智能算法赛：智慧海洋建设

Champion Chasing Boy

2020/3/16

Contents

1 前言

海上安全治理是海洋发展中至关重要的环节，了解各个区域船只的工作情况以及具体位置，可以对于防止因为船只的碰撞等事故而造成的巨大损失，而要提升海上安全治理能力，首要任务是“看得清”，即看得清“是什么、谁在用、做什么”。船舶避碰终端（AIS）、北斗定位终端等通信导航设备的应用，给海上交通和作业带来了极大便利，但同时存在设备信息使用不规范造成的巨大人身和财产损失，给海上安全治理带来了新的挑战。

本次赛题基于位置数据对海上目标进行智能识别和作业行为分析，要求选手通过分析渔船北斗设备位置数据，得出该船的生产作业行为，具体判断出是拖网作业、围网作业还是流刺网作业。

在下面的章节，我们团队会针对“智能算法赛：智慧海洋建设”赛题的算法部分进行详细地阐述介，文章的内容包括赛题背景介绍，数据探索分析，数据建模，其中数据建模部分我们分为特征工程和模型选择训练测试，以及预测结果分析和改进等，具体地，本篇文章会分成下面几个部分，

- 赛题解析，在这一部分，我们会介绍本次比赛的赛题背景，比赛的规则，包括赛题的数据，评测指标，提交格式等；然后我们对此类问题存在的难点以及需要注意的事项进行思考分析，为了更好的解释后续建模的特征，此处我们补充上了关于刺网，拖网和围网的一些背景知识；
- 数据探索分析，在该部分我们对赛题的数据进行细致的分析，包括数据的整体观察、数据集中每个变量的观测以及变量与标签的关系分析等；通过对这些信息的分析，我们可以更好地了解数据的质量，例如数据集中缺失的情况，是否出现了某些异常的值，例如速度达到 1000 之类的，是否存在某些特殊的船只等；为后续的建模带来更好地参考。
- 算法建模，在该部分，我们将会阐述此次赛题中我们团队所采用的特征工程的框架，模型的选择，包括各个模型的比较以及 N 折训练中 N 的选取，以及各个特征变量的观测等；
- 其他尝试，在该部分，我们会总结整个赛题建模过程中，我们团队做的一些其他探索尝试，这些探索尝试包括一些成功的尝试，例如最优权重的探索以及基于经纬度不同个数的建模等，同时我们也会补充上我们团队在建模过程中的一些失败的探索；
- 总结，在该部分，我们会对我们团队的建模方案做最终的总结与展望。

2 赛题解析

在本章节，我们首先对本次赛题的内容进行回顾，包括本次比赛的应用场景，赛题的意义；比赛规则的内容（该比赛已经结束，此处为了文章的完整性将其加入）以及赛题中我们所能获得的数据；然后我们对此类问题中需要注意的细节以及对应的难点进行简要的分析与阐述。

2.1 赛题细节

2.1.1 赛题背景

本赛题基于位置数据对海上目标进行智能识别和作业行为分析，要求选手通过分析渔船北斗设备位置数据，得出该船的生产作业行为，具体判断出是拖网作业、围网作业还是流刺网作业。初赛将提供 11000 条 (其中 7000 条训练数据、2000 条 testA、2000 条 testB) 渔船轨迹北斗数据。

复赛考虑以往渔船在海上作业时主要依赖 AIS 数据，北斗相比 AIS 数据，数据上报频率和数据质量均低于 AIS 数据，因此复赛拟加入 AIS 轨迹数据辅助北斗数据更好的做渔船类型识别，其中 AIS 数据与北斗数据的匹配需选手自行实现，具体细节复赛开赛时更新。同时，希望选手通过数据可视化与分析，挖掘更多海洋通信导航设备的应用价值。

2.1.2 数据说明

初赛提供 11000 条渔船北斗数据，数据包含脱敏后的渔船 ID、经纬度坐标、上报时间、速度、航向信息，由于真实场景下海上环境复杂，经常出现信号丢失，**设备故障等原因导致的上报坐标错误、上报数据丢失、甚至有些设备疯狂上报**等。

数据示例：

渔船 ID	x	y	速度	方向	time	type
1102	6283649.656204367	5284013.963699763	3	12.1	0921 09:00	围网

其中每个字段对应的含义如下，

- 渔船 ID：渔船的唯一识别，结果文件以此 ID 为标示
- x: 渔船在平面坐标系的 x 轴坐标
- y: 渔船在平面坐标系的 y 轴坐标
- 速度：渔船当前时刻航速，单位节
- 方向：渔船当前时刻航首向，单位度
- time：数据上报时刻，单位月日时：分
- type：渔船 label，作业类型

原始数据经过脱敏处理，渔船信息被隐去，坐标等信息精度和位置被转换偏移。

选手可通过学习围网、刺网、拖网等专业知识辅助大赛数据处理。

2.1.3 提交格式

本次大赛要求选手通过使用主办方提供的数据，训练算法来预测测试集中的渔船作业类型，渔船 ID 和作业类型一一对应生成 csv 文件提交评测。

提交文件示例（result.csv）：注：为避免评测过程中文编码格式导致评测错误的影响，请统一以 utf-8 编码，并且注意**不要包含表头**，type 需包含三种类型

```
1901,围网
1902,刺网
```

Figure 2.1: 提交格式

2.1.4 评分指标

选手提交结果与实际渔船作业类型结果进行对比，以 3 种类别的各自 $F1$ 值取平均做为评价指标，结果越大越好，具体计算公式如下： 其中 P 为某类别的准确率，R 为某类别的召回率，评测程序

$$Score = \frac{F1_{围网} + F1_{刺网} + F1_{拖网}}{3}$$

$$F1 = \frac{2 * P * R}{P + R}$$

Figure 2.2: 提交格式

f1 函数为 sklearn.metrics.f1_score，average='macro'。

2.2 相关专业知识

2.2.1 海里，节，公里

网上的文献中经常会出现海里，节和公里的字段，其对应的关系如下，

- 1 节等于每小时 1 海里，也就是每小时行驶 1.852 千米（公里）。

2.2.2 刺网，围网，拖网

2.2.2.1 围网

- **简介**：围网渔船是利用围网捕鱼法来捕捞鱼类的船舶。围网渔船大都是木船，平底方尾，航速较快机动灵活，横向稳定性好。围网渔船作业时有双船围网、单船围网两种。双船操作时，两渔船各带住网的一端，相背绕鱼群航行一周，鱼群即被网围住；单船作业时，先放下与网一端相连的舢板，**然后带住网的另一端航行一周，至舢板处收网**。围网渔船船长较短，吃水较小，

干舷较低，并具良好的操纵性。随着网具向大型化发展，需要较高的主桅及吊杆，加之起网时渔捞人员又都聚集在一舷，所以对稳性有较高的要求。

- **定义**：从事围网作业，主要围捕中、上层水域鱼类的专用渔船。包括双船及单船、舷侧起网及尾部起网灯光诱鱼、金枪鱼等围网渔船。
- **特点**: 为了追越鱼群达到围捕的目的，**围网渔船应具有一定的航速**。当收到某海区发现密集鱼群的信息时，需要快速驶向中心渔场。金枪鱼的游速较快，因此捕捞金枪鱼鱼群时，围网渔船航速要**达到 13 节以上**。应该指出，网船围捕的航速并非自由航速。因为船舶**从直线运动转为圆周运动时**，船体阻力增加同时网具下水时产生了对船体的摩擦力。由于以上原因使船舶减速 35% 左右。

2.2.2.2 拖网

- **简介**：拖网渔船是指利用拖网进行捕捞的渔船的统称。拖网的种类很多，最常见的是底拖网：渔网在海底被拖曳，地势平坦，鱼群密集，有利于捕鱼。中层拖网也称变水拖网，是指拖网曳行时，网具不在海底而处于海水中层。
- **定义**：拖曳网具进行捕捞作业的渔船；从事拖网作业，捕捞中、下层水域鱼虾类的专用渔船。分双拖和单拖渔船，舷拖、尾拖和桁拖渔船；从事拖网作业，捕捞水域底层及中、下层鱼虾等水产物的渔船；用拖曳袋形网具捕捞水域底层或中、下层鱼类及甲壳类等的渔船。
- **工作方式**: 一条圆锥形的大网，在其口端由巨型金属板（门板）固定来控制网口大小。在水中，随着船体的拖曳，会将一路上的所有东西卷入其中。主要可分为底层拖网和中层拖网。这种捕捞方式主要用于大规模捕捞，如阿拉斯加狭鳕及大多数鲆鲽鱼类。最大的缺点在于拖曳过程中可能会对底层生态造成影响。
- **特点**: 拖力较大，一般为 3 到 5 吨，，大型拖网渔船可达 30 吨。**拖网速度一般为 3 到 6 海里/小时**，**自由航速为 10 到 14 海里/小时**。主机功率大，因拖网捕鱼时需要很大的拖力，同时往返渔港也需要较高的航速。

2.2.2.3 刺网

- **简介**：流网（driftnet），很多情况下也叫刺流网（Drifting Gillnet），是一种捕鱼手段。简单说是用船拖曳一张垂直的大网，处于海洋中的一定深度，然后当鱼冲撞到网上时，鱼的鳃盖以及鳍等被网所缠绕而困住，从而达到捕获的目的。英文名中的“drift”指的是它可以沿着海流运动，不受风的影响。而“Gillnet”则是以鱼鳃被网缠住而得名。

  流刺网是一种捕鱼方法，主要用于捕捞洄游性鱼类。流刺网使用一层或多层塑胶丝所织成的长方形网片，一般会将多张网片结合在一起，**上缘系多个海绵塑胶所制的浮子，下端配附铅制沉子，垂直张开设于接近海平面附近的位置，等待鱼类游入而被网目缠住**。**因为会随海流移动**，而且被缠住的鱼像刺一样挂在网上，故称“流刺网”。
- **分类**:
  1. **漂流刺网**。简称流网或流刺网。作业时网列和渔船相连而随风、流漂移，是刺网类中使用最广泛的作业方式。作业水层可任意选定
  2. 定置刺网。用桩或锚等固定网具于捕捞对象栖息的水层。
  3. 围刺网。以包围方式作业的刺网，用单片型结构的长列网具包围集群鱼类，辅以声响、击水等驱赶方法使鱼受惊乱窜而被捕获。一般用于浅水、礁石、静水或水草丛生水域。
  4. 拖刺网。以拖曳方式作业的刺网，通常用两船牵引一长列单片刺网拖曳前进，使鱼类刺入或缠挂在网上。

3 数据探索分析

数据探索分析是所有传统比赛都必须做的，通过数据探索我们可以知道我们的数据的类型，大小，每个字段的缺失情况，字段之间的相关情况，标签的分布等信息，使我们能对手上的数据有更好的了解；此外，通过数据探索还可以深层次地挖掘特征之间的潜在关系，很多强特的构建都来源于深入的数据探索，有时还能为我们的建模以及预测结果的后处理带来非常大的帮助。因为数据探索的方式每个人都不一样，此处我们先从数据信息到局部单变量；再从局部特征变量与标签，特征变量与特征变量展开分析；在该比赛中，我们从下面的几个角度出发对我们的数据进行探索分析：

1. **数据整体信息观察**：在该部分我们对数据集的整体信息进行观察分析，包括数据的类型，数据的缺失情况，数据的 nunique 分布等；
2. **单变量分析**：观察每个字段的分布情况，是否存在奇异值，是否分布有偏等；
3. **特征变量 & 标签的关系分析**：在该部分，我们重点分析特征变量与标签之间的关系，包括单个特征变量与标签之间的关系，双特征变量与标签之间的关系等，希望通过分析特征变量与标签的关系能帮助我们探索更深层次的特征与变量之间的关系。
4. **特征变量 & 特征变量的关系分析**：特征变量之间往往存在着冗余的情况，分析特征变量之间的关系可以帮助我们发现此类冗余情况，删除冗余的特征不仅可以帮助我们节省内存，同时也可以帮助我们进一步提升模型的预测准确率；不仅如此，很多情况下，某些特征会出现缺失的情况，而对于该特征的强相关特征进行分析，我们可以很好地预测出对应特征的缺失值，例如某个数据集性别信息缺失，但是我们发现该用户购物时，经常会购买一些女性用品，那么很大概率该用户是一个女生。

3.1 基础数据信息观察

3.1.1 训练集数据观察

本次比赛就一张核心的数据表，其格式如图3.1所示，

	渔船ID	lat	lon	速度	方向	time	type
0	25943	23.881	119.010	5.29	192	0906 20:42:05	拖网
1	25943	23.891	119.011	4.10	186	0906 20:33:00	拖网
2	25943	23.911	119.011	3.08	189	0906 20:12:34	拖网
3	25943	23.932	119.011	3.08	184	0906 19:52:13	拖网
4	25943	23.942	119.011	3.40	182	0906 19:42:08	拖网

Figure 3.1: 训练数据集

3.1.1.1 基础统计信息：参见图3.2

1. 训练集的大小为 261.1+MB；
2. 其中有 2 个 int 型数据；3 个 float 型数据以及 2 个 object 型数据；

3. 所有数据有 4889341 个记录;

```
RangeIndex: 4889341 entries, 0 to 4889340
Data columns (total 7 columns):
渔船ID    int64
lat     float64
lon     float64
速度      float64
方向      int64
time    object
type    object
dtypes: float64(3), int64(2), object(2)
memory usage: 261.1+ MB
```

Figure 3.2: 训练集基础统计信息

3.1.1.2 数据缺失情况：参见图3.3

1. 数据集中不存在缺失情况。

3.1.1.3 字段 Nunique 分布：参见图3.3

1. 渔船 ID 有 8166 个不同的值;
2. lat 有 20715 个不同的值;
3. lon 有 19132 个不同的值;
4. 速度有 294 个不同的值;
5. 方向有 361 个不同的值;
6. time 有 2785302K 个不同的值;
7. type 有 3 个不同的值;

```
渔船ID    0
x       0
y       0
速度      0
方向      0
time    0
type    0
dtype: int64
```

(a) 字段缺失信息

```
渔船ID       7000
x         1064269
y         1059948
速度            248
方向            361
time      1472312
type            3
dtype: int64
```

(b) 字段 Nunique 分布

Figure 3.3: 训练集字段缺失 & Nunique 分布

3.1.1.4 数据整体分布观察：参见图3.4

1. **经纬度的最小值都是 0**, 有些奇怪, 可能是缺失值填充了, 最大值看上去问题不大;
2. 速度和方向相对也较为合理;
3. 渔船的 ID 变化了, 最小值是 20000, 最大值是 28165。

	渔船ID	lat	lon	速度	方向
count	4.889341e+06	4.889341e+06	4.889341e+06	4.889341e+06	4.889341e+06
mean	2.408578e+04	2.517322e+01	1.192768e+02	2.188371e+00	1.267817e+02
std	2.359798e+03	2.422063e+00	3.684982e+00	2.666564e+00	1.166772e+02
min	2.000000e+04	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	2.204300e+04	2.362100e+01	1.179840e+02	1.100000e-01	0.000000e+00
50%	2.407600e+04	2.486100e+01	1.191210e+02	8.100000e-01	1.010000e+02
75%	2.613300e+04	2.637700e+01	1.204330e+02	3.670000e+00	2.290000e+02
max	2.816500e+04	4.484200e+01	1.278030e+02	9.423000e+01	3.600000e+02

Figure 3.4: 训练集整体分布

3.1.2 小结

从上面的分析中，我们发现本次竞赛的训练数据集中不存在数据缺失的情况，整体数据量也不是非常大，除了经纬度都出现了为 0 的情况，后续需要进一步研究原因，并未发现其它不合理的情况。

3.2 单变量分析

在基础数据观察部分，我们并未发现明显的异常或者不合理的地方，在单变量分析部分，我们会重点观察每个变量的分布，进一步判断数据的质量，研究是否存在一些奇怪的地方。

3.2.1 标签 type

此处我们将标签进行转换，0 对应拖网，1 对应围网，2 对应刺网，通过观察3.5，我们发现：

- 三种网的分布没有太大的差异，基本类似;
- 拖网的数量相对最多，围网和刺网相差不大。

3.2.2 lat 的分布

我们绘制训练集中纬度 lat 的分布，具体地，如图3.6所示，我们发现：

- lat 的大部分值都分布在 25 附近, 存在少部分值分布在两端;
- 0 的过渡稍有突兀, 但也存在一些小于 10 的值, 整体看没太大的问题.

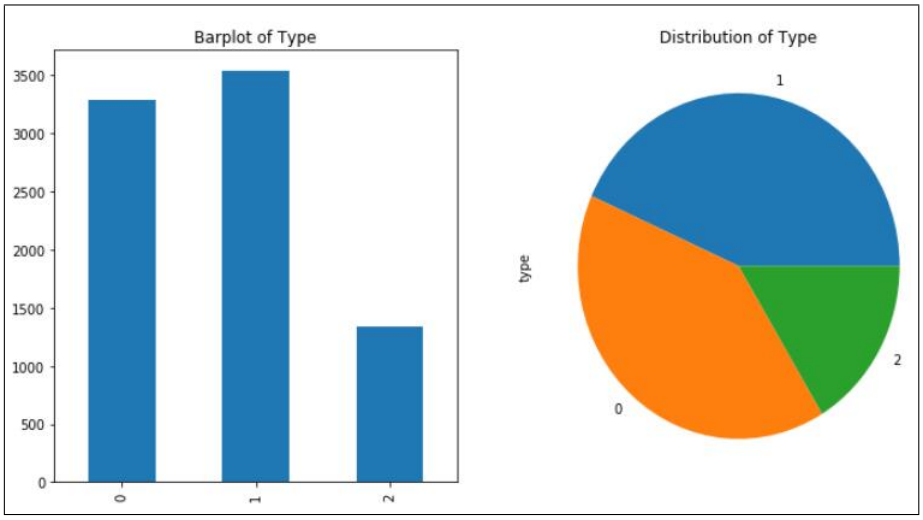

Figure 3.5: 训练集标签分布

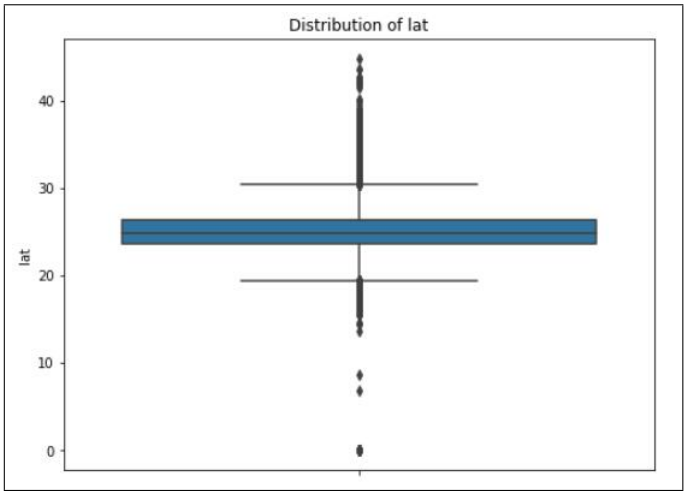

Figure 3.6: 训练集中 lat 的分布

3.2.3 lon 的分布

我们绘制训练集中经度 lon 的分布，具体地，如图3.7所示，我们发现：

- lon 的大部分值都分布在 120 附近；
- 存在少部分值分布在两端，0 看上去像是一个异常值，没有任何的过渡。

3.2.4 速度的分布

我们绘制训练集中速度的分布，具体地，如图3.8所示，我们发现：

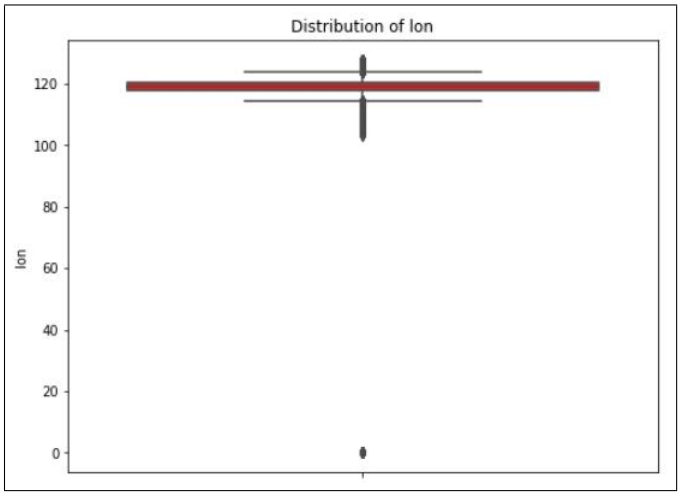

Figure 3.7: 训练集中 lon 的分布

- 训练集中速度的分布主要集中在 10 以下, 存在少数渔船速度超过了 80 的情况;
- 速度的分布没有出现突然断开跳跃的情况, 整体看较为合理;

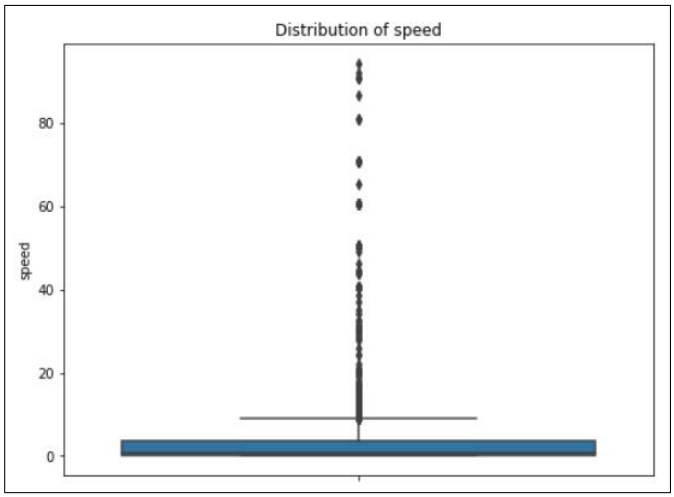

Figure 3.8: 训练集中速度的分布

3.2.5 方向的分布

我们先将方向平均分为 12 份，每份 30 度，然后再绘制训练集中方向的分布，具体地，如图3.9所示，我们发现：

- 训练集大多数的方向都在 0-30 之间,330-360 的最少;

- 在 31-330 之间的分布较为均匀。

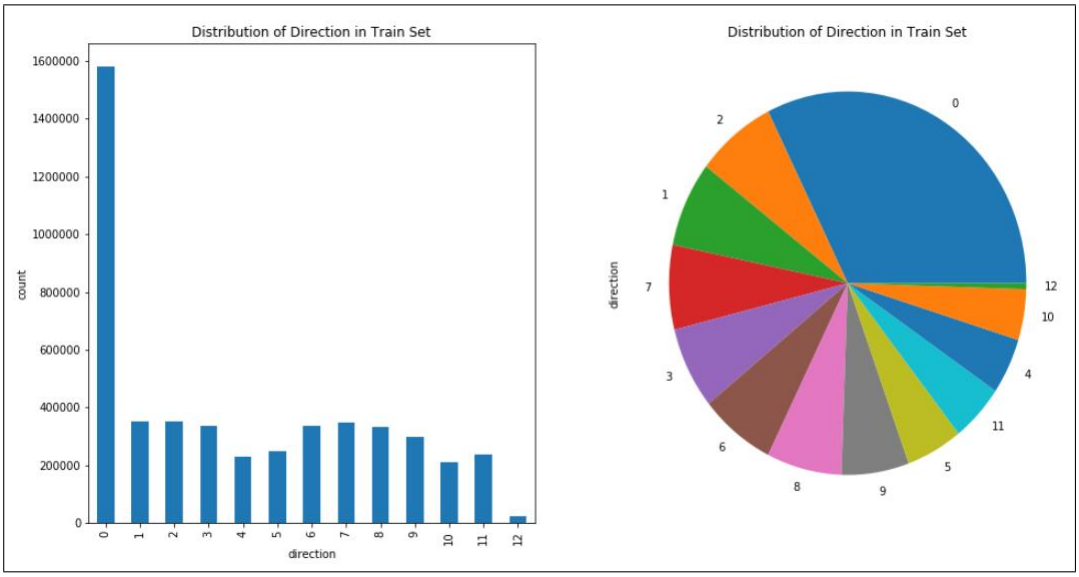

Figure 3.9: 训练集方向的分布

3.3 特征变量 & 标签的关系分析

3.3.1 坐标 lon, lat 与 label 的关系

我们分别随机抽取每种渔船的 8 个 ID 进行可视化，

3.3.1.1 拖网

拖网对应的坐标变化可以参考图3.10所示，我们发现：

- 拖网的坐标看上去有些乱，lon 的移动相较于 lat 要大一些。
- 可能因为拖网的关系，会明显出现几段直线；

3.3.1.2 围网

围网的可以参考图3.11所示，我们发现：

- 围网很多都有明显的画圆或者半圆的痕迹；
- 有些围网看上去像是快速转圈的情况；

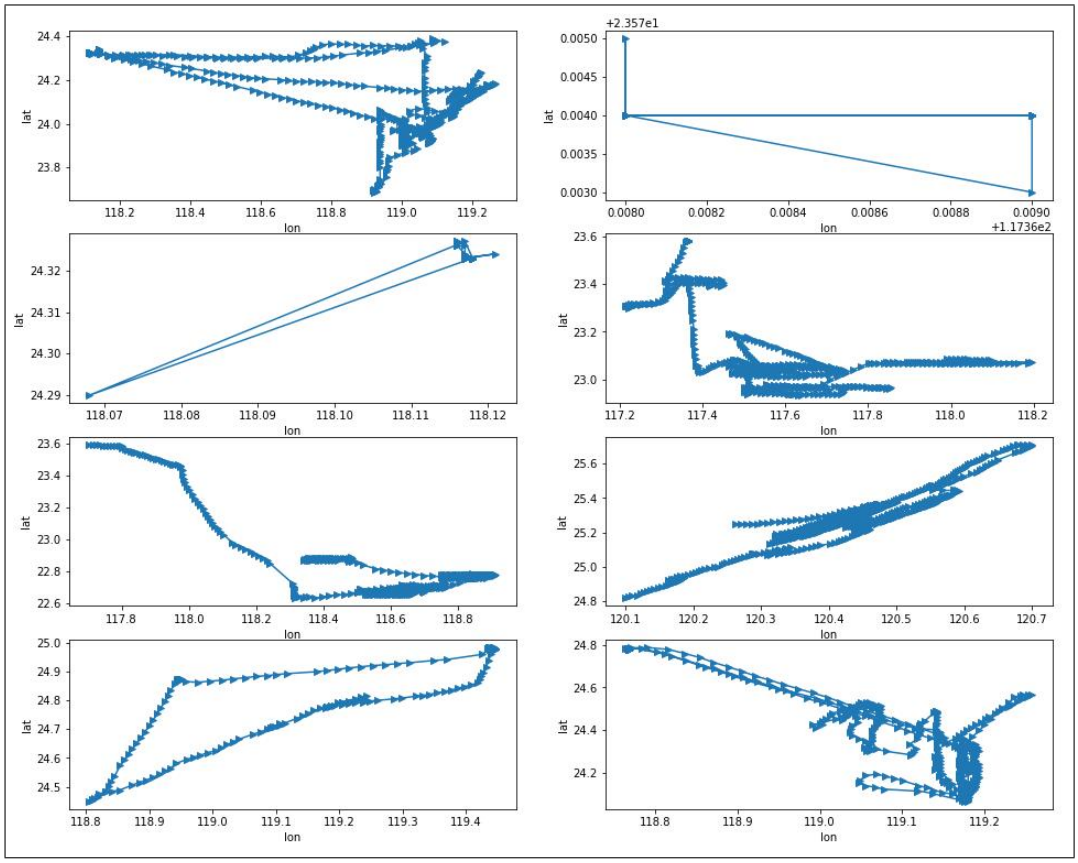

Figure 3.10: 拖网坐标变化情况

3.3.1.3 刺网

刺网的可以参考图3.12所示，我们发现：

- 刺网的线段看上去很多较为规范，很多看上去像是在很多地方放了很多网，然后船去收网的样子。
- 有些许刺网看上去和拖网相似，例如最左下角的图。

通过上面的观察，我们发现大部分围网渔船，刺网渔船和拖网渔船是有着较为明显的运动轨迹的，但也存在部分渔船的轨迹相似，可能只通过经纬度信息难以区分开。在起初的调研部分，我们发现不同渔船在捕鱼的时候速度和方向也存在较为明显的区别，例如围网渔船等会出现快速转圈撒网的情况等，下面我们再看看速度以及方向与标签之间的关系。

3.3.2 速度与 label 的关系

通过观察不同渔船的速度分布3.13，我们发现：

- 刺网渔船的速度分布都处于下层，不管是均值还是中位数还是其他分位数，刺网渔船都是最小的；

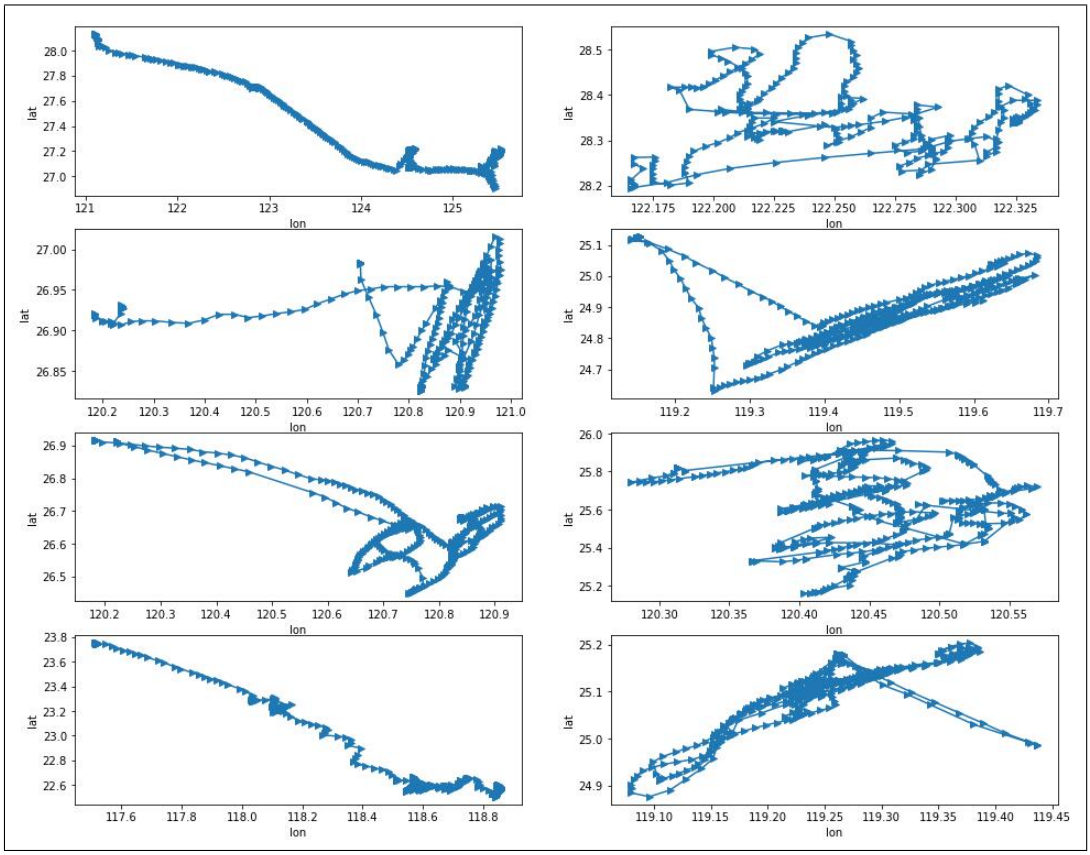

Figure 3.11: 围网坐标变化情况

- 在 0.5 分位数到 0.8 分位数之间，拖网的速度要略高于围网；
- 在 0.9 分位数的时候，围网的速度要高于拖网。

我们发现在速度特征这块，不同类型的渔船的差异也会相对较大。

3.3.3 方向变化与 label 的关系

有些渔船，例如围网渔船在捕鱼的时候可能需要快速调整方向，所以渔船的方向变化可能也能反映渔船的类型，不同渔船的方向变化的分布可以参见3.14，因为渔船的变向可能会有正负数，我们全部将其转换为绝对值用来表示渔船转向的大小，注意这边会有一些错误，例如上一时刻渔船的方向是 350，下一时刻是 10，我们其实是不知道渔船是正向转了还是方向转了，此处我们忽略此类信息，通过分析，我们发现：

- 刺网调整方向的分位数最高；
- 拖网调整方向的分位数是最低的；

对于方向的变化，我们很惊讶地发现刺网调整方向（相邻方向的差的绝对值）的分位数是最大的，而拖网是最低的。

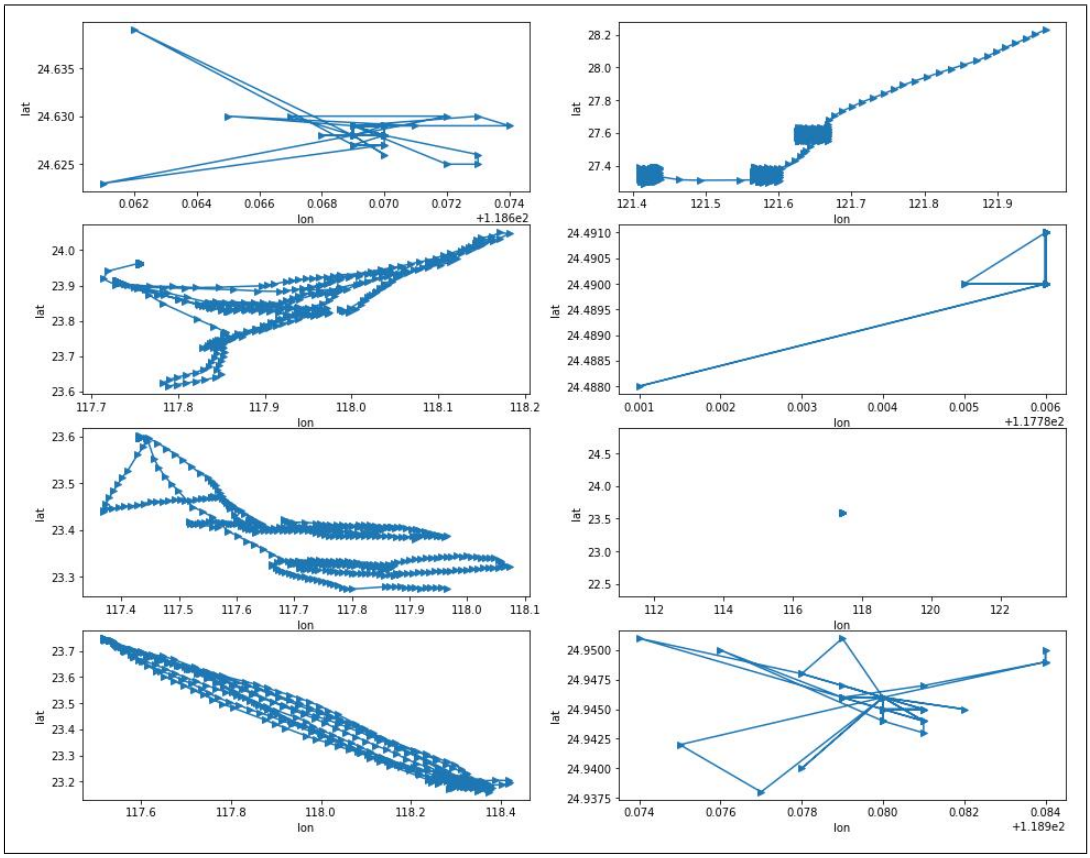

Figure 3.12: 刺网坐标变化情况

type	mean	median	quantile_60	quantile_70	quantile_80	quantile_90
刺网	1.153828	0.22	0.32	0.32	1.67	3.99
围网	2.061048	0.38	1.30	2.81	3.78	7.07
拖网	2.761558	2.97	3.45	3.89	4.32	5.40

Figure 3.13: 速度与标签的关系

3.3.4 一些特殊情况的观察

在之前的分析观察中，我们发现存在一些经纬度为 0 的情况，我们将此类数据单独拿出来发现，发现只要经纬度若有其一为 0，那么该渔船就是刺网渔船，在训练集中经纬度为的 0 情况全部出现在

type	mean	median	quantile_60	quantile_70	quantile_80	quantile_90
刺网	85.933658	33.0	71.0	123.0	181.0	264.0
围网	69.099488	19.0	35.0	74.0	143.0	239.0
拖网	54.595538	14.0	22.0	40.0	91.0	197.0

Figure 3.14: 方向变化与标签的关系

刺网渔船中。

4 算法建模

在这一部分，我们会详细地介绍我们团队的解决方案，具体地，可以分为下面几个模块：

1. 特征工程模块：在此处我们会介绍完整的特征工程模块，我们的方案从单个属性到多个属性之间的关系来刻画渔船的行为；
2. 模型训练：此处我们会介绍模型的选择，对比，模型的参数以及 N 折训练的 N 的选择；
3. 其他尝试：此处我们会介绍在本次比赛中，我们团队所进行的其他的尝试，包括一些有效的和无效的尝试等；

4.1 特征工程

我们的特征工程主要由单个属性刻画以及多种不同属性关系刻画两部分，其中单属性刻画主要是基于单个属性，例如速度，或者经度等来对我们的渔船进行行为的刻画；而多种不同属性关系刻画则是从多个属性的角度，例如经纬度，速度方向等来对我们的渔船的行为进行刻画。

4.1.1 单属性刻画

4.1.1.1 速度相关的特征

1. **传统的渔船速度的分布信息 1**，包括：
   - quantile(0.05),quantile(0.95),
   - quantile(0.1,0.2,...,0.9)
   - max,min
   - max - min, quantile(0.95) - quantile(0.05)
   - skew,mean
2. **渔船的速度分布信息 2，基于 qcut 的 bin 操作**，此处我们不再计算速度的位置的值，而是统计每个速度区间内渔船的行驶的次数，具体地可以参考下面的代码，

```
df['speed_bin']      = pd.qcut(df['速度'], 100, duplicates='drop').
    apply(lambda x:x.left).astype(float)

df['speed_bin_cnt'] = df.groupby(['speed_bin','渔船ID'])['速度'].
    transform('count')

df_speed_fea         = pd.pivot_table(data= df, values='speed_bin_cnt
    ',columns='speed_bin', index='渔船ID', fill_value=0 ).
    reset_index()
```

3. **渔船的速度分布信息 3，基于 cut 的 bin 操作**，和 qcut 类似，这边我们用整除替代了 cut，具体地可以参考下面的代码，

```
df['speed_bin2']      =  df['速度'].apply(lambda x: min(x,50))
```

```
df['speed_bin2']      =  df['speed_bin2'] // 1

df['speed_bin2_cnt'] = df.groupby(['speed_bin2','渔船ID'])['速度'].
    transform('count')

df_speed_fea          = pd.pivot_table(data= df, values='
    speed_bin2_cnt',columns='speed_bin2', index='渔船ID', fill_value
    =0 ).reset_index()
```

4. **渔船行驶中最常行驶的速度范围**：
   - 对渔船的速度分布信息 2,3 中的 bin 统计次数，并以次数最大的 top3 个对应的 index 作为渔船行驶中最常行驶的速度范围；

   因为 topN 个众数的计算比较耗时，所以大家可以一起算了，下面提供我们团队的计算函数供大家参考，

```
def mode123(x):
        t = x.value_counts()
        mode123_ind_vals = []
        if len(t) >= 3:
                mode123_ind_vals.append([t.index[0], t.index[1], t.
                    index[2],t.values[0], t.values[1], t.values[2]])
        else:
                if len(t) == 0:
                        mode123_ind_vals.append([-1,-1,-1,0,0,0])
                elif len(t) == 1:
                        mode123_ind_vals.append([t.index[0],-1,-1,t.
                            values[0],0,0])
                elif len(t) == 2:
                        mode123_ind_vals.append([t.index[0], t.index
                            [1],-1,t.values[0], t.values[1],0])
return mode123_ind_vals
```

5. **渔船以某种速度连续航行的次数**：例如拖网在捕鱼的时候需要以某一种速度连续航行一段时间，
   - 按时间排序，然后计算渔船在某个速度范围内连续移动的次数，例如速度以 [3,10] 连续航行的次数等；
   - 此处我们将硬间隔的计算方式改为软间隔，这样可以防止因为信号传输失败等带来的问题；

   对应的函数如下：

```
def get_soft_series_num(x, low, up = 999999, stops_ = [3,5]):
        x_series = (x >= low) & (x <= up)
```

```
        len_          = len(stops_)
        max_seq_lens,cur_seq_lens,thresholds        = np.ones((len_,)
            ), np.zeros((len_,)), np.zeros((len_,))

        for i,s in enumerate(x_series):
                if s > 0:
                        cur_seq_lens += 1
                else:
                        for t in range(len_):
                                thresholds[t]  = thresholds[t] + 1
                                if thresholds[t]  >= stops_[t]:
                                        thresholds[t]    = 0
                                        max_seq_lens[t] = max(
                                            max_seq_lens[t] ,
                                            cur_seq_lens[t])
                                        cur_seq_lens[t] = 0
        return max_seq_lens
```

6. **过滤了速度小于 0.2 的速度重新计算速度的第一种分布特征**：我们发现速度中存在较多的 0，这些值很有可能是渔船停泊之后在浪下传出的速度信息，会给很多分位数的计算带来噪音，所以我们取速度大于 0.2 的数据，也就是渔船在航行或者捕捞时的速度的信息。

4.1.1.2 方向相关的特征

1. **渔船在不同方向的行驶次数**：方向的分布特征，direction_bin(pd.cut) 的 pivot count 特征，
   - 方向在 [0,15],[15,30]....,[345,360] 的出现次数

   具体地，可以参考下面的代码，

```
df['direction_bin']      =  df['方向'] // 15

df['direction_bin_cnt'] = df.groupby(['direction_bin','渔船ID'])['速
    度'].transform('count')

df_direction_fea         = pd.pivot_table(data= df, values='
    direction_bin_cnt',columns='direction_bin', index='渔船ID',
    fill_value=0 ).reset_index()
```

2. **渔船航向的变化的分布**：渔船相邻的航向的差的分布，
   - quantile(0.1)，quantile(0.2)，...，quantile(0.9)
3. **渔船转向最多的角度**：
   - 计算渔船转向最多的角度的范围的 hash 值；
4. **渔船行驶最多的角度**：
   - 计算渔船行驶最多的角度的范围的 hash 值；

4.1.1.3 lon 相关的特征

1. **lon 的分布特征**：包括：
   - quantile(0.05),quantile(0.95),
   - quantile(0.1,0.2,...,0.9)
   - max,min
   - max - min, quantile(0.95) - quantile(0.05)
2. **渔船的经度分布信息，基于 qcut 的 bin 操作**，此处我们不再计算 lon 的分位数的值，而是统计每个经度区间内渔船的行驶的次数，对应的代码如下，

```
df['lon_bin']      = pd.qcut(df['lon'], 1024, duplicates='drop').
    apply(lambda x:x.left).astype(float)
df['lon_bin_cnt'] = df.groupby(['lon_bin','渔船ID'])['速度'].
    transform('count')

df['lon_bin2']      = pd.cut(df['lon'], 1024, duplicates='drop').
    apply(lambda x:x.left).astype(float)
df['lon_bin2_cnt'] = df.groupby(['lon_bin2','渔船ID'])['速度'].
    transform('count')

df_lon_fea1 = pd.pivot_table(data= df, values='lon_bin_cnt',columns=
    'lon_bin', index='渔船ID', fill_value=0 ).reset_index()

df_lon_fea2 = pd.pivot_table(data= df, values='lon_bin2_cnt',columns
    ='lon_bin2', index='渔船ID', fill_value=0 ).reset_index()
```

3. **渔船行驶中最常经过的经度范围**：
   - 对渔船的经度分布信息中的每个 bins 统计次数，并以次数最大的 top3 个对应的 index 作为渔船行驶中最常行驶的经度区间；

   代码可以参考如下，为了从不同粒度对渔船的常工作的经度进行定位，我们切分了不同粒度的 bins，对应下面代码中的 *lon_bins_cols*，

```
main_key = '渔船ID'
for main_col_name in lon_bins_cols:
        dic = df_grp[main_col_name].apply(lambda x: mode123(x)).
            to_dict()
        df_lon_fea[main_col_name + '_series']      = df_lon_fea[
            main_key].map(dic).values
        df_lon_fea[main_col_name + '_mode_1']      = df_lon_fea[
            main_col_name + '_series'].apply(lambda x: x[0][0]).
            values
        df_lon_fea[main_col_name + '_mode_2']      = df_lon_fea[
            main_col_name + '_series'].apply(lambda x: x[0][1]).
            values
```

```
        df_lon_fea[main_col_name + '_mode_3']      = df_lon_fea[
            main_col_name + '_series'].apply(lambda x: x[0][2]).
            values
```

4. **不同经度区间船只的分布**，这个特征可以反映当前经度是不是许多渔船常工作区域，该步先计算每个经度区间内渔船出现的次数，然后基于渔船再对这些区间的分布计算统计量，此处我们的统计量主要是下面两种，
   - mean;
   - max;
5. **渔船航行的经度的聚类特征**，聚类的特征是通过计算经度的分位数得到，然后采用下面两种聚类方法进行聚类：
   - Kmeans 聚类;
   - DBSCAN 聚类;

4.1.1.4 lat 相关的特征

1. **lat 的分布特征**：包括：
   - quantile(0.05),quantile(0.95),
   - quantile(0.1,0.2,...,0.9)
   - max,min
   - max - min, quantile(0.95) - quantile(0.05)
2. **渔船的纬度分布信息，基于 qcut 的 bin 操作**，此处我们不再计算 lon 的分位数的值，而是统计每个纬度区间内渔船的行驶的次数，对应的代码如下，

```
df['lat_bin']      = pd.qcut(df['lat'], 1024, duplicates='drop').
    apply(lambda x:x.left).astype(float)
df['lat_bin_cnt'] =  df.groupby(['lat_bin','渔船ID'])['速度'].
    transform('count')

df['lat_bin2']     = pd.cut(df['lat'], 1024, duplicates='drop').
    apply(lambda x:x.left).astype(float)
df['lat_bin2_cnt'] =  df.groupby(['lat_bin2','渔船ID'])['速度'].
    transform('count')

df_lat_fea1 = pd.pivot_table(data= df, values='lat_bin_cnt',columns=
    'lat_bin', index='渔船ID', fill_value=0 ).reset_index()

df_lat_fea2 = pd.pivot_table(data= df, values='lat_bin2_cnt',columns
    ='lat_bin2', index='渔船ID', fill_value=0 ).reset_index()
```

3. **渔船行驶中最常经过的纬度范围**：
   - 对渔船的纬度分布信息中的每个 bins 统计次数，并以次数最大的 top3 个对应的 index 作为渔船行驶中最常行驶的纬度区间;

代码可以参考如下，为了从不同粒度对渔船的常工作的经度进行定位，我们切分了不同粒度的 bins，对应下面代码中的 *lat_bins_cols*，

```
main_key = '渔船ID'
for main_col_name in lat_bins_cols:
        dic = df_grp[main_col_name].apply(lambda x: mode123(x)).
            to_dict()
        df_lat_fea[main_col_name + '_series']      = df_lat_fea[
            main_key].map(dic).values
        df_lat_fea[main_col_name + '_mode_1']      = df_lat_fea[
            main_col_name + '_series'].apply(lambda x: x[0][0]).
            values
        df_lat_fea[main_col_name + '_mode_2']      = df_lat_fea[
            main_col_name + '_series'].apply(lambda x: x[0][1]).
            values
        df_lat_fea[main_col_name + '_mode_3']      = df_lat_fea[
            main_col_name + '_series'].apply(lambda x: x[0][2]).
            values
```

4. **不同纬度区间船只的分布**，这个特征可以反映当前纬度是不是许多渔船常工作区域，该步先计算每个纬度区间内渔船出现的次数，然后基于渔船再对这些区间的分布计算统计量，此处我们的统计量主要是下面两种，
   - mean;
   - max;
5. **渔船航行的纬度的聚类特征**，聚类的特征是通过计算纬度的分位数得到，然后采用下面两种聚类方法进行聚类：
   - Kmeans 聚类;
   - DBSCAN 聚类;

上面的特征基本都是对模型有帮助的，从不同的角度反映了渔船航行时的各种信息；但是还不够细，所以我们补充上第二块核心特征，用多个属性的关系来进一步刻画不同渔船的航行信息。

4.1.2 多属性关系刻画

在上面的部分，我们从单属性的角度对渔船的航行信息进行了刻画，在这一节，我们从多个属性的关系出发进一步细化上面的特征。

4.1.2.1 lat 和 lon 的关系刻画，细化渔船的路径信息

先将经纬度的图划分成不同的小块，采用类似于下图的思想4.1，

1. **描述渔船工作次数最多的区域**：寻找船只工作次数最多的区域，并返回 hash 的值;
2. **描述每个渔船 lat 上 lon 的分布以及每个 lon 上 lat 的分布**: 具体的代码参考如下：

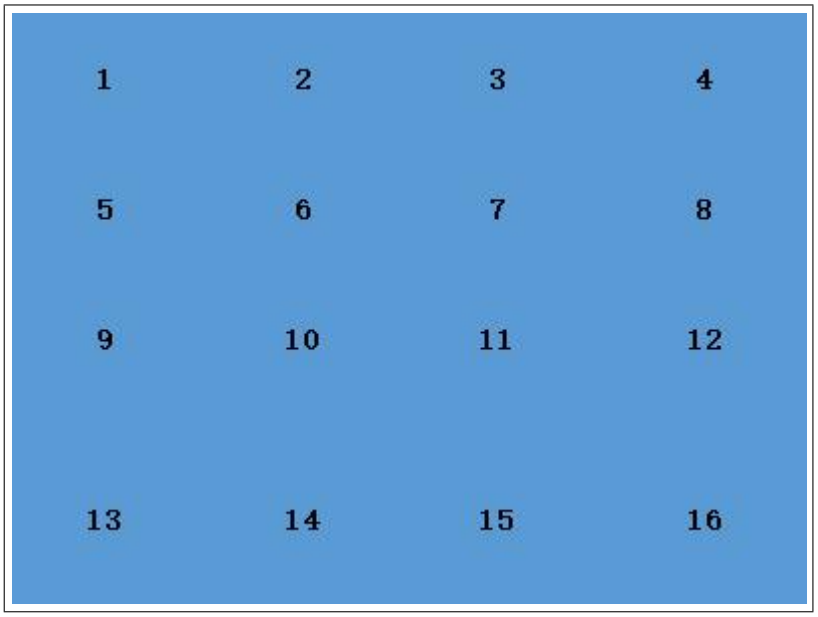

Figure 4.1: 经纬度 hash

```
df_lonbin_ymean    = df.groupby(['渔船ID','lon_bin'])['lat'].agg({'
    mean'}).reset_index()
df_lonbin_ymean.rename(columns = {"mean": "lat_tpt"},  inplace=True)
df_lonbin_ymean    = pd.pivot_table(data= df_lonbin_ymean, values='
    lat_tpt',columns='lon_bin', index='渔船ID', fill_value=0 ).
    reset_index()

df_latbin_xmean    = df.groupby(['渔船ID','lat_bin'])['lon'].agg({'
    mean'}).reset_index()
df_latbin_xmean.rename(columns = {"mean": "lon_tpt"},  inplace=True)
df_latbin_xmean    = pd.pivot_table(data= df_latbin_xmean, values='
    lon_tpt',columns='lat_bin', index='渔船ID', fill_value=0 ).
    reset_index()
```

3. **描述渔船的工作的范围**: 基于 lon,lat 计算近似面积;
4. **不同经纬度区间内船只的分布**，这个特征可以反映当前经纬对应的区间内渔船的频繁程度，该步先计算每个纬度区间内渔船出现的次数，然后基于渔船再对这些区间的分布计算统计量，和之前经度区间以及纬度区间一样，此处我们的统计量主要是下面两种，
   - mean;
   - max;
5. **渔船航行的经纬度的聚类特征**，聚类的特征是通过分别计算经纬度的分位数得到，然后采用下面两种聚类方法进行聚类:

- Kmeans 聚类;
- DBSCAN 聚类;

4.1.2.2 速度和 lat 的关系刻画

此处我们主要细化渔船的在不同速度时 lat 的位置分布信息，具体地，

```
df_speed_bin_lat    = df_tr_te.groupby(['渔船ID','speed_bin'])['lat'].
    agg({'mean'}).reset_index()
df_speed_bin_lat.rename(columns = {"mean": "lat_mean"},  inplace=True)
df_speed_bin_lat = pd.pivot_table(data= df_speed_bin_lat, values='
    lat_mean',columns='speed_bin', index='渔船ID', fill_value=0 ).
    reset_index()
```

4.1.2.3 速度和 lon 的关系刻画

和上面的类似，此处我们主要细化渔船的在不同速度时 lon 的位置分布信息，对应地代码如下，

```
df_speed_bin_lon = df_tr_te.groupby(['渔船ID','speed_bin'])['lon'].agg
    ({'mean'}).reset_index()
df_speed_bin_lon.rename(columns = {"mean": "lon_mean"},  inplace=True)
df_speed_bin_lon = pd.pivot_table(data= df_speed_bin_lon, values='
    lon_mean',columns='speed_bin', index='渔船ID', fill_value=0 ).
    reset_index()
```

4.1.2.4 速度和方向的关系刻画

此处我们主要计算在不同速度上方向转向变化的分布，对应地代码如下，

```
df['speed_direction_bin']       =  df['速度'].values * df['
    direction_diff_bin'].max() + df['direction_diff_bin'].values
df['speed_direction_bin']       =  pd.qcut(df['speed_direction_bin'],
    100, duplicates='drop').apply(lambda x:x.left).astype(float)
df['speed_direction_bin_cnt']   =  df.groupby(['speed_direction_bin','渔
    船ID'])['速度'].transform('count')

df_speed_direction_bin          = pd.pivot_table(data= df, values='
    speed_direction_bin_cnt',columns='speed_direction_bin', index='渔船
    ID', fill_value=0 ).reset_index()
```

在本章节，我们介绍了此次比赛我们团队所采用的特征工程的整体框架，在下一章节，我们会介绍模型相关的部分。

4.2 模型训练

在该部分，我们将会介绍我们模型的训练与测试部分，主要分为两块，第一块是模型选择的部分；第二块是 N 折训练中 N 的选择部分。

4.2.1 模型选择

在此次比赛中，我们选用了多种模型对数据进行训练测试，包括传统的 GBDT 类模型 CATBoost，XGBoost 以及 LightGBM，还有一些神经网络的尝试，在初赛的时候，NN 的表现相较于 GBDT 类的模型相差较大，所以最终我们选择 GBDT 类的模型进行尝试。

在我们线下的测试中，我们发现 CATBoost 和 XGBoost 的效果相较于 LightGBM 要差一些，我们尝试对三个模型进行融合，虽然线下有一定的提升，但是线上的效果变差了，最终我们选择线下和线上效果相对稳定的 LightGBM 模型，此次比赛中我们采用的 LightGBM 参数如下：

```
lgb_model = lgb.LGBMClassifier(boosting_type="gbdt", num_leaves=32,
    reg_alpha=0, reg_lambda=0.,metric="None",max_depth=-1, n_estimators
    =3000, objective='multiclass',class_weight='balanced',subsample
    =0.95, colsample_bytree=0.15, subsample_freq=1,min_child_samples=10,
    learning_rate=0.05, random_state=2020, n_jobs=20,)
```

采用上面的参数进行八折训练，用复赛的数据线下可以达到 0.936+，线上可以达到 0.90+；采用类似的方法对初赛的数据进行训练测试，线下八折可以达到 0.942+，线上可以达到 0.90+；相对较为稳定。

4.2.2 N 折训练之 N 选取

此次比赛我们发现 N 折训练 N 的选择对模型的训练预测影响较大，平时我们习惯采用五折，六折训练，但是这么做有一个较大的问题，线下的验证结果是没法比较的，也就是说，虽然六折的结果看上去比五折的好，但是我们却无法保证在线上我们模型六折的结果比五折的好，所以此处我们设计了一种寻找最优 N 的线下验证策略，验证框架如下：

上面算法部分的 NFold_Model 是传统的 N 折训练测试模型。使用上面的模型，我们线下验证时发现：

- 7-10 折的线下效果比 5 折要普遍好一些 (基本每折都会有提升)；
- 线下 7-10 折的效果会有些许波动，有的时候是 8 折好，有的时候 10 折好一些；

经过比较之后，最终我们选择采用 $N = 8$。

4.3 特征重要性观察

此处我们对模型的特征进行观察，我们将模型最重要的 20 个特征绘制出来，参考图4.2，通过观察，我们发现：

- 基于速度的 skew 特征是最重要的；
- 船只分布最多的经纬度区域的 hash 也非常靠前；
- 经纬度对应船只的统计特征以及速度的其他分布特征较为重要；

Algorithm 1 双层寻找最佳 N

Input: 训练集的特征 X，训练集的标签 y;
Output: 最佳 N

```
function FindBestN(X, y)
    best_N ← 5, best_score ← 0
    for N in [5, 6, 7, ...M] do
        for tr_ind, val_ind in KFold(N).split(X, y) do
            X_tr, y_tr, X_val, y_val = X[tr_ind], y[tr_ind], X[val_ind], y[val_ind]
            model = NFold_Model(X_tr, y_tr, Fold = N)
            pred = model.predict(X_val)
            score = score_function(pred, y_val)
            if score > best_score then
                best_score = score
                best_N = N
            end if
        end for
    end for
    return best_N
end function
```

从特征的重要性来看，上述的发现也都较为符合我们的直观感受。在这一章节，我们详细地介绍了我们此次竞赛的算法框架，包括特征工程的构建框架，各种算法的尝试以及 N 折训练模型中 N 的选择过程，最后我们对 LightGBM 模型的特征重要性进行了观察，发现基本符合我们的观察。

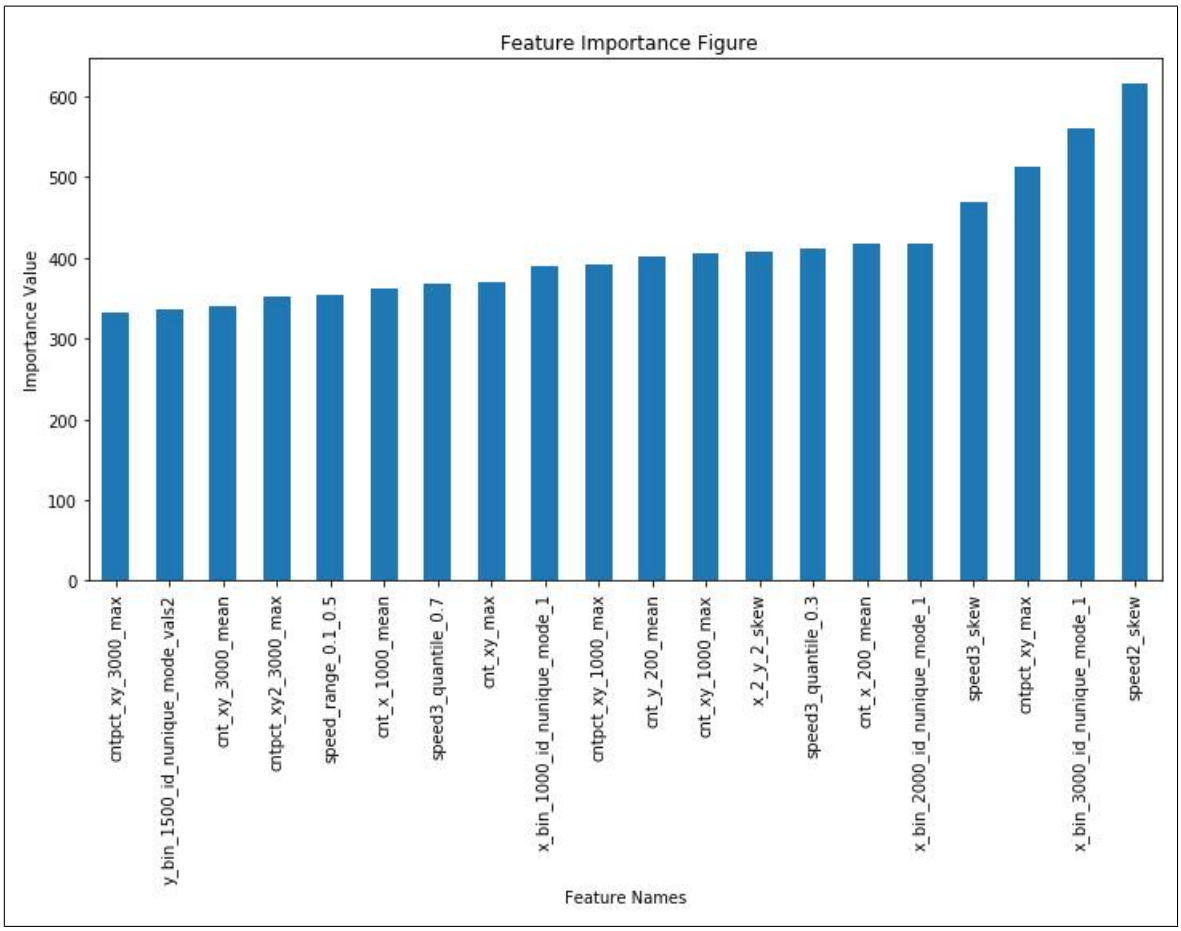

Figure 4.2: Top20 特征重要性

5 其他尝试

在算法建模部分，我们介绍了此次比赛我们的建模框架以及里面的一些细节，此外，我们还做了非常多其他的尝试，但是因为运行时间等原因，最终我们放弃了里面的一些有用的操作，此处我们对一些有价值的操作进行详细的介绍。

5.1 一些成功的尝试

此处我们重点介绍两种成功的操作，一种是最优权重搜索，如果模型训练是采用的原始的 Loss 训练，例如 MultiLogloss 之类，那么针对多分类的 Fscore，调整预测结果的概率矩阵能带来较大的帮助；第二种是基于特殊经纬度数据的预测；

5.1.1 最优权重搜索

在此次比赛中，我们前期使用了最优权重搜索的方法，具体地步骤如下，使用该方法对 MultiLogloss

Algorithm 2 单层权重搜索最优 w

Input: 验证集的预测 probability matrix(P), 验证集的 label(y)；
Output: 最佳权重 w；

function BestWeightSearch(P, y)
  $initialize w \in R^3 \leftarrow 0$
  for $round$ in $[1, 2, 3, ...M]$ **do**
    for $class$ in $[0, 1, 2]$ **do**
      Search the best weight of class i;
      Set old probability matrix into new one by multiplying weight of class i;
      Record the best weight of class i by multiply corresponding old w_i;
    end for
  end for
  return $w \in R^3$
end function

的结果进行线下调整寻找到最优权重 w，然后对预测概率矩阵进行调整，我们发现线下线上都有不错的提升，这种权重搜索的方案在数据标签分布不平衡的时候效果更佳明显，但在此次比赛中，我们在训练时对标签进行了加权，发现再寻找最优权重并不能再带来太大的提升，所以最终我们没有再进行权重的进一步调整。

5.1.2 基于特殊经纬度的建模

在此次比赛中，我们发现存在一些特殊的渔船，这些渔船的经纬度的不同个数较少，有些渔船可能就只有几个不同的经纬度，例如5.1，而这些渔船的航向和航速有时却会不一样，例如渔船 ID = 20010 的渔船，参见5.2，它的航向和速度会变化，但是更像是海浪带来的噪音。针对此类情况，我们选择对基于渔船出现的经纬度不同数来进行建模，具体地，我们分别构建两个模型，一个是只使用经纬度进行特征的提取，然后采用 LightGBM 训练得到模型 1；另外一个我们采用所有的特征列，除了经纬度之外，还包括速度、航向等信息，提取特征之后，我们再使用 LightGBM 训练得到模型 2，具体地可以参考图5.3，然后我们从测试集中选出经纬度出现的不同数都较少的情况，使用模型 1 进行预测；我们在验证的时候发现，经度的不同数小于 80，维度的不同数小于 80 的情况使

渔船ID	lat_nunique	lon_nunique	type
20010	1	1	2
20056	1	1	2
20221	1	1	2
20257	1	1	1
20301	1	1	0
20473	1	1	1
20555	1	1	1
20636	1	1	1
20699	1	1	2

Figure 5.1: 特殊的经纬度

渔船ID	lat	lon	速度	方向	time	type
20010	25.095	119.14	0.16	0	1020 02:05:47	刺网
20010	25.095	119.14	0.00	0	1020 01:55:47	刺网
20010	25.095	119.14	0.05	0	1020 01:45:48	刺网
20010	25.095	119.14	0.16	210	1020 01:35:48	刺网
20010	25.095	119.14	0.16	0	1020 01:25:48	刺网
20010	25.095	119.14	0.05	0	1020 01:15:48	刺网
20010	25.095	119.14	0.27	57	1020 01:05:48	刺网
20010	25.095	119.14	0.05	0	1020 00:55:48	刺网
20010	25.095	119.14	0.00	0	1020 00:45:48	刺网

Figure 5.2: 渔船 ID = 20010

用模型 1 进行预测；经度和纬度都大于 80 的使用模型 2 进行预测，线下验证可以提升 0.001 0.002 左右；线下八折里面有 6 折是提升的，剩下的 2 折波动也只变差了一点点。这种建模方式可以降低海浪带来的速度以及航向的噪音的影响，但最终考虑到时间的因素，需要训练两个模型，我们团队没有使用该方案。上面的这两种方法此次比赛中的一些有效的尝试，但是因为模型训练时间以及其他的替换方式存在，我们没有使用。在下面一节，我们将介绍此次比赛中，我们团队的其他的尝试，这些尝试有一些线下能带来不错的提升，但是线上没有明显的效果，此处我们将其列举在下一节，供大家参考。

5.2 其他探索

这边的探索包括一些特征的尝试以及模型融合部分的探索，我们将其列举在下方，

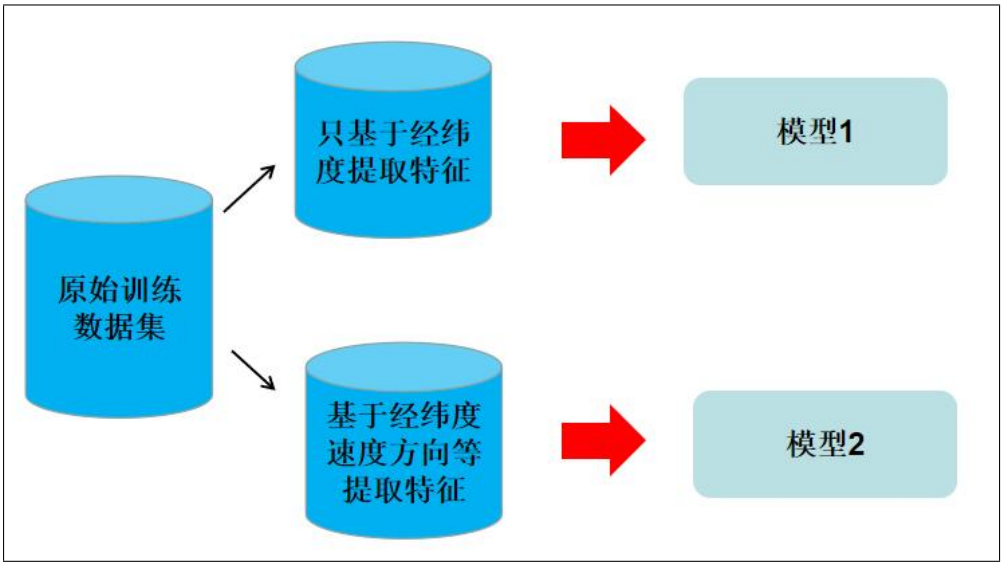

Figure 5.3: 两种建模方式

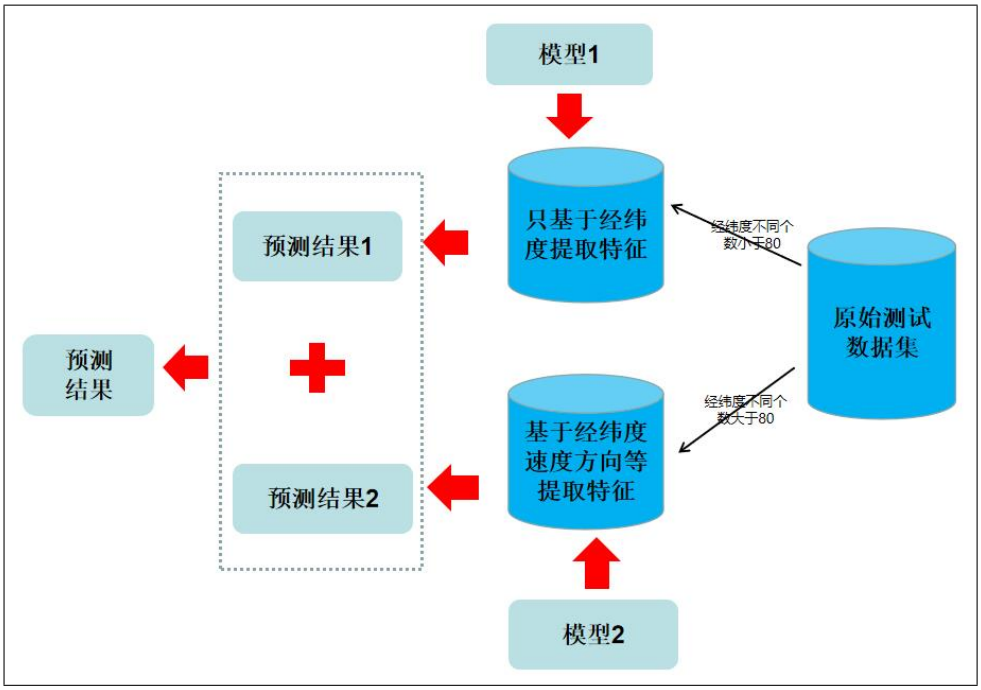

Figure 5.4: 模型预测

1. 基于时间戳的统计特征；
2. 基于时间差的统计特征；
3. 基于经纬度计算相邻位置的距离，然后基于这些距离构建统计特征；
4. 使用时间差与相邻速度的均值近似计算相邻时间戳内船的航行距离，然后基于此类距离构建统计特征；
5. 采用不同的聚类算法聚类，包括 KMedoids(距离选择 manhattan 距离以及 cosine 距离)；
6. 对经纬度进行一些数学操作，例如 x^2/y 等，然后再基于构建的新的特征列构建统计特征；
7. 对速度、方向、经纬度进行其他的组合特征构建；
8. 一些 NN 模型的尝试，包括 CNN，LSTM 模型等；
9. 对多个不同模型的预测结果进行 stacking；

在这一章节，我们汇总了我们团队的一些其他尝试，包括特征层面的以及模型层面；在下一章节，我们将对我们团队针对此次比赛的方案进行最后的总结。

6 总结

海上安全治理是海洋发展中至关重要的环节，了解各个区域船只的工作情况以及具体位置，可以对于防止因为船只的碰撞等事故而造成的巨大损失。在本次比赛中，大家需要基于渔船上传的经纬度、速度、时间以及航向等信息来对渔船的类型进行预测，判断其是否为拖网作业、围网作业还是流刺网作业。

对于该问题，我们团队将建模一共分为了四个模块，包括赛题解析，数据探索分析，算法建模以及其他尝试。在赛题解析部分，我们对比赛的赛题背景，比赛的规则，包括赛题的数据，评测指标，提交格式等进行了解读，并查阅相关资料，加上了关于刺网，拖网和围网的背景知识以便我们更好的熟悉了解该赛题；在数据探索分析部分，我们对赛题的数据进行了细致的探索分析，包括数据的整体观察、数据集中单个变量的观测以及变量与标签的关系分析等，通过对这些信息的分析，我们发现我们的数据集中并未出现缺失值，也没有出现明显的异常值等情况，数据质量相对较高；在算法建模部分，我们对特征工程框架进行了细致的描述，包括基于单属性的特征对渔船行为进行刻画，基于多属性关系对渔船行为进行刻画等，在模型选择部分，我们介绍了我们团队针对不同模型进行线下测试比较的情况、最终模型选择，以及 N 折训练验证 N 的选取等，并加上了特征重要性观察；在其他尝试部分，我们介绍了团队所做的一些其他探索尝试。

总体看来，我们的模型相对鲁棒，特征工程非常完善，构建逻辑也非常清晰，线下模型的尝试也较为丰富。在未来的工作中，我们会继续探索基于路径信息的一些其他方面尝试，例如挖掘更加深层次的轨迹速度方向的交叉特征，探索尝试基于轨迹的神经网络等。