上海大学

《数据挖掘》
课程研讨报告

课程研讨成绩（总分 10）		分　数
表达	有自己的见解，观点表述清楚，建议中肯（3-4 分）	
	照抄别人的想法，缺乏自己的思考（0-2 分）	
内容	内容清晰完整，语言逻辑通顺（2-3 分）	
	内容不完整，语言欠通顺（0-1 分）	
格式	格式正确，符合要求（2-3 分）	
	格式不正确，封面或字体有误（0-1 分）	

课程名称	数据挖掘
课程编号	08306150
学　号	姓　名
23121677	范舒舰

2026　年　4　月

《数据挖掘》课程研讨报告

1. 任务要求

1.1 题目描述

本任务基于 Porto Taxi Trajectory 数据集完成出租车轨迹终点预测。数据集中每条记录对应一次出租车完整行程，轨迹由按时间顺序记录的 GPS 点组成。

任务要求对每条轨迹只保留前部分轨迹点作为输入，并预测该行程最终到达的终点。建模方式可以采用回归方法，直接预测终点经纬度；也可以采用分类方法，先将终点空间离散化，再预测终点所属网格或聚类编号。

1.2 题目要求

本任务需要先将每条完整轨迹切分为“前缀轨迹”和“真实终点”。前缀轨迹作为模型输入，完整轨迹最后一个点作为预测目标。

在特征设计方面，需要从前缀轨迹中提取有效信息，例如前缀起点位置、当前最后一个轨迹点位置、前缀平均速度、出发时间、星期信息和前缀方向特征等。

在模型设计方面，可以选择经纬度回归或终点区域分类。本实验在直接回归的基础上进一步尝试终点聚类方法，将终点空间划分为聚类区域，再结合类内偏移回归得到最终终点位置。

在结果评价方面，需要使用距离误差衡量预测效果，并比较 20%、40%、60% 三种前缀长度下的误差变化。

2. 程序设计

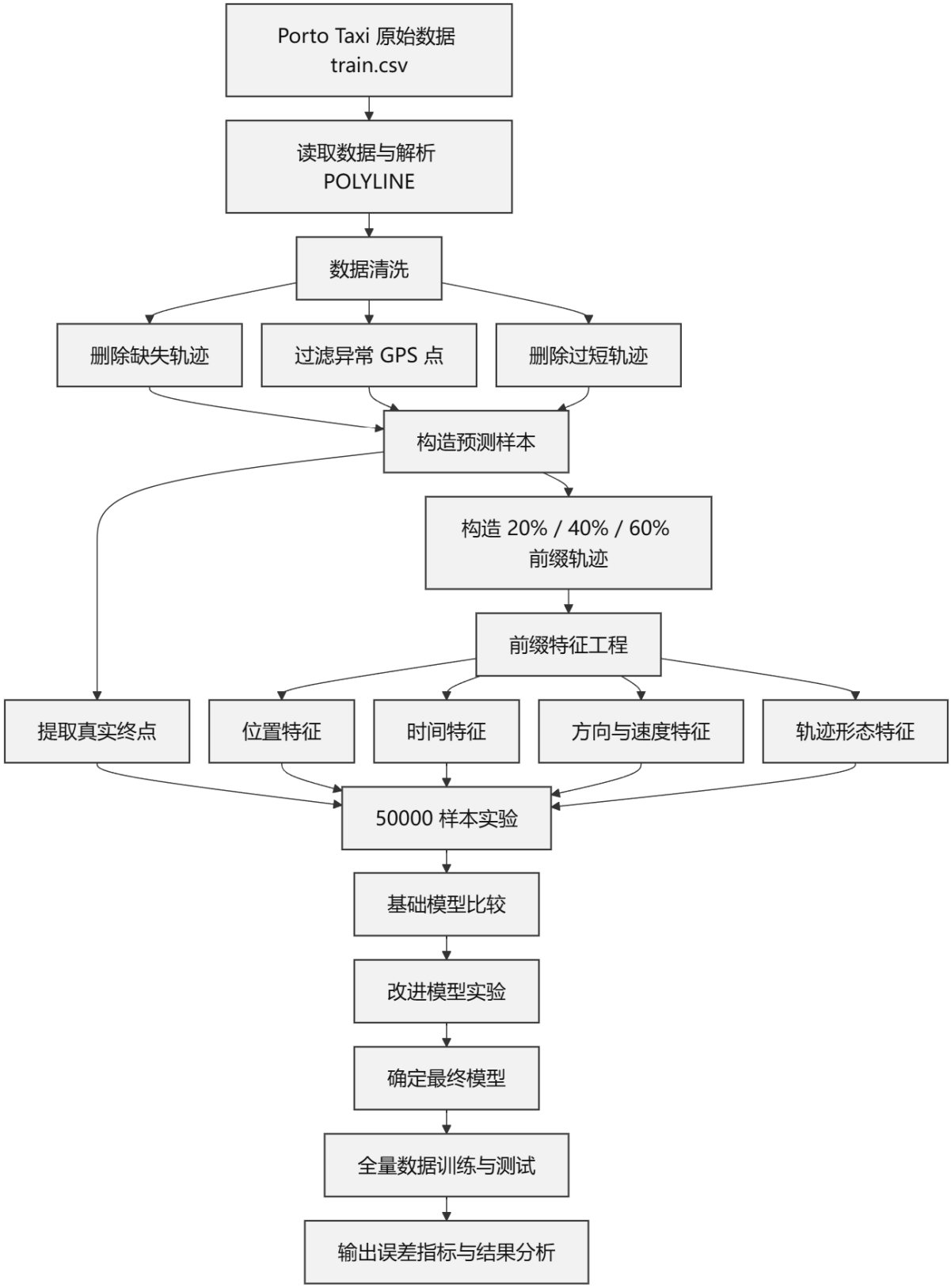

图 1 程序设计流程图

2.1 设计思路

本实验围绕 Porto Taxi Trajectory 数据集完成出租车终点预测任务。原始数据中，每一条记录对应一次出租车行程，主要字段包括 TRIP_ID、TAXI_ID、TIMESTAMP、MISSING_DATA 和 POLYLINE 等。其中，POLYLINE 字段最关键，它以字符串形式保存了一次行程中的 GPS 轨迹点序列，每个轨迹点由经度和纬度组成。本实验的目标不是根据完整轨迹预测终点，而是只保留轨迹前一部分作为输入，再预测该行程最终到达的位置。因此，程序设计的第一步不是建模，而是先把原始轨迹数据清洗并转换成适合预测任务的数据格式。

在数据清洗阶段，程序首先读取 train.csv 文件，并检查每条记录的 MISSING_DATA 字段。如果该字段为 True，说明这条轨迹存在缺失数据，程序会直接删除该样本，避免缺失轨迹影响后续建模。之后，程序解析 POLYLINE 字段。成功解析为轨迹点序列的数据，才会进入后续处理。

解析完成后，程序继续对轨迹点进行有效性检查。程序设置了一个大致的经纬度范围，用来过滤明显不属于该区域的异常 GPS 点。经度范围设置为 −8.80 到 −8.45，纬度范围设置为 41.00 到 41.35。如果某个轨迹点超出这个范围，就认为它可能是异常点或无效点，不参与后续特征计算。

同时，程序还会删除轨迹点数量过少的样本。本实验中，最短轨迹点数设置为 10。原因是轨迹点太少时，无法稳定提取前缀轨迹的速度、方向、弯曲度和分段距离等特征。

表 2-1 数据清洗流程

清洗步骤	处理内容	处理目的
缺失数据检查	删除 MISSING_DATA=True 的记录	避免缺失轨迹进入模型
轨迹字段解析	将 POLYLINE 字符串转换为经纬度点序列	将原始文本字段转换为可计算数据
异常点过滤	删除超出 Porto 经纬度范围的 GPS 点	减少异常坐标对距离和方向计算的影响
短轨迹删除	删除轨迹点数少于 10 的样本	保证轨迹前缀具有足够信息
终点提取	将完整轨迹最后一个点作为真实终点	构造模型预测目标
前缀构造	分别截取 20%、40%、60% 前缀	满足题目对不同前缀长度比较的要求

在 50000 条原始样本实验中，经过上述数据清洗和轨迹筛选后，最终保留了 48177 条有效轨迹。该数据量用于前期模型选择、参数比较和方法验证。全量实验中，原始数据共有 1710670 条记录，清洗后得到 1668152 条有效终点样本，用于验证最终模型在大规模数据上的表现。数据处理后的基本情况如表 2-2 所示。

表 2-2 数据处理后的基本情况

数据规模	原始记录数	有效记录数	用途
样本实验	50000	48177	用于模型选择、参数比较和方法验证
全量实验	1710670	1668152	用于最终模型效果验证

完成清洗后，程序将每条完整轨迹切分为“前缀轨迹”和“真实终点”。真实终点取完整轨迹最后一个 GPS 点。前缀部分分别取完整轨迹的 20%、40% 和 60%，对应行程早期、中段和较后阶段的预测场景。

特征工程围绕位置、时间、方向和前缀状态展开。程序从前缀轨迹中提取多类特征。这样可以更完整地描述车辆已经行驶的状态。

表 2-3 前缀轨迹特征设计

特征类别	主要特征	含义
位置特征	start_lng、start_lat、current_lng、current_lat	起点和当前点位置
轨迹采样点	p25_lng、p25_lat、p50_lng、p50_lat、p75_lng、p75_lat	保留前缀轨迹中间形态
时间特征	hour、weekday、is_weekend	出发时间和星期信息
前缀状态	prefix_len、prefix_duration_sec、prefix_distance_km	已观察到的轨迹长度、时间和距离
速度与距离	prefix_avg_speed、direct_distance_km	平均速度和起点到当前点直线距离
方向与形态	direction_angle、recent_direction_angle、angle_change、tortuosity	行驶方向、方向变化和轨迹弯曲程度
分段距离	seg_d1、seg_d2、seg_d3、seg_d4	前缀不同阶段的运动变化

特征设计的出发点是避免模型只依赖当前点。两辆出租车即使位于相同位置，只要出发点、行驶方向、速度和前缀轨迹形状不同，最终目的地也可能不同。

模型设计从简单基线开始。Baseline_CurrentPoint 直接将前缀轨迹最后一个

点作为预测终点，不需要训练。该方法用于衡量后续模型是否真正有效。50000 样本实验中，该基线在 20%、40%、60% 前缀下的平均误差分别为 3.2147 km、2.5852 km 和 1.7087 km。随着前缀比例增加，基线误差明显下降，但 60% 前缀下仍有约 1.7 km 的平均误差，说明仅使用当前位置仍然不够。

基础模型比较阶段，程序测试了 KNN、RandomForest、ExtraTrees、HistGradientBoosting、XGBoost 和 XGBoostDeep（增强参数版）。所有模型都使用相同的前缀轨迹特征，输出形式都是终点经纬度。实验结果如表 2-4 所示。

表 2-4 基础模型平均误差对比

模型	20%前缀/km	40%前缀/km	60%前缀/km
Baseline_CurrentPoint	3.2147	2.5852	1.7087
RandomForest	2.3498	1.6836	1.0383
ExtraTrees	2.3322	1.6701	1.0347
HistGradientBoosting	2.4242	1.7854	1.1447
KNN	2.4892	1.8464	1.2489
XGBoost	2.3533	1.7144	1.0858
XGBoostDeep	2.3247	1.6686	1.0459

从基础模型结果看，所有机器学习模型都明显优于当前点基线（Baseline_CurrentPoint），说明前缀特征对终点预测有效。KNN 的效果相对较弱，说明只依赖普通特征空间中的近邻样本不够稳定。随机森林和 ExtraTrees 能处理非线性关系，效果优于 KNN。XGBoostDeep 在 20% 和 40% 前缀下表现最好，在 60% 前缀下虽然略低于 ExtraTrees，但整体稳定性更强。因此，后续模型设计选择 XGBoostDeep 作为主要强基准。

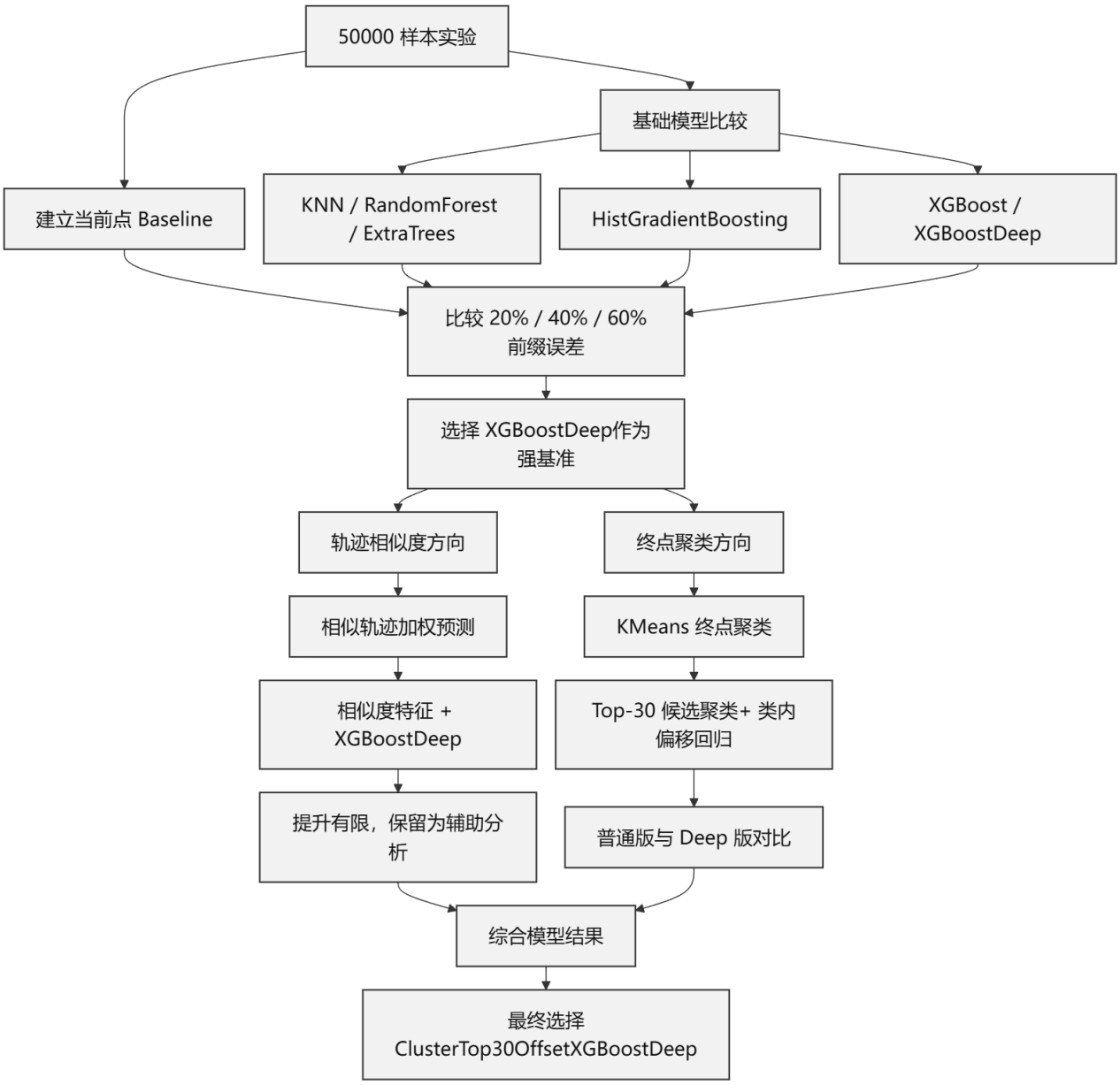

图 2 模型选择与实验路线图

在确定强基准后，继续尝试两类更贴合轨迹任务的数据分析方向：轨迹相似度分析和终点聚类分析。

轨迹相似度分析的出发点是：如果两条出租车在前缀阶段的轨迹形状相似，它们可能具有相似的目的地区域。程序将前缀轨迹中的起点、中间采样点和当前点转换为固定长度向量，在训练集中查找相似轨迹，并用相似轨迹的真实终点进行加权预测。该方法不只是为了追求误差下降，也用于验证“轨迹形状相似性”是否能为终点预测提供额外信息。

实验结果显示，轨迹相似度单独预测优于简单基线，但弱于 XGBoostDeep；

将相似轨迹预测结果、近邻距离和近邻终点离散程度加入 XGBoost 后，只带来小幅提升。说明轨迹相似度具有解释价值，但不能作为最终主模型。

出租车终点通常不会均匀分布在城市空间中，而是集中在若干高频区域，如交通枢纽、商业区、住宅区或道路节点附近。全量终点聚类实验中，轮廓系数约为 0.4078，说明终点存在一定空间聚集结构。直接回归经纬度时，模型需要在整个城市范围内一次性预测终点；聚类方法可以先判断大致目的地区域，再在该区域内部预测具体位置，更符合终点分布特点。

基于这两类分析，程序进一步比较了轨迹相似度增强模型、终点聚类模型和混合模型。主要结果如表 2-5 所示。

表 2-5 改进模型平均误差对比

模型	20%前缀/km	40%前缀/km	60%前缀/km
XGBoostDeep	2.3247	1.6686	1.0459
TrajectorySimilarityOnly	2.4606	1.8277	1.1807
SimilarityEnhancedXGBoost	2.3111	1.6635	1.0472
ClusterTop30ffsetXGBoost	2.3932	1.6557	1.0190
ClusterTop300ffsetXGBoost	2.3053	1.6345	1.0065
ClusterTop300ffsetXGBoostDeep	2.3039	1.6324	1.0047

程序设计了 ClusterTop30OffsetXGBoostDeep：先用 KMeans 将训练集终点划分为 50 个聚类区域，再用 XGBoost 分类器预测测试轨迹可能到达的 Top-30 个区域，随后将前缀特征与候选聚类中心拼接，使用 XGBoost 回归终点相对聚类中心的偏移量。在保留聚类结构的同时，提高区域内部精细定位能力。

表 2-6 ClusterTop30 普通版与 Deep 版对比

前缀比例	普通 ClusterTop30/km	ClusterTop30Deep/km	Top-30 命中率
20%	2.3053	2.3039	0.9601
40%	1.6345	1.6324	0.9825
60%	1.0065	1.0047	0.9920

该模型不是单纯堆叠复杂模型，而是利用了两个已被实验验证的信息：一是 XGBoostDeep 适合作为强基准学习器，二是终点具有明显聚集结构。模型将终点预测拆成“候选区域判断”和“区域内偏移回归”两个步骤，既保留了分类方法对多目的地区域的表达能力，也保留了回归方法对连续经纬度的精细预测能力。

2.2 算法描述

本实验的算法流程由三部分组成：轨迹前缀构造、特征提取和终点预测模型。程序首先将原始轨迹清洗并切分为不同长度的前缀轨迹，然后从前缀中提取位置、时间、方向和运动状态特征，最后使用 ClusterTop30OffsetXGBoostDeep 预测终点经纬度。该模型先判断车辆可能到达的候选终点区域，再在候选区域内部预测具体位置。

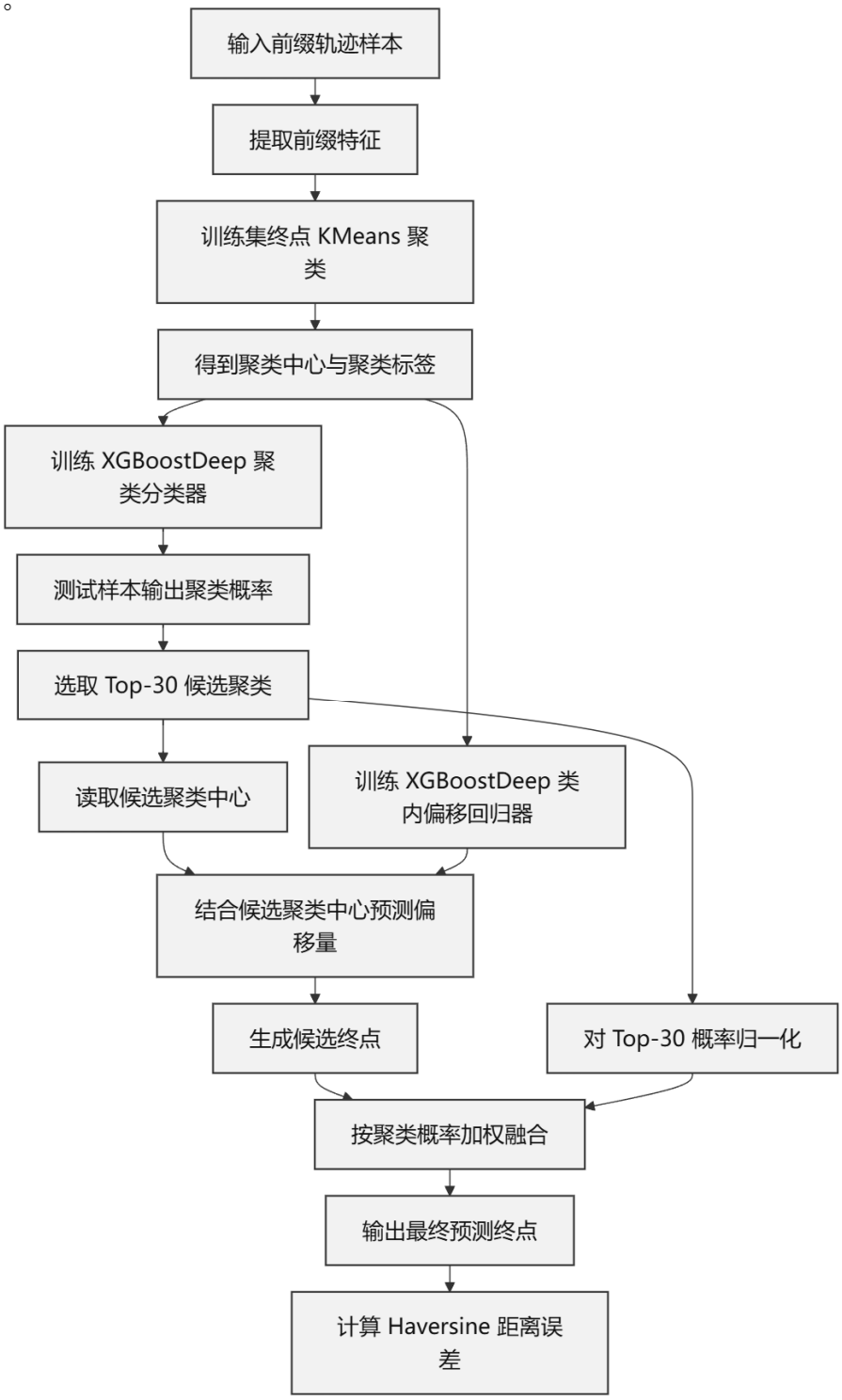

图 3 ClusterTop30OffsetXGBoostDeep 算法流程图

2.2.1 轨迹前缀构造

设第 i 条完整轨迹为 T_i，其中包含 n_i 个 GPS 点。每个轨迹点由经度和纬度组成。对于前缀比例 r，程序截取前 m_i 个轨迹点作为输入，并将完整轨迹最后一个点作为真实终点。

$$T_i = \{p_{i1}, p_{i2}, ..., p_{in_i}\}, \quad p_{ij} = (lon_{ij}, lat_{ij})$$

$$m_i = \max(2, \lfloor r \cdot n_i \rfloor), \quad P_i^{(r)} = \{p_{i1}, p_{i2}, ..., p_{im_i}\}, \quad y_i = p_{in_i}$$

其中，r 分别取 0.2、0.4 和 0.6，对应 20%、40%、60% 三种前缀长度。这样可以模拟出租车行程在不同阶段进行终点预测的场景。20% 前缀表示早期预测，已知轨迹较少；60% 前缀表示车辆已经行驶较长距离，目的地信息更明显。

2.2.2 特征提取

程序从前缀轨迹 $P_i^{(r)}$ 中提取特征，形成模型输入向量 x_i 。特征包括起点、当前点、轨迹采样点、出发时间、星期、前缀距离、平均速度、方向角、弯曲度和分段距离等。

$$x_i = f(P_i^{(r)})$$

其中，$f(\cdot)$ 表示特征提取过程。特征设计主要覆盖四类信息：位置、时间、方向和前缀运动状态。这样做是为了避免模型只依赖当前点，而是同时利用车辆已经行驶过的路径信息。

表 2-7 主要输入特征

特征类别	代表特征	含义
位置特征	起点、当前点、轨迹采样点	描述车辆已行驶到的位置
时间特征	小时、星期、是否周末	描述出行时间背景
距离与速度	前缀距离、直线距离、平均速度	描述车辆运动状态
方向与形态	方向角、方向变化、弯曲度	描述轨迹走势
分段特征	四段轨迹距离	描述前缀不同阶段的变化

2.2.3 终点聚类

数据分析发现，出租车终点在空间上具有聚集特征。因此，最终模型没有直接在整个城市范围内回归终点，而是先将训练集终点划分为若干目的地区域。程序使用 KMeans 对训练集真实终点进行聚类，聚类数量设为 $K = 50$。

$$\min_{\mu_1,\ldots,\mu_K} \sum_{i=1}^{N} \|y_i - \mu_{c_i}\|^2, K = 50$$

其中，μ_k 表示第 k 个终点聚类中心，c_i 表示第 i 条轨迹终点所属的聚类编号。聚类后，终点预测任务可以先转化为目的地区域预测，再进一步预测区域内部的具体坐标。

2.2.4 Top-30 候选聚类与类内偏移回归

得到终点聚类标签后，程序使用 XGBoostDeep 分类器，根据前缀特征 x_i 预测该轨迹属于每个终点聚类的概率。之后，选取概率最高的 Top-30 个聚类作为候选目的地区域。

$$\pi_{ik} = P(c_i = k|x_i), S_i^{30} = \mathrm{Top30}(\pi_{i1}, \pi_{i2}, \ldots, \pi_{iK})$$

只选择 Top-1 聚类容易受到分类错误影响。特别是在 20% 前缀下，车辆目的地仍然不够明确，保留 Top-30 候选区域可以减少单一分类错误带来的偏差。

对于每个候选聚类中心 μ_k，程序将前缀特征和聚类中心拼接，输入 XGBoostDeep 回归器，预测终点相对聚类中心的偏移量。

$$z_{ik} = [x_i, \mu_k], \quad \Delta_{ik} = g(z_{ik}), \quad \hat{y}_{ik} = \mu_k + \Delta_{ik}$$

其中，z_{ik} 是候选聚类下的回归输入，$g(\cdot)$ 表示类内偏移回归模型，$\hat{y}_{ik}$ 表示第 k 个候选区域给出的候选终点。这个步骤使模型不只是输出聚类中心，而是能进一步预测区域内部的具体位置。

2.2.5 概率加权融合

每个测试样本会得到 30 个候选终点。程序根据分类器输出的聚类概率，对这些候选终点进行加权融合，得到最终预测终点。

$$w_{ik} = \frac{\pi_{ik}}{\sum_{j \in S_i^{30}} \pi_{ij}}, \hat{y}_i = \sum_{k \in S_i^{30}} w_{ik} \hat{y}_{ik}$$

该步骤把多个候选区域的结果综合起来。当前缀较短时，不同候选区域可以共同参与预测；当前缀较长时，分类概率通常会更集中，权重也会更偏向最可能的目的地区域。

2.2.6 误差评价

预测结果使用真实终点与预测终点之间的球面距离进行评价。经纬度点位于地球表面，直接使用普通欧氏距离不够准确，因此程序使用 Haversine 距离。最终主要统计平均误差。

$$\text{Error} = \frac{1}{N}\sum_{i=1}^{N} d\,(y_i, \hat{y}_i)$$

其中，$d(y_i, \hat{y}_i)$ 表示真实终点和预测终点之间的 Haversine 距离，单位为千米。程序还会输出中位数误差、90% 分位误差和最大误差，用于观察模型在多数样本和较大误差样本上的表现。

表 2-8 最终模型各模块作用

模块	输入	输出
前缀构造	完整轨迹	20%、40%、60% 前缀
特征提取	前缀轨迹	位置、时间、方向、运动特征
KMeans 聚类	训练集真实终点	聚类中心和聚类标签
聚类分类器	前缀特征	各终点聚类概率
Top-30 选择	聚类概率	30 个候选聚类
类内偏移回归	前缀特征 + 聚类中心	终点偏移量
加权融合	候选终点 + 聚类概率	最终预测终点
误差计算	预测终点和真实终点	距离误差

ClusterTop30OffsetXGBoostDeep 的有效性来自两个方面。KMeans 聚类利用了终点空间聚集的特点，使模型先判断目的地区域，而不是直接在全城市范围内回归坐标。Top-30 候选区域保留了短前缀下的目的地不确定性，类内偏移回归又能在候选区域内部进一步修正位置。同时保留了分类方法的区域判断能力和回归方法的坐标预测能力。

2.3 程序源码

本实验完整程序包含数据清洗、前缀构造、特征提取、基础模型比较、轨迹相似度实验、终点聚类实验和全量运行等多个脚本。由于完整源码较长，正文中只保留最终模型 ClusterTop30OffsetXGBoostDeep 的核心代码。该代码体现了本实验的主要改进点：终点聚类、Top-30 候选区域选择、类内偏移回归和概率加权融合。

完整项目工程包含全部 Python 源码、运行脚本、样本实验结果、全量实验结果和部分可视化图片。为便于复现，项目工程已整理并上传至百度网盘，链接如下：

链接：https://pan.baidu.com/s/1V3lIGJ55hziJa2GKKVlWbw?pwd=0000

提取码：0000

正文中展示的代码为最终主模型的核心实现。完整工程还包括数据清洗、前

缀构造、基础模型比较、轨迹相似度实验、混合模型实验、CPU 全量运行和 GPU 全量运行等代码，均可在上述项目工程中查看。

代码 2-1 ClusterTop30OffsetXGBoostDeep 核心代码

```
import numpy as np
import pandas as pd
from sklearn.cluster import MiniBatchKMeans
from sklearn.multioutput import MultiOutputRegressor
from xgboost import XGBClassifier, XGBRegressor

BASE_FEATURE_COLUMNS = [
    "hour", "weekday", "is_weekend",
    "start_lng", "start_lat",
    "current_lng", "current_lat",
    "prefix_len", "prefix_duration_sec",
    "prefix_distance_km", "direct_distance_km",
    "prefix_avg_speed",
    "direction_angle", "recent_direction_angle",
    "lng_delta", "lat_delta",
    "p25_lng", "p25_lat",
    "p50_lng", "p50_lat",
    "p75_lng", "p75_lat",
    "tortuosity", "angle_change",
    "lng_range", "lat_range",
    "seg_d1", "seg_d2", "seg_d3", "seg_d4",
]

def make_xgb_deep_regressor():
    # 增强参数版 XGBoost 回归器。
    model = XGBRegressor(
        n_estimators=1200,
        max_depth=8,
        learning_rate=0.02,
        subsample=0.90,
        colsample_bytree=0.90,
        objective="reg:squarederror",
        tree_method="hist",
        n_jobs=-1,
        random_state=42,
    )
    return MultiOutputRegressor(model)

def make_xgb_classifier(num_classes):
    # XGBoost 聚类分类器。
```

```
    return XGBClassifier(
        n_estimators=700,
        max_depth=6,
        learning_rate=0.035,
        subsample=0.90,
        colsample_bytree=0.90,
        objective="multi:softprob",
        num_class=num_classes,
        eval_metric="mlogloss",
        tree_method="hist",
        n_jobs=-1,
        random_state=42,
    )

def train_offset_model(train_df, train_labels, centers_df):
    # 训练类内偏移回归器。目标不是直接预测终点，而是预测终点相对聚类中心的偏移量。
    data = train_df.copy()
    data["cluster_id"] = train_labels

    data = data.merge(
        centers_df[["cluster_id", "center_lng", "center_lat"]],
        on="cluster_id",
        how="left",
    )
    data["offset_lng"] = data["target_lng"] - data["center_lng"]
    data["offset_lat"] = data["target_lat"] - data["center_lat"]
    feature_cols = BASE_FEATURE_COLUMNS + ["center_lng", "center_lat"]
    model = make_xgb_deep_regressor()
    model.fit(
        data[feature_cols].values,
        data[["offset_lng", "offset_lat"]].values,
    )

    return model, feature_cols

def    predict_with_top30(test_df,    top_clusters,    top_probs,    centers_df,
offset_model, feature_cols):
    # 对 Top-30 候选区域分别预测类内偏移，再按照候选区域概率加权得到最终终点。
    center_lng = centers_df.set_index("cluster_id")["center_lng"].to_dict()
    center_lat = centers_df.set_index("cluster_id")["center_lat"].to_dict()

    pred_lng = np.zeros(len(test_df))
    pred_lat = np.zeros(len(test_df))
```

```
    base_features = test_df[BASE_FEATURE_COLUMNS].reset_index(drop=True)

    for rank in range(top_clusters.shape[1]):
        candidate = top_clusters[:, rank]
        weight = top_probs[:, rank]

        temp = base_features.copy()
        temp["center_lng"] = [center_lng[int(c)] for c in candidate]
        temp["center_lat"] = [center_lat[int(c)] for c in candidate]

        offset = offset_model.predict(temp[feature_cols].values)

        pred_lng += (temp["center_lng"].values + offset[:, 0]) * weight
        pred_lat += (temp["center_lat"].values + offset[:, 1]) * weight

    return pred_lng, pred_lat

def  run_cluster_top30_offset_xgboost_deep(train_df,  test_df,  n_clusters=50,
top_k=30):
    # ClusterTop30OffsetXGBoostDeep 主流程。输入为前缀轨迹特征，输出为预测终点经纬度。
    # 1. 对训练集真实终点进行聚类
    train_xy = lonlat_to_xy_km(
        train_df["target_lng"].values,
        train_df["target_lat"].values,
    )
    kmeans = MiniBatchKMeans(
        n_clusters=n_clusters,
        batch_size=8192,
        random_state=42,
        n_init=10,
    )

    train_labels = kmeans.fit_predict(train_xy)

    center_lng, center_lat = xy_km_to_lonlat(
        kmeans.cluster_centers_[:, 0],
        kmeans.cluster_centers_[:, 1],
    )

    centers_df = pd.DataFrame({
        "cluster_id": np.arange(n_clusters),
        "center_lng": center_lng,
        "center_lat": center_lat,
    })
```

```
    # 2. 训练聚类分类器，输出每个终点区域的概率
    clf = make_xgb_classifier(num_classes=n_clusters)
    clf.fit(train_df[BASE_FEATURE_COLUMNS].values, train_labels)

    prob_matrix = clf.predict_proba(test_df[BASE_FEATURE_COLUMNS].values)

    # 3. 选择 Top-30 候选终点区域
    top_pos = np.argsort(-prob_matrix, axis=1)[:, :top_k]
    top_probs = np.take_along_axis(prob_matrix, top_pos, axis=1)
    top_clusters = clf.classes_[top_pos]

    top_probs = top_probs / top_probs.sum(axis=1, keepdims=True)

    # 4. 训练类内偏移回归器
    offset_model, feature_cols = train_offset_model(
        train_df=train_df,
        train_labels=train_labels,
        centers_df=centers_df,
    )

    # 5. 对 Top-30 候选区域加权预测最终终点
    pred_lng, pred_lat = predict_with_top30(
        test_df=test_df,
        top_clusters=top_clusters,
        top_probs=top_probs,
        centers_df=centers_df,
        offset_model=offset_model,
        feature_cols=feature_cols,
    )

    return pred_lng, pred_lat
```

完整实现保存在工程文件中，正文中不展开辅助函数，避免源码部分过长。

上述代码展示了最终模型的主要创新点：MiniBatchKMeans 负责将终点空间划分为目的地区域，XGBClassifier 负责预测 Top-30 候选终点区域，XGBoostDeep 回归器负责预测候选区域内部的终点偏移量。最终结果不是单个聚类中心，而是多个候选终点按概率加权后的经纬度坐标。这样既保留了终点聚类的区域判断能力，也保留了回归模型对连续坐标的细化能力。

2.4 运行结果

实验先在 50000 条样本数据上完成模型筛选，再使用全量数据进行验证。50000 样本清洗后得到 48177 条有效轨迹，样本轨迹平均点数为 48.8649，平均行程距离为 6.1573 km，平均行程时长为 717.9734 秒。全量实验使用的有效样本数为 1640633 条。

2.4.1 基准模型选择结果

基础模型实验的目标是从常见机器学习模型中选出较强的基准模型。所有模型使用相同的前缀轨迹特征，预测目标均为终点经纬度。评价指标采用预测终点与真实终点之间的平均距离误差，单位为 km。

表 2-9 基础模型平均误差对比

模型	20%前缀/km	40%前缀/km	60%前缀/km
Baseline_CurrentPoint	3.2147	2.5852	1.7087
RandomForest	2.3498	1.6836	1.0383
ExtraTrees	2.3322	1.6701	1.0347
HistGradientBoosting	2.4242	1.7854	1.1447
KNN	2.4892	1.8464	1.2489
XGBoost	2.3533	1.7144	1.0858
XGBoostDeep	2.3247	1.6686	1.0459

从表 2-9 可以看出，所有训练模型都明显优于 Baseline_CurrentPoint，说明前缀轨迹特征对终点预测是有效的。KNN 的误差相对较高，说明仅依赖普通特征空间中的近邻样本不够稳定。RandomForest 和 ExtraTrees 效果较好，能够处理一定的非线性关系。HistGradientBoosting 整体表现不如 XGBoost 系列。

XGBoostDeep 在 20% 和 40% 前缀下取得最低误差，分别为 2.3246 km 和 1.6680 km。在 60% 前缀下，ExtraTrees 的平均误差为 1.0347 km，略低于 XGBoostDeep 的 1.0500 km。但从三个前缀比例整体看，XGBoostDeep 更稳定，并且与后续聚类分类和类内偏移回归模型结构一致。因此，后续实验将 XGBoostDeep 作为主要强基准模型。

前缀长度对预测误差的影响也比较明显。以 XGBoostDeep 为例，平均误差从 20% 前缀下的 2.3246 km 降至 60% 前缀下的 1.0500 km。车辆行驶轨迹越长，模型能够获得的位置、方向和路径形态信息越充分，终点预测难度随之下降。

2.4.2 轨迹相似度与终点聚类改进结果

在确定 XGBoostDeep 作为强基准后，实验继续比较轨迹相似度方法和终点聚类方法。轨迹相似度方法用于检验“前缀形状相似的历史轨迹是否具有相似终点”；终点聚类方法用于利用出租车终点在城市空间中的聚集特征。

终点聚类实验首先通过肘部图观察不同聚类数量下的聚类变化趋势。该图用于辅助说明聚类数选择的合理性。

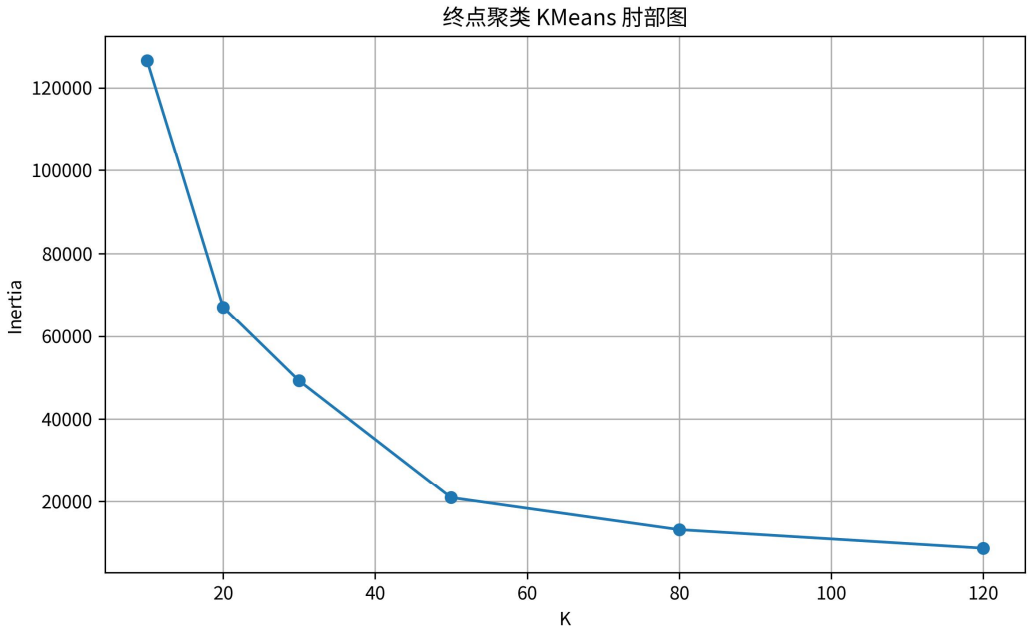

图 4 终点聚类肘部图

从样本终点聚类结果看，随着聚类数量增加，聚类损失持续下降，但下降幅度逐渐变缓。实验最终采用 $K = 50$ 作为终点聚类数量，在聚类精度和后续分类复杂度之间取得平衡。

随后，对样本终点进行空间可视化。该图用于展示终点在城市空间中的分布情况。

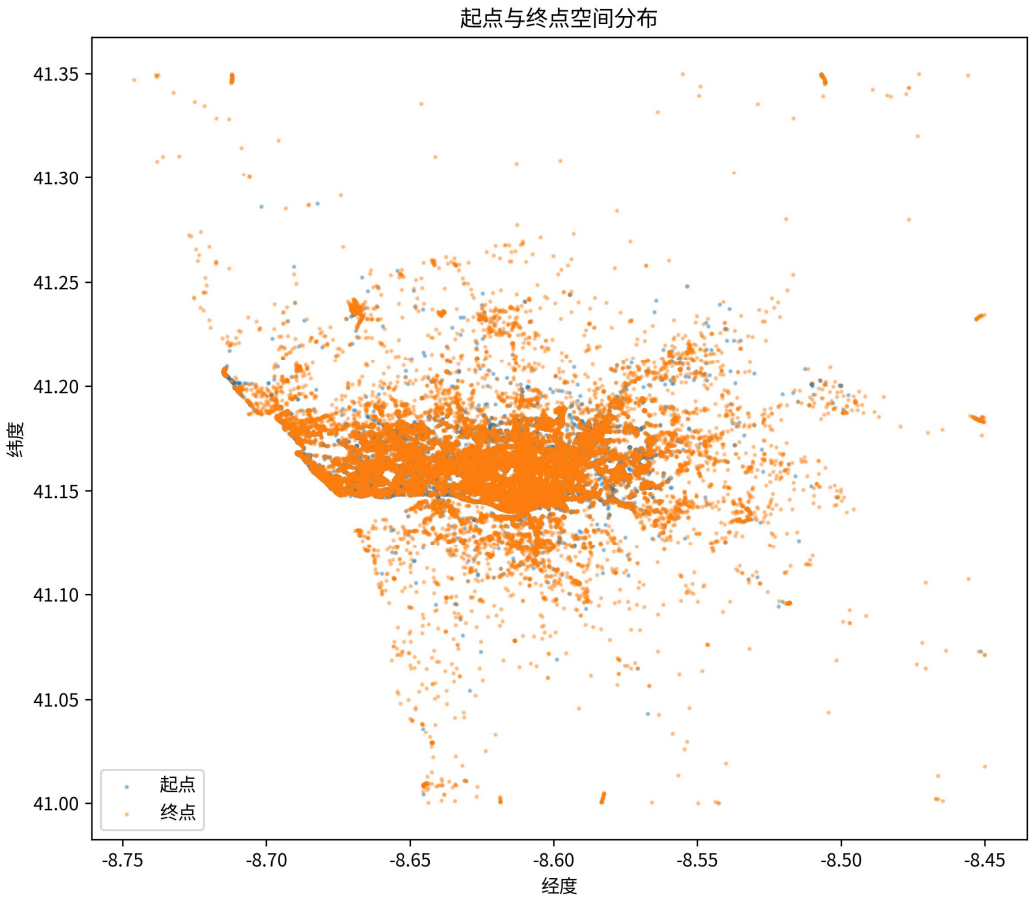

图 5 样本终点聚类散点图

从图 5 可以看出，出租车终点并不是均匀分布在城市空间中，而是集中在若干区域。这说明终点预测不仅可以看作普通经纬度回归问题，也可以先判断目的地区域，再在区域内部进一步预测具体终点位置。

为了进一步观察高频终点区域，实验统计了终点密度最高的聚类区域。

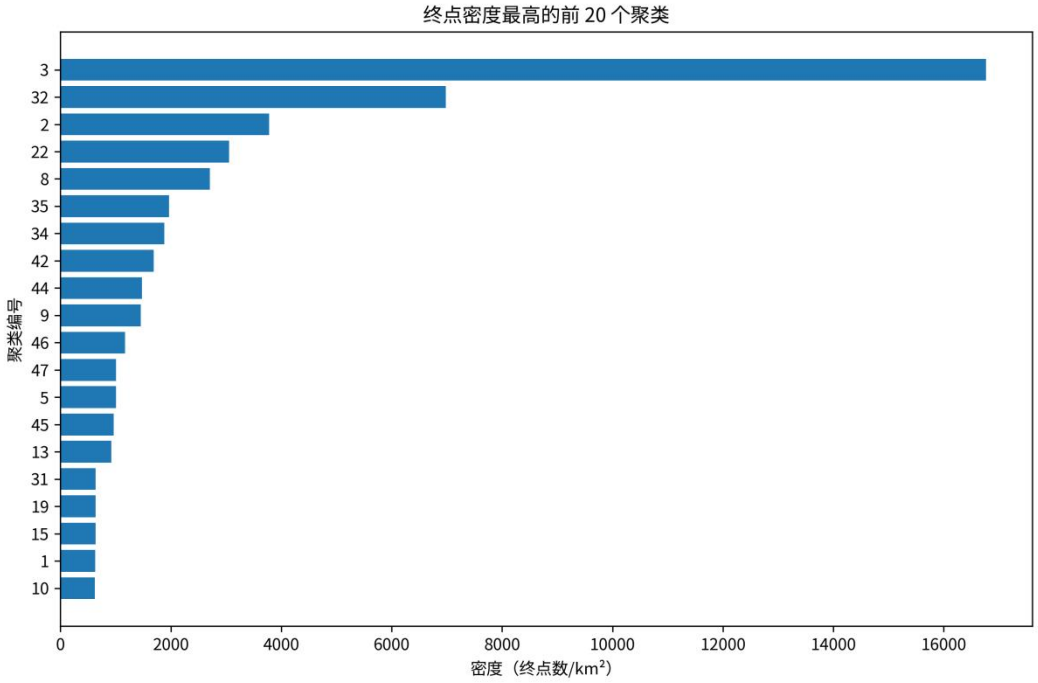

图 6 样本终点聚类密度 Top20

图 6 显示了样本中终点密度最高的 20 个聚类区域。部分区域的终点密度明显高于其他区域，说明出租车终点具有较强的空间集中性。该现象为后续设计 ClusterTop30OffsetXGBoostDeep 提供了依据。

样本数据的终点聚类轮廓系数为 0.4206，全量数据的轮廓系数为 0.4150。该结果说明终点分布具有一定聚类结构。基于这一观察，实验进一步比较了轨迹相似度方法、相似度增强模型和终点聚类增强模型。

表 2-10 改进模型平均误差对比

模型	20%前缀/km	40%前缀/km	60%前缀/km
XGBoostDeep	2.3247	1.6686	1.0459
TrajectorySimilarityOnly	2.4606	1.8277	1.1807
SimilarityEnhancedXGBoost	2.3111	1.6635	1.0472
ClusterTop30ffsetXGBoost	2.3932	1.6557	1.0190
ClusterTop300ffsetXGBoost	2.3053	1.6345	1.0065
ClusterTop300ffsetXGBoostDeep	2.3039	1.6324	1.0047

从表 2-10 可以看出，TrajectorySimilarityOnly 的误差低于当前点基线，但没有超过 XGBoostDeep。在 20%、40%、60% 三种前缀下，轨迹相似度单独预测的平均误差分别为 2.4606 km、1.8277 km 和 1.1807 km，均高于

XGBoostDeep。这说明相似轨迹能够提供一定参考，但仅依靠历史相似轨迹的终点进行加权预测，稳定性仍然不足。

SimilarityEnhancedXGBoost 在轨迹相似度结果的基础上，将相似轨迹预测终点、近邻距离和近邻终点离散程度等信息加入 XGBoost 模型。该方法在 20% 和 40% 前缀下略优于 XGBoostDeep，平均误差分别从 2.3247 km 降至 2.3111 km、从 1.6686 km 降至 1.6635 km。但在 60% 前缀下，误差为 1.0472 km，略高于 XGBoostDeep 的 1.0459 km。说明轨迹相似度特征有一定补充作用，但提升幅度有限，并不适合作为最终主模型。

终点聚类方法的效果更明显。ClusterTop3OffsetXGBoost 在 40% 和 60% 前缀下已经优于 XGBoostDeep，但在 20% 前缀下误差较高。这说明短前缀阶段目的地不确定性较强，只保留 Top-3 候选聚类容易漏掉真实终点区域。将候选范围扩大到 Top-30 后，ClusterTop30OffsetXGBoost 在三种前缀比例下均优于 XGBoostDeep，平均误差分别降至 2.3053 km、1.6345 km 和 1.0065 km。该结果说明，先预测多个候选终点区域，再在候选区域内进行偏移回归，比直接回归经纬度更适合本任务。

在此基础上，进一步将类内偏移回归器替换为增强参数版 XGBoost，得到 ClusterTop30OffsetXGBoostDeep。与普通 ClusterTop30OffsetXGBoost 相比，Deep 版在 20%、40%、60% 前缀下的误差分别从 2.3053 km 降至 2.3039 km、从 1.6345 km 降至 1.6324 km、从 1.0065 km 降至 1.0047 km。虽然提升幅度不大，但三个前缀比例下方向一致，因此最终选择 ClusterTop30OffsetXGBoostDeep 作为主模型。

表 2-11 ClusterTop30OffsetXGBoostDeep 聚类命中率

前缀比例	Top-1 命中率	Top-3 命中率	Top-10 命中率	Top-30 命中率
20%	0.2291	0.4503	0.7646	0.9609
40%	0.3553	0.6328	0.8826	0.9826
60%	0.5169	0.7994	0.9502	0.9924

表 2-11 说明了为什么保留 Top-30 候选聚类是必要的。Top-1 命中率并不高，特别是 20% 前缀下只有 0.2291。如果只选择一个最可能的聚类区域，短前缀样本很容易因为区域判断错误导致预测偏离真实终点。Top-30 命中率在三种前缀下都超过 0.96，说明真实终点大概率位于候选区域中。后续类内偏移回归

可以在这些候选区域中进一步修正位置。

2.4.3 全量运行结果

样本实验确定最终模型后，使用全量数据进行验证。全量运行使用 90% 数据训练、10% 数据测试，比较三个模型：Baseline_CurrentPoint、XGBoost_Deep_DirectRegression 和 ClusterTop30OffsetXGBoostDeep。全量实验有效样本数为 1640633 条。

全量实验中也对终点分布进行了可视化，用于验证样本阶段观察到的聚类结构是否仍然存在。

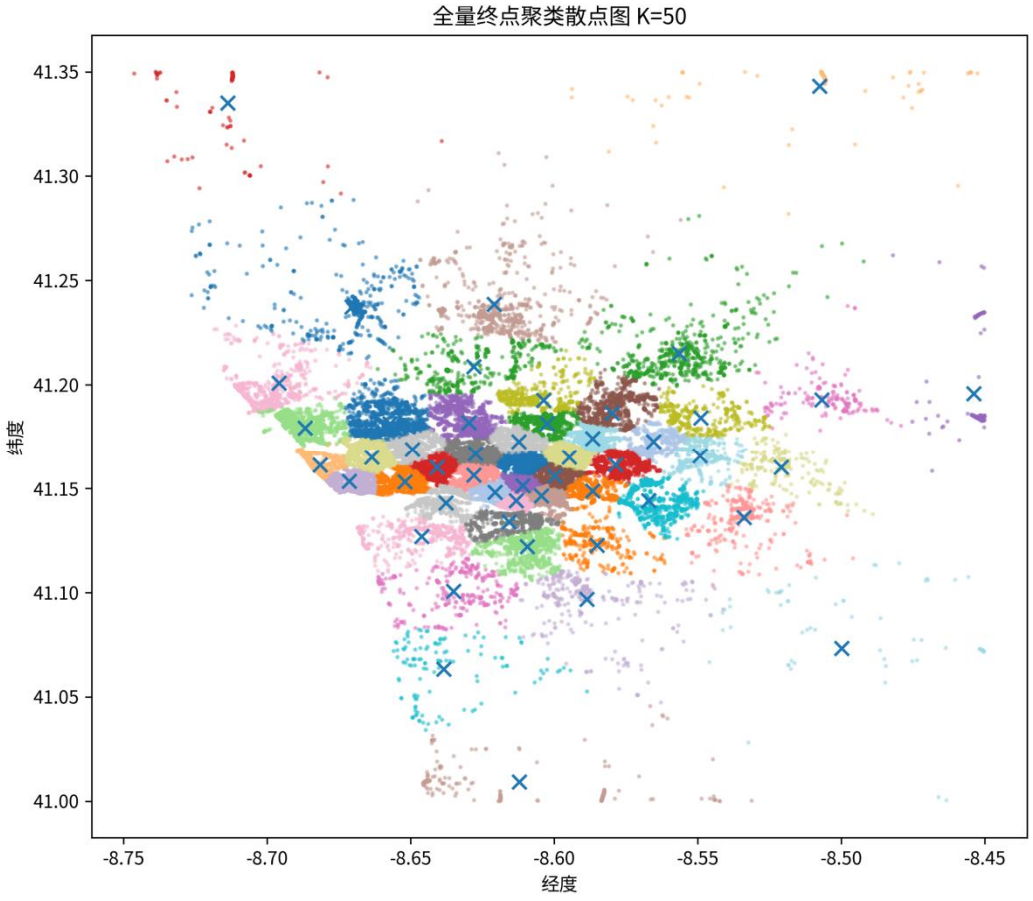

图 7 全量终点聚类散点图

从图 7 可以看出，全量数据中的终点仍然呈现明显的空间聚集现象。这与样本实验结果一致，说明样本阶段得到的终点聚类规律不是偶然结果。

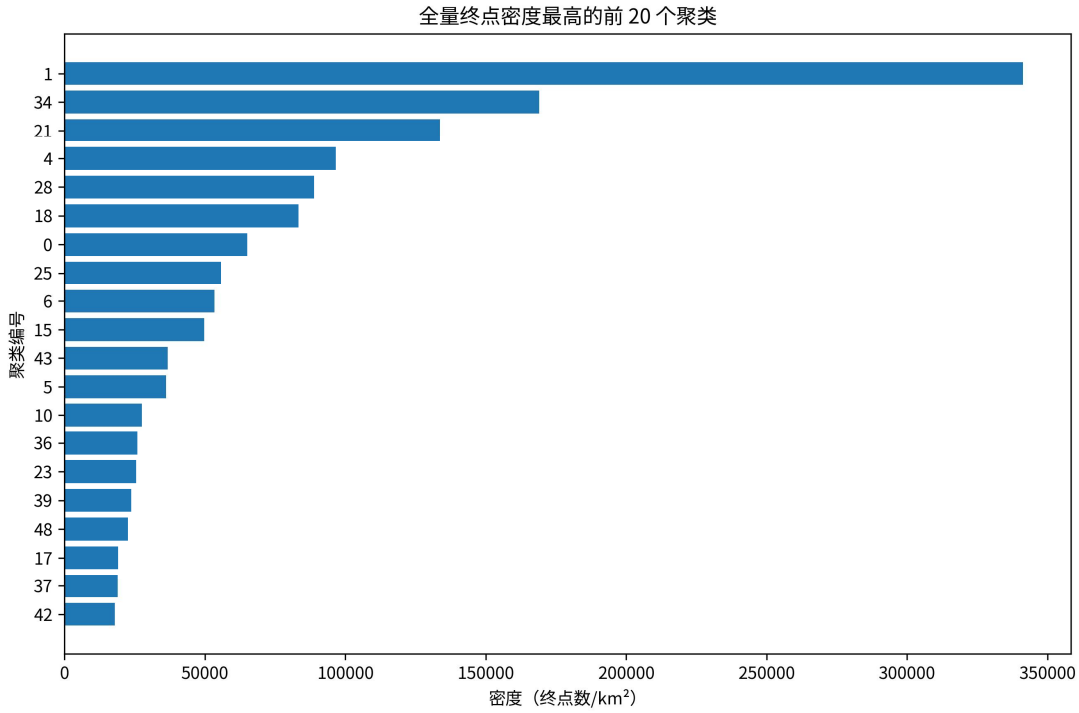

图 8 全量终点聚类密度 Top20

图 8 展示了全量数据中终点密度最高的聚类区域。高密度区域的存在说明终点空间分布具有明显热点，这进一步支持了“先预测目的地区域，再预测区域内部偏移”的模型设计。全量终点聚类的轮廓系数为 0.4150，与样本实验接近，说明终点聚集结构在更大规模数据上仍然稳定。

表 2-12 全量实验平均误差对比

前缀比例	Baseline/km	XGBoostDeep/km	ClusterTop30Deep/km	相比 Baseline 提升	相比 XGBoostDeep 提升
20%	3.1416	2.1511	2.1505	31.55%	0.03%
40%	2.5104	1.5209	1.4857	40.82%	2.32%
60%	1.6377	0.9378	0.8840	46.02%	5.74%

全量实验结果与样本实验趋势一致。随着前缀比例从 20% 增加到 60%，三个模型的误差都明显下降。最终模型 ClusterTop30OffsetXGBoostDeep 在三种前缀比例下都取得最低平均误差。20% 前缀下，它与 XGBoostDeep 的差距很小，说明行程早期可用信息较少，聚类模型的优势还不明显。40% 前缀下，最终模型平均误差为 1.4857 km，相比 XGBoostDeep 提升 2.32%。60% 前缀下，最终模型平均误差降至 0.8840 km，相比 XGBoostDeep 提升 5.74%。

这说明前缀越长，终点区域判断越准确，聚类模型的优势越明显。短前缀阶段，车辆目的地仍然有较强不确定性；中后段轨迹提供了更多方向和空间位置信息，模型更容易判断候选终点区域，类内偏移回归也能更有效地修正坐标。

表 2-13 全量实验聚类命中率

前缀比例	Top-1 命中率	Top-3 命中率	Top-10 命中率	Top-30 命中率
20%	0.2919	0.5338	0.8299	0.9794
40%	0.4194	0.7033	0.9253	0.9924
60%	0.5712	0.8515	0.9737	0.9973

表 2-13 进一步说明 Top-30 候选区域设计的有效性。全量数据中，20% 前缀下 Top-30 命中率达到 0.9794，40% 和 60% 前缀下分别达到 0.9924 和 0.9973。真实终点大多数都被包含在候选区域内，后续偏移回归可以在较可靠的候选区域中完成终点修正。

综合样本实验和全量实验，最终模型 ClusterTop30OffsetXGBoostDeep 具有更稳定的预测效果。它相较于直接经纬度回归的 XGBoostDeep，在 40% 和 60% 前缀下优势更明显；在 20% 前缀下提升较小，但仍保持最低平均误差。实验结果说明，终点预测任务不仅可以看作普通回归问题，也可以利用终点空间聚类结构，将预测过程拆分为候选区域判断和类内偏移回归两个阶段。

3. 课程的学习体会与建议

3.1 程序中存在的问题与局限性

本实验完成了从数据清洗、特征构造、模型比较到全量验证的流程。最终采用的 ClusterTop30OffsetXGBoostDeep 在样本实验和全量实验中都优于当前点基线，也优于直接回归的 XGBoostDeep。但这个结果并不说明程序已经解决了终点预测中的所有问题。由于出租车轨迹本身存在不确定性，模型仍然有一些明显限制。

数据清洗部分主要依靠规则判断。程序通过 MISSING_DATA 字段删除缺失轨迹，并用 Porto 附近的经纬度范围过滤异常 GPS 点，同时删除轨迹点数少于 10 的样本。这些规则可以去掉明显无效的数据，但不能处理所有异常情况。例如，有些轨迹虽然坐标没有越界，但可能存在 GPS 跳点、短时间大幅移动、路径绕行或中途截断。

特征工程主要基于轨迹本身，没有引入外部地理信息。程序提取了起点、当前点、前缀距离、平均速度、方向角、轨迹弯曲度和分段距离等特征，能够描述车辆已经走过的路径。但出租车目的地往往和道路结构、商业区、住宅区、交通枢纽等城市功能区域有关。当前程序没有使用道路网络、天气、节假日和实时交通信息。

评价指标也有一定局限。实验使用预测终点和真实终点之间的 Haversine 距离作为误差指标，适合衡量经纬度点之间的空间距离。但出租车实际行驶不是直线移动，而是受道路网络限制。两个点之间的直线距离较短，并不一定代表实际行驶距离或时间也短。后续可以加入道路最短路径距离、预计行驶时间误差等指标。

最终模型中的部分参数仍然带有经验性。实验中使用 KMeans 将终点划分为 50 个聚类区域，并保留 Top-30 候选聚类。但聚类数和候选数量并不是理论上唯一最优的选择。如果换成其他城市、其他时间范围的数据，或者清洗规则发生变化，这些参数可能需要重新实验。当前程序还没有实现自动搜索最优聚类数和候选数量。

模型在短前缀场景下仍然不够强。20% 前缀时，车辆刚开始行驶，目的地信息不足。全量实验中，ClusterTop30OffsetXGBoostDeep 相比 XGBoostDeep 的提升只有 0.03%，说明聚类方法在早期预测阶段优势并不明显。这个结果也说明，仅靠前 20% 轨迹点，很难稳定判断真实终点。模型在 40% 和 60% 前缀下表现更好，主要是因为车辆方向和目的地区域逐渐变得明确。

ClusterTop30OffsetXGBoostDeep 不只是训练一个回归模型，而是包含终点聚类、聚类分类、Top-30 候选区域选择和类内偏移回归。全量数据规模较大时，训练时间和内存占用都会增加。当前程序仅适合离线分析，无法用于实时预测。

3.2 有关课程的学习体会

之前参加过多次数学建模比赛，对一般的建模流程有一定了解。数学建模通常会从问题分析、数据处理、模型建立、结果检验和论文写作几个环节展开。这门课让我在原有建模经验的基础上，进一步接触了数据挖掘中的具体方法，尤其是机器学习模型、特征工程和评价指标的选择。

数据挖掘并不是简单地调用模型。一个完整任务通常要经过数据理解、数据

清洗、特征工程、模型训练和结果评价。本次出租车终点预测中，真正影响结果的不只是模型，还包括轨迹清洗、前缀构造和特征设计。原始轨迹不能直接用于训练，必须先解析 POLYLINE 字段，再根据轨迹长度、坐标范围和缺失情况进行筛选。

KNN、随机森林、ExtraTrees、XGBoost 等模型都可以用于预测，但它们的适用情况不同。KNN 更依赖特征空间中的距离关系，树模型更适合处理非线性特征，XGBoost 在多组实验中表现更稳定。通过这几次实验，可以更直观地看到不同模型在同一组特征上的差异。

课上提到的熵、信息增益和基尼指数，本质上都和“如何选择更有效的划分”有关。理解这些概念后，再看随机森林和 XGBoost，会更容易理解树模型为什么能从多维特征中学习规律。模型不是直接记住结果，而是在不断寻找更能区分样本的特征划分。

KMeans 在本次实验中也有比较实际的作用。它不只是用来画聚类图，而是被用于划分终点区域。先聚类，再做候选区域预测和类内偏移回归，可以把一个复杂的终点回归问题拆成两个更清楚的步骤。这也说明数据挖掘中的算法不是孤立使用的，不同方法可以结合起来解决一个具体问题。

评价指标的选择也很重要。信用卡欺诈任务不能只看 Accuracy，因为少数类样本很容易被忽略；租房价格和出租车费预测更适合使用 MAE、RMSE 等指标；本次终点预测则使用距离误差评价模型效果。指标必须和任务目标一致，否则结果看起来好，但不一定有实际意义。

数据挖掘更重视完整流程和结果解释。模型效果提升只是结果之一，更重要的是说明数据为什么要这样处理、特征为什么这样设计、最后为什么采用这个模型。本次课程中接触到的多个实验都体现了这一点：实验结论不能只看最终指标，还要和前面的数据分析、特征工程、模型对比对应起来。

3.3 有关课程的意见和建议

课程实验安排比较贴近实际数据任务。实验从数据清洗、数据分析、特征构造、模型训练到结果评价，流程比较完整。几次实验做下来，对数据挖掘的整体过程会更清楚，也能看出数据处理和特征设计对模型结果的影响。

开课时提到课程会有研讨环节，当时以为需要上台汇报，后来发现没有真正

安排课堂展示，也算虚惊一场。一个学期可以抽一节课做一次完整的数据挖掘案例讲解，从数据理解、清洗、特征工程、模型训练到结果分析完整走一遍。

实验部分可以提供统一的项目模板，例如数据目录、代码目录、输出目录和运行脚本。这样在完成较复杂的实验时，文件结构会更清楚，结果也更容易复现和整理。

参考文献：

[1] Chen T, Guestrin C. XGBoost: A scalable tree boosting system[C]//Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016: 785-794.

[2] Breiman L. Random forests[J]. Machine Learning, 2001, 45(1): 5-32.

[3] Cover T M, Hart P E. Nearest neighbor pattern classification[J]. IEEE Transactions on Information Theory, 1967, 13(1): 21-27.

[4] Geurts P, Ernst D, Wehenkel L. Extremely randomized trees[J]. Machine Learning, 2006, 63(1): 3-42.

[5] MacQueen J. Some methods for classification and analysis of multivariate observations[C]//Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. 1967, 1: 281-297.

[6] Pedregosa F, Varoquaux G, Gramfort A, et al. Scikit-learn: Machine learning in Python[J]. Journal of Machine Learning Research, 2011, 12: 2825-2830.

[7] Monreale A, Pinelli F, Trasarti R, Giannotti F. WhereNext: a location predictor on trajectory pattern mining[C]//Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2009: 637-646.

[8] Yuan J, Zheng Y, Zhang C, et al. T-Drive: driving directions based on taxi trajectories[C]//Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems. 2010: 99-108.

[9] Besse P C, Guillouet B, Loubes J M, Royer F. Destination prediction by trajectory distribution-based model[J]. IEEE Transactions on Intelligent Transportation Systems, 2018, 19(8): 2470-2481.

[10] Rossi A, Barlacchi G, Bianchini M, Lepri B. Modelling taxi drivers' behaviour for the next destination prediction[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 21(7): 2980-2989.

[11] Zhang L, Zhang G, Liang Z, Ozioko E F. Multi-features taxi destination prediction with frequency domain processing[J]. PLOS ONE, 2018, 13(3): e0194629.

[12] Tang J, Liang J, Yu T, Xiong Y, Zeng G. Trip destination prediction based on a deep integration network by fusing multiple features from taxi trajectories[J]. IET Intelligent Transport Systems, 2021, 15(9): 1131-1141.