步骤 4. 输出波尔图出租车 GPS 测试集中每段旅程对应的目的地, 输出的数据是 WGS84 坐标.

### 3.5 参数选择和评估

选择通过网格搜索进行优化的参数有:

1) Nr, 储备池规模大小;

2) rho 或 sigma, 光谱半径或最大奇异值;

3) $\alpha$, 泄漏率;

4) Lambda, 岭回归正则化参数;

5) Conn, 连接因子, 默认情况下为 100%.

参数的每个组合定义了一个模型, 而模型又在验证集上进行了评估. 可以通过交叉验证获得更准确的评估, 并在所有折叠上使用具有最小平均验证误差的模型.

另一个限制是随机发生器种子是固定的; 应该从不同的种子开始进行全网搜索, 从而可以生成不同的网络权重. 通过多次实验进行比较分析得到网络搜索的最佳参数情况如表 1.

表 1 网络搜索的最佳参数

| 参数 | Nr | sigma | Lambda | $\alpha$ | Conn(%) |
|---|---|---|---|---|---|
| 最佳值 | 250 | 0.4 | 0.01 | 1 | 30 |

## 4 基于随机森林算法预测抵达时间

### 4.1 预测流程

经过预处理的波尔图 GPS 出租车数据集在算法处理器中完成出租车目的的预测. 预测流程主要包括两个阶段: 训练数据集抽样预处理、测试数据集.

### 4.2 评估指标

对于行程时间, 使用均方根误差 (RMSLE) 评估预测, 定义如下:

$$RMSLE=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\ln(p_i+1)-\ln(a_i+1))^2} \qquad (11)$$

这里的$n$是测试数据集的总观测值, $p_i$是观测值, $a_i$是旅行时间的实际值, ln是自然对数.

### 4.3 算法描述

随机森林中的每一棵分类树为二叉树, 其生成遵循自顶向下的递归分裂原则, 即从根节点开始依次对训练集进行划分. 在二叉树中, 根节点包含全部训练数据, 按照节点纯度最小原则, 分裂为左节点和右节点, 它们分别包含训练数据的一个子集, 按照同样的规则节点继续分裂, 直到满足分支停止规则而停止生长. 若节点$n$上的分类数据全部来自于同一类别, 则此节点的纯度$I(n)=0$, 纯度度量方法是 Gini 准则, 即假设$P(X_j)$是节点$n$上属于$X_j$类样本个数占训练.

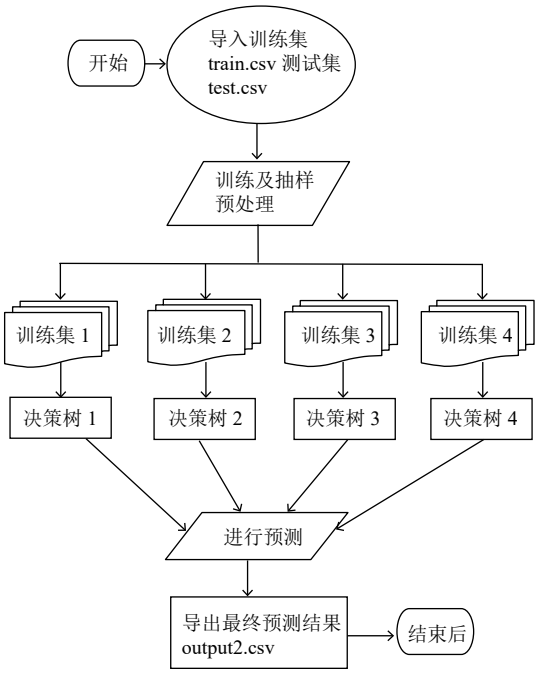


图 6 随机森林算法预测抵达时间流程图

具体算法实现过程如下:

步骤 1. 原始训练集为$N$, 应用自助法 (bootstrap) 有放回地随机抽取$k$个新的自助样本集, 并由此构建$k$棵分类树, 每次未被抽到的样本组成了$k$个袋外数据.

步骤 2. 设有$m_{\text{all}}$个变量, 则在每一棵树的每个节点处随机抽取$m_{\text{try}}$个变量 ($m_{\text{try}}$ $n$ $m_{\text{all}}$), 然后在$m_{\text{try}}$中选择一个最具有分类能力的变量, 变量分类的阈值通过检查每一个分类点确定.

步骤 3. 每棵树最大限度地生长, 不做任何修剪.

步骤 4. 将生成的多棵分类树组成随机森林, 用随机森林分类器对新的数据进行判别与分类, 分类结果按树分类器的投票多少而定.

### 4.4 抽取特征预测抵达时间

用于时间预测的一组特征与目的地预测的特征集非常相似, 差异在于将最近旅行的抵达时间视为目标变量而不是目的地. 为此提取的特征如下:

a) 旅行时间和 10 个最邻近的 Haversine 距离.

b) 内核回归特征.

除了前面描述的从部分观察到的行程中提取的所有特征之外, 我们还考虑了直接从观察到的不完全行

程中提取的以下附加的时间预测特征 (即仍在进行的行程):

a) 在迄今为止观察到的部分轨迹的最后$d$米和整个不完全行程上计算的平均速度, 其中$d \in \{10,20,50,100,200\}$. 这些功能在进行预测时传达最新的交通状况.

b) 到目前为止观察到的不完整旅行的最后$d$米的平均加速度, 其中$d \in \{10,20,50,100,200\}$.

c) 形状复杂度: (欧几里德) 行进距离与第一个和最后一个 GPS 位置之间的 Haversine 距离之间的比率. 具有更高复杂性的旅行 (例如“z - zag 之旅”(zig- zag trips) 往往是出租车司机在城市周围开车寻找乘客的行程. z-zag 的旅行时间往往较长, 所以事先确定这些行程是合理的.

d) 通过计算任何一对连续 GPS 更新之间的速度来识别 GPS 踪迹中的缺失值. 如果估计的速度超过速度限制ŷkm / h, 即使在部分观察到的行程中只有一对连续的 GPS 更新, 该行程被标记为缺少 GPS 更新的行程. 我们使用速度限制$\hat{y} \in \{100,120,140,160\}$ km/h, 缺少值的旅行往往有更长的旅程时间.

总的来说, 得出 66 个特征来预测出租车旅程的抵达时间[8].

## 5 检测结果分析

本系统检测算法在 Sklearn 开源库处理平台上编写, 操作系统为 Windows10, 服务器 CPU 配置为 Intel Core i5-5200U 2.2 GHz, 每台节点为 8 GB 内存. 还有一些核心包包括: Numpy, Scipy, Pandas, Matplotlib.

### 5.1 出租车目的地预测实验结果

出租车数据集包含 1 710 670 次旅行, 从 01/07/2013 到 24/06/2014, 其中一些是空的或缺失值, 我们在预处理时去除空轨迹, 但一些缺失值不会影响实验结果.

如图 7 是数据集中的 200 016 条训练集的终点.

图 7 部分训练集终点

对于该实验, 只使用每个行程的轨迹折线, 丢弃其他特征和空轨迹. 折线在 0-1 与最小 - 最大归一化之间进行归一化; 也可以使用 Z 分数归一化. 这样可以限制数据集的范围, 并且保证程序运行时收敛加快.

目标是轨迹的最后一点, 它被分配为每个点的目标. 因此, 网络被训练用来预测每个前缀轨迹的终点.

表 2 算法模型 MHD 值对比

| 算法模型 | MHD |
|---|---|
| 回声状态网络算法 (ESN) | 2.612 19 |
| 核回归算法 (KR)[9] | 2.952 36 |
| K 最近邻算法 (KNN)[9] | 2.975 27 |

表 2 为本算法模型的 MHD 值结果比较. 抵达目的地预测评价指标为平均 Haversine 距离 (MHD), 回声状态网络算法 (ESN) 计算结果为 2.612 19. 该值越小越好, 故 ESN 算法相对较好.

如表 3 是测试集中各段旅程目的地坐标预测的部分结果, 包括旅行 ID, 经纬度的坐标, 共有 327 条预测结果.

表 3 测试集中各旅程的目的地坐标

| TRIP_ID | LATITUDE | LONGITUDE |
|---|---|---|
| T1 | 41.152 76 | –8.588 52 |
| T2 | 41.170 07 | –8.608 96 |
| T3 | 41.172 13 | –8.589 54 |
| T4 | 41.147 92 | –8.611 06 |
| T5 | 41.149 01 | –8.616 17 |
| T6 | 41.178 07 | –8.631 67 |
| T7 | 41.159 28 | –8.590 06 |
| T8 | 41.186 95 | –8.601 18 |
| T9 | 41.132 16 | –8.597 57 |
| T10 | 41.202 05 | –8.609 02 |
| T11 | 41.173 77 | –8.601 04 |
| T12 | 41.155 35 | –8.600 55 |
| T13 | 41.163 47 | –8.594 59 |
| T14 | 41.235 02 | –8.678 71 |
| T15 | 41.149 93 | –8.608 34 |

### 5.2 出租车抵达时间预测实验结果

表 4 对本算法的抵达时间预测和 GBRT 和 ERT 进行了性能分析. 评价指标为 RMSLE, 可以发现随机森林算法计算结果为 0.416 74. 该值越小越好, 故 RF 算法相对较好.

表 5 是测试集中各段旅程抵达目的地所需花费时间的部分结果, 包括旅行 ID, 旅行时间, 共有 327 条预测结果. 总体表明: 提出的预测系统可以较好的完成出

租车的多项关键信息预测, 有较好的实用价值.

表 4 算法模型 *RMSLE* 值对比

| 算法模型 | *RMSLE* |
|---|---|
| 随机森林算法 | 0.416 74 |
| 迭代决策算法[10] | 0.419 85 |
| 极端随机数算法[11] | 0.416 76 |

表 5 测试集中各旅程抵达目的地的所需花费时间

| TRIP_ID | TRAVEL_TIME(s) |
|---|---|
| T1 | 908.2515 |
| T2 | 1020.258 |
| T3 | 823.8574 |
| T4 | 742.6127 |
| T5 | 582.9438 |
| T6 | 3180.974 |
| T7 | 810.4445 |
| T8 | 598.2083 |
| T9 | 1162.7 |
| T10 | 1552.454 |
| T11 | 1841.116 |
| T12 | 652.3195 |
| T13 | 536.0251 |
| T14 | 1493.95 |
| T15 | 1199.231 |

## 6 结语

出租车公司以及近年来兴起的一批打车平台在进行车辆的动态调度时, 都需要掌握每个车辆出行终点和抵达时间的信息. 如果车辆调度员能够知道出租车完成当前出行的终点和抵达目的地所需时间, 就可以为下一个乘车需求分配距离最近且时间点最契合的车辆. 尤其是在城市的中心地带, 出租车抵达的目的地附近往往就有新的乘车需求. 因此, 对车辆目的地和抵达时间的预测具有实际的应用价值和广泛的应用市场. 本文提出了基于机器学习的智能交通预测系统, 可大致预测出租车的终点和抵达时间. 不足之处在于实验过程中, 因为电脑设备的问题, 波尔图出租车 GPS 数据集实在是太大了, 只抽取了部分的训练集来训练, 所以测试集得到的目的地和抵达时间结果有可能不够精确. 但总体来说这两种算法还是较符合我们实验的要求, 整体上性能和效果也是挺不错的.

## 参考文献


1 Kusner M, Tyree S, Weinberger KQ, *et al*. Stochastic neighbor compression. In: Jebara T, Xing EP, eds. Proceedings of the 31st International Conference on Machine Learning. 2014. 622–630.

2 本文使用的波尔图出租车 GPS 数据集下载官方网址: https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i/rules

3 Lukosevicius M, Jaeger H. Reservoir computing approaches to recurrent neural network training. Computer Science Review, 2009, 3(3): 127–149. [doi: 10.1016/j.cosrev.2009.03.005]

4 Gallicchio C, Micheli A. Architectural and Markovian factors of echo state networks. Neural Networks, 2011, 24(5): 440–456. [doi: 10.1016/j.neunet.2011.02.002]

5 Breiman L. Random forests. Machine Learning, 2001, 45(1): 5–32. [doi: 10.1023/A:1010933404324]

6 Fulkerson B. Pattern recognition and neural networks. Technometrics, 2009, 39(2): 233–234. [doi: 10.1080/00401706.1997.10485099]

7 Lukoševičius M. A practical guide to applying echo state networks. In: Montavon G, Orr GB, Müller KR, eds. Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol 7700. Springer. Berlin, Heidelberg. 2012. 86–124. [doi: 10.1007/978-3-642-35289-8_36]

8 Tiesyte D, Jensen C. Similarity-based prediction of travel times for vehicles traveling on known routes. Annals of GIS, 2008, 14(1): 1–10. [doi: 10.1080/10824000809480633]

9 Lam HT, Bouillet E. Flexible sliding windows for kernel regression based bus arrival time prediction. Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2015. Lecture Notes in Computer Science. Springer. Cham. 2015. 68–84. [doi: 10.1007/978-3-319-23461-8_5]

10 Friedman JH. Greedy function approximation: A gradient boosting machine. The Annals of Statistics, 2001, 29(5): 1189–1232. [doi: 10.1214/aos/1013203451]

11 Geurts P, Ernst D, Wehenkel L. Extremely randomized trees. Machine Learning, 2006, 63(1): 3–42. [doi: 10.1007/s10994-006-6226-1]