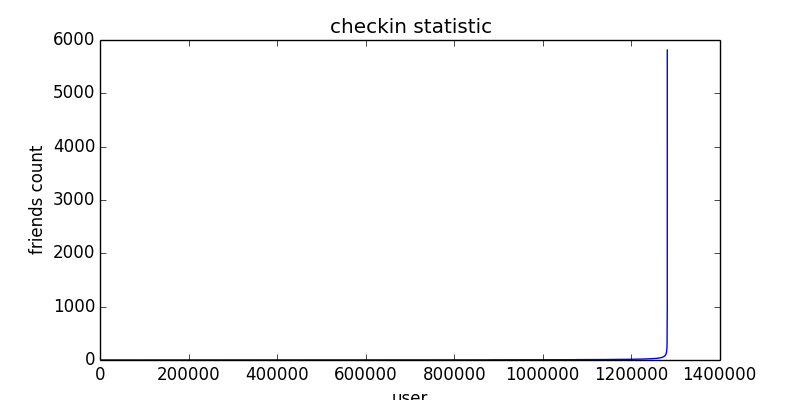
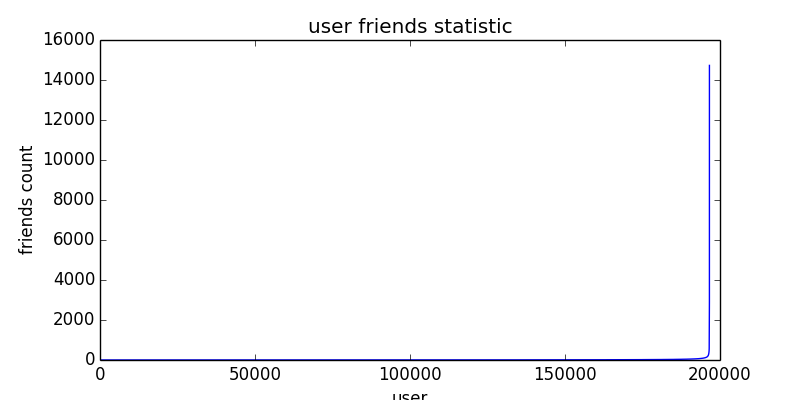
任务三文档

# 预处理

在预处理的时候分析edges.txt和checkin.txt两个文件的好友数量和地点登陆数量，统计一个用户拥有的好友数量（下左图）和一个地点被登录的次数（下右图）的分布如下：



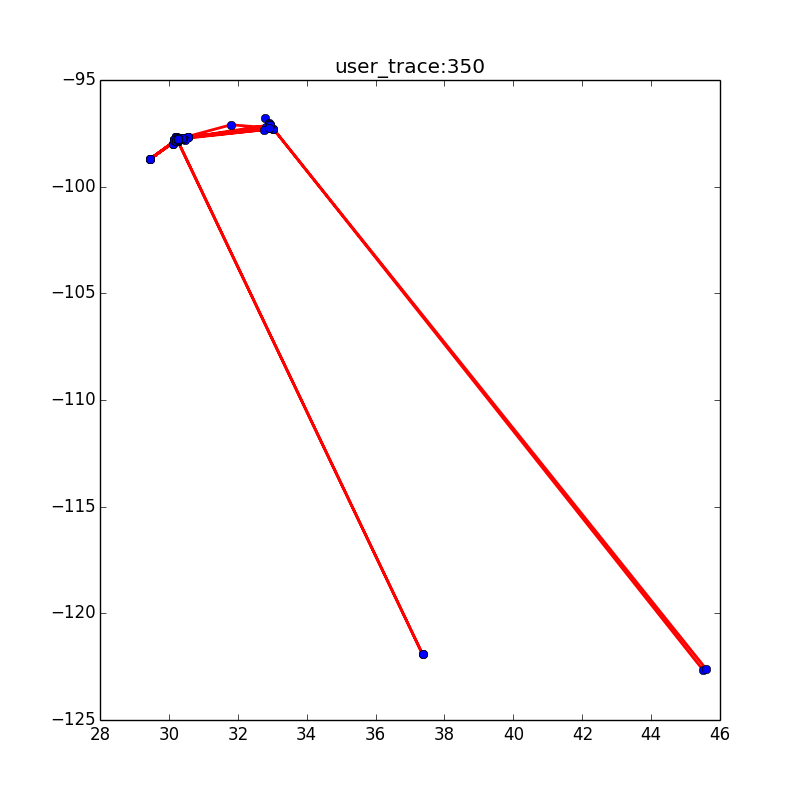
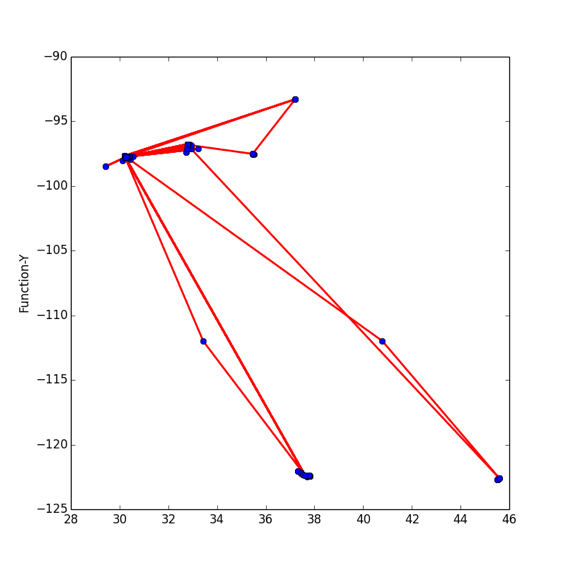
可以得到初步的结论：极少数的人拥有极多的好友，大部分人的好友数量非常少。同样，极少数地点被登录了非常多次，大部分地点登陆的非常少。

观察以上数据，过滤掉过小的一部分值。Location设为100，edges(user)设为50，凡是少于以上条件的user或location都会被过滤掉。过滤后的结果大约使得数据减少到十分之一。然后对edges删减，对于不存在于过滤后的checkin中的user，删除好友关系，则edges减少到24w（原190w）

# 时序关系推荐

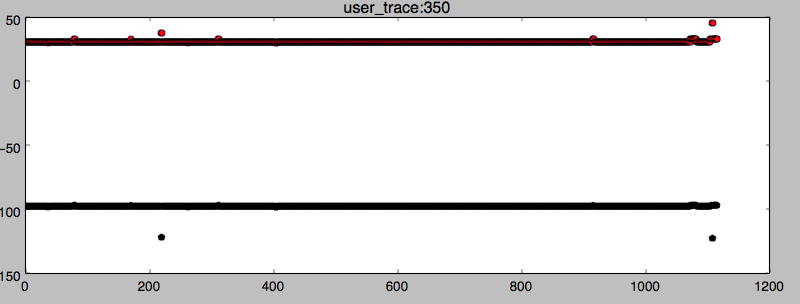
本实验基于对经纬度的聚类。

原始数据按照用户-时间两个维度排序，这是非常好的性能，我们在预处理的时候仍然保留了这一特性。抽取一些用户的轨迹特征如下：



可以惊喜的看到，用户的空间关联度非常好，上述图片实际上是由500~700个点构成的，但是直观的可以看出，该用户只在少数的几个地方活动。

下图是单个用户所处的经纬度按照时序绘点，符合上图，用户绝大多数时间都在一个位置。



，由于如此的聚集，用户的运动在绝大多数情况下，直接以上次签到的地方推荐给他。更优化的考虑，取前n个点判断所述区域，然后服从多数原则推荐。

算法如下：

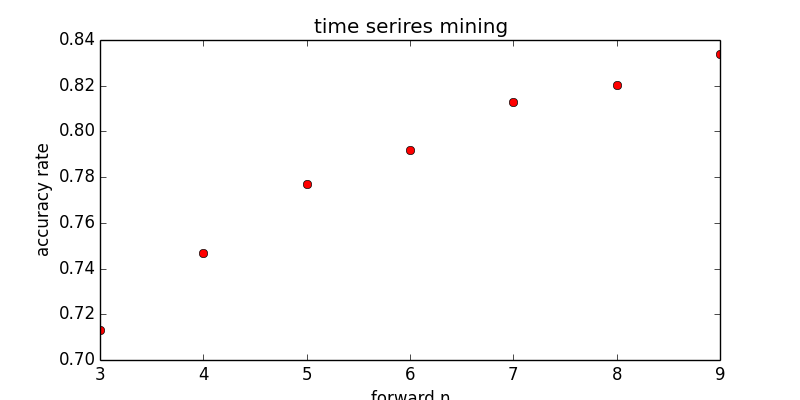
算法1：基于用户过去时序关系的聚类推荐

输入：用户在t时间之前的时间与到达地点序列；location的聚类结果

输出：t时刻可能的地点预测。

1. 选取t之前的n个时间-地点对。
2. 分析n个点各自所述的类别，以多数通过原则一个最可能的类别，作为下一次的预测。
3. 如果t+1超过时序数据长度，跳出。
4. 删掉t-n点，增加t+1点的信息。
5. 重复步骤2.

由于输出的仅仅是所属类别，粒度较粗，精度较好，这是由user运动规律决定的。同时随着向前的窗口区域增大，准确率提高。



# 基于好友动态的预测

可以看到由于运动分布，我们可以较好的分析出所属类别，这是一个粗粒度的，而具体到该大类下的location\_id，我们尝试用好友的累计经验矩阵做一下分析。假设用户在t时刻去过loc1。过往历史上如果loc1向loc2出现的次数最多，则认为下一次用户最可能去loc2。

我们为用户自己的历史路径分配了较重的权重，他的好友较轻。

算法如下：

算法2：基于用户好友的过去状态转移的精确推荐

输入：用户u及其好友在t时间之前的时间与到达地点序列；location的聚类结果

输出：t时刻之前的单点转移预测best\_point和类别转移预测best\_cluster。

1. u的权重weight设为2，其他好友为1
2. 初始化所有地点之间的转移数为0，即初始化一个zero（n）的转移数量矩阵matrix。
3. 对包括u和他的好友在内的所有人在t之前的活动记录进行如下操作：
4. 如果ti时刻在loc1，ti+1时刻在loc2，则matrix[loc1][loc2] += weight
5. 更新单点预测列表：

best\_point[loc1]=max(matrix[loc1][loc2] ,best\_point[loc1])

1. 更新大类预测列表，其中cl1是loc1所属的聚类：

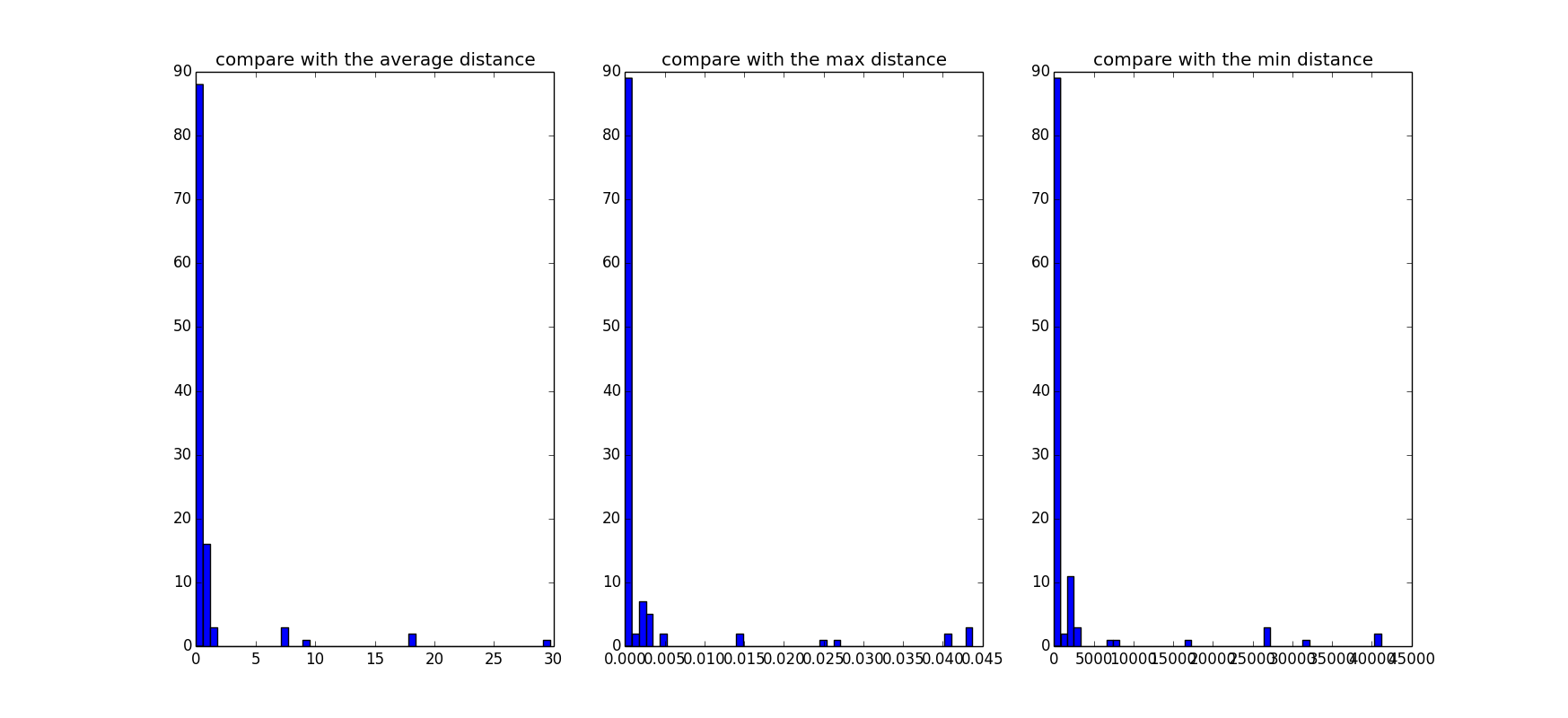
best\_cluster[cl1]=max(matrix[loc1][loc2] ,best\_cluster[cl1])

1. 输出best\_point和best\_cluster

开始对t进行预测时，如果该用户在t-1时刻所处位置loc在单点转移矩阵中有记录，则将该单点矩阵中出现最多的位置作为预测；如果该点从没有出现过（则单点转移矩阵中没有记录），则将该点所属的类的转移矩阵结果作为预测（即该大类最可能的动向）。如果整个类都没有记录，则根据马尔可夫性，直接将t-1时刻他的位置作为预测结果。

对于每一个类，计算平均距离原则上要用O()，为了节省时间，我们随机抽样100个点计算平均距离。同时计算每个类的最大边界距离。

在评估准确度时，我们采用了三种方式。我们计算了预测值与真实值之间的距离偏差，分别同1.该大类的平均距离；2.该大类的最小距离；3.该大类的最大距离进行比较，作直方图分布如下：



可以看到，绝大多数的预测结果基本都与真实距离只差一个点左右的距离，而远小于该类的平均距离和最大距离，这样的预测可以接受。

# 代码说明：

preprocessing.py包含预处理、数据删除的操作函数；

distribution\_analyze.py包含edges和checkin的数量分布分析和画图；

analyzing.py包含一些距离计算、画图、文件读取与数据处理的函数

time\_series\_mining.py包含时序关系推荐和好友历史推荐的主要函数，分别在time\_series\_predict\_cluster\_hist()和friend\_transfer\_hist()中，用户可以通过释放该文件末尾的两行注释来获取分析结果