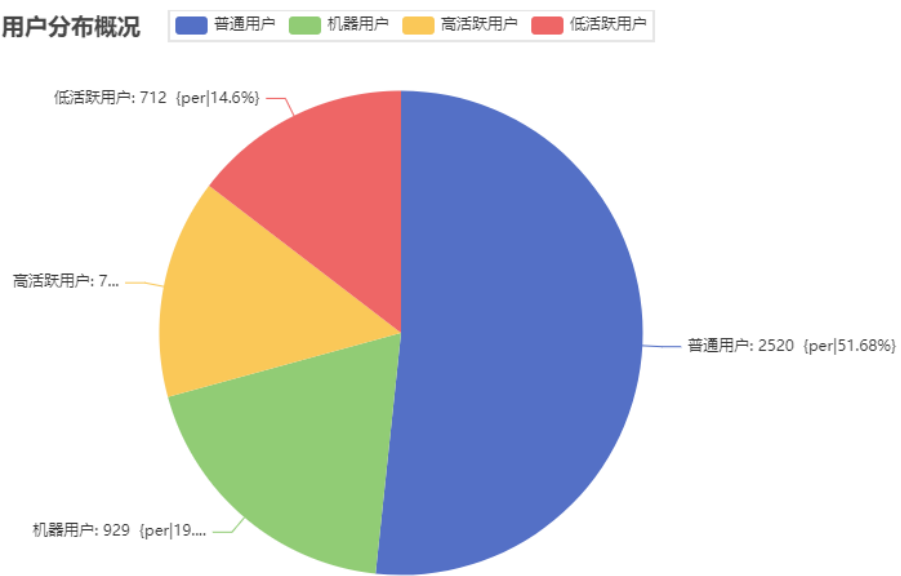
用户IP属地分布概况



中国地图，这里是不同用户，他们归属地的分布数量，地图，根据左下角的颜色，可以区分数量的大小，数量越大，则对应的颜色也会发生变化

饼图的话，就是不同用户的一个占比状况

用户分布概况



以上便是整个项目的解析，如果有什么不懂的，可以在群里讨论