2020

第四届长风杯全国大学生大数据分析与挖掘竞赛创新分析类（A 组）交通物流方向数据分析报告

作品名称：面向物流场景的数据分析报告

团队编号：NCDM0157

团队名称：黄金矿工

院校：北京邮电大学信息与通信工程学院

团队成员：罗浩 罗如瑜 丁博

指导教师：聂高峰

目录

项目简介

为了破解交通物流行业中存在的物流资源浪费问题，帮助网络货运平台降本增效，本项目通过数据分析、机器学习与数学建模的方法，对客户物流数据集和车辆基础数据集从运量走势、车货匹配、线路分布三个方面进行分析。在运量走势方面，通过整理客户历史运单数据得到给出提货、卸货地情况下的运量时间序列，使用 ARIMA 时间序列分析方法对客户未来的运输需求进行较好的预测。在车货匹配方面，使用数学建模与机器学习相结合的方法分析客户历史数据，对客户未来的运货需求提供车辆推荐。在线路分布方面，使用统计与数学建模相结合的方法，对客户过往运单的线路数据进行统计分析。在给出需求的情况下，参考图论模型推荐的最短路线，通过加权评分的方法给出一条推荐最优路线。本项目希望通过上述三个方面的预测或推荐，为网络货运平台的资源调配和线路优化提供指导性意见。

1. 物流领域问题场景

近年来，随着物流行业和车货匹配平台的快速发展，我国公路货运市场规模已超过 6 万亿。准确把握公路货运量等发展趋势，可以为相关决策提供依据。运量预测是指根据国民经济和社会发展对交通运输的需求，探索未来旅客和货物运量发展的趋势，对未来一定时期内交通运输业所应承担的工作量所做的测算和判断。是国家经济预测的组成部分之一，也是研究分析交通运输发展战略的重要内容和决策的依据。

运量预测的主要内容包括四个方面：社会总运输量的预测；各种运输方式的运输量预测；地区之间的运输量预测；运输企业在运输市场上的占有率预测。在四类预测中，前两类属于宏观预测的范畴，后两类属于微观预测的范畴。由于预测的目的要求不同，因此内容的粗细也不同。一般来讲，宏观预测与长期预测内容要粗一些，微观预测和短期预测的内容则要细一些。例如，列入本企业(或部门)经营的运输量，不仅有客、货运量和周转量，还应包括上行、下行的运输量，淡、旺季的运输量，货物运量中主要货物的分类和比重等。

中国物流的特点是大而复杂。在物流市场中，存在地区性差异和季节性差异，参与其中的角色众多：个体司机，车队老板，物流公司等等。而近年来我国公路运输不断涌现出资源浪费现象：车找不到货、货找不到车的情况在物流市场普遍存在。为了降低物流成本，减少不必要的社会资源浪费，许多企业在互联网的大时代背景下开始探索如何实现车货匹配。研究如何充分整合社会运力资源，实现高效率的车货匹配具有重要的社会意义与研究价值。

物流企业在发展自身时，最先需要分析的就是运输线路，运输线路的测定直接影响着运输工具的规划，间接影响着运输成本的高低，也是决定利润收益的关键。线路的制定一般只有三种，即干线、专线、直线，这三种线路基本上是以区域面积划分的。通俗的讲，干线是省际之间，运输的路线也是国道；专线也是省际之间，通常是城市之间为运输终端，专线基本上建立在一线城市之间；支线则是设立在省内的运输分支，运输的货物量也只是承担省内的运输货物。三者之间是相辅相成的，形成递进式运输。

运量预测是一个复杂问题，能够对运量造成影响的因素很多，包括经营问题、社会问题、环境问题等等。本文采用时间序列分析法预测运量趋势。时间序列分析需要长时间且大量的数据积累，才能达到较为准确预测未来数据的目的。但是，由于比赛数据集只有 2018 年 1 月-2019 年 7 月共近一年半时间内的运单数据，数据的时间跨度太短，我们对数据集进行处理后得到的运量月度时间序列只有 18 个点。面对这样短的时间序列，大多数模型都很难准确预测未来运单运量的发展趋势。

车货匹配中货源是唯一品和非标准品，针对每一次新的订单要求，需要均衡考虑反馈率

和环境。本模型中采取简单的将数学建模模型和机器学习 XGBoost 结果进行加权，从而综合考虑历史与现状，但是这种模型对车辆实时承运情况考虑不足，可能模型给出的最佳匹配车辆当时正在运输其它运单，最终执行运单的车辆不是最佳匹配结果。

面对有多个提货、卸货节点的运单需求时，对节点次序进行调整和优化以满足路线总长最小，以及对节点顺序有特殊要求的优化问题是困难的问题。图论中必须经过某些中间节点的最短路径规划问题是一个 NP-hard 问题，目前除了穷举之外没有快速的精确算法，有蚁群算法、遗传算法等近似解法可以较高效的求取近似最优解。本文中提出的最短路线模型只能提出相对优解。

货运路线的选择除了考虑路线长度，还要考虑道路状况、车辆状况、司机习惯、客户需求、车辆装载量、道路坡度等其它因素的影响。这些影响因素对于道路选择的影响机制和影响程度难以建模，本文中提出的模型评分项目和权值的指定全凭经验，可能会影响模型优化结果的准确性。

2. 分析目的及思路

2.1. 分析目的

2.1.1. 运量走势

运量预测是运输组织工作中规划运能利用和编制日常运输计划的基本依据，也是对运输设备新建和扩能改造提出运营要求的基本依据。因此，运量预测的准确性以及运量发展变化趋势的正确认识与把握，对提高运输组织工作的预见性、改进运输规划工作具有重要意义。

2.1.2. 车货匹配

为了充分利用大数据资源，通过互联网技术提高信息检索能力和匹配效率，减少因信息不对称问题造成的物流资源浪费问题，达到去中介化的目的，尽可能实现最优车货匹配方案。

2.1.3. 线路分布

通过统计历史运单的路线分布，对于新的运货需求，规划出一条更多司机走过的、距离最短的路线。

2.2. 分析思路

2.2.1. 运量走势

运量预测的方法很多，总的来说，可以分为定性预测和定量预测两大类。定性预测方法又叫判断分析法，是依据人们在市场活动中获得的经验和分析能力，通过对影响市场变化的各种因素的分析、判断和推理，来预测未来的发展变化。定性预测方法的特点是简便易行，特别是在不可控因素和不可定量因素比较多时，采用这种方法进行短期判断有其明显的优势。然而，这种方法也有其缺陷，它不能提供以精确数据为依据的预测结果，主观随意性比较大，有时易发生疏忽和失误。定量预测方法。定量预测法又称数量分析法或数理统计预测

法，它是根据市场调查所取得的数据资料，运用数字模型进行计算，并据此预测未来市场变化的一种预测方法。这种预测方法的一个显著特点是运用数学、统计学和计算机等方法或工具，用数据对未来进行客观描述，因此，其科学性、严密性更强。

综上所述，本文采用定量预测法中的时间序列预测法对运量进行预测。在社会经济发展过程中，很多经济变量的发展变化都表现出与时间呈某种特定关系，运输需求也是如此。基于此，我们可以通过对运量的时间数列外推的方法预测未来运量变化趋势。时间序列预测法的特点是把预测变量看做是时间的函数。当所研究的运量时间数列变化没有大的波动时，这种方法较为理想。

模型在运量分析的处理过程中，首先设置提货点和卸货点，然后在地址数据中搜索符合条件的运单 ID，并记录。最后在运单数据中，搜索运单 ID 对应的运单，记录该运单的吨数和方数到时间序列中。在完成时间序列的收集后，我们使用 ARIMA 模型时间序列分析方法对时间序列进行分析。

2.2.2. 车货匹配

车货匹配算法场景本质可看做推荐场景，尤其需要注意的是其中的反馈率。在一个时间窗口内，进行联系的货源除以总货源叫做反馈率。反馈率和地区（区县一级）的供需关系呈现强烈正相关。反馈率一旦达到一个阈值，就会在地区形成一种新的平衡：用户自然流失等于或小于平台自然流入，地面团队可以把更多精力放在服务用户身上，而非拉新促活。所以对于业务指标来说，完成反馈率提升甚至比提供更有效的用户匹配更重要。

本方案中的车货匹配采用了建模与机器学习相结合的方式，车货匹配算法场景本质可看做推荐场景，即可以套在 CTR、CVR 的模型上，此处选用 XGBoost 进行加权。车货匹配方案中，考虑起始点终点后，将地址数据、运单数据、整理后的车辆数据进行全连接，可以对同一辆车的历史起终点进行统计，对于同一辆车，考虑到反馈率的影响，历史频次越高则权重更大，即相同需求下，认为同一辆车更倾向选择权重大的起终点进行运输，同时使用 XGBoost 来预测车-货的基础相关性，实际是一个 CTR 和 CVR 混布模型。结合基础相关性和反馈率最后给出合理的车货匹配方案。

2.2.3. 线路分布

本方案中的线路分布采用了统计与图论建模相结合的方式。首先对每个客户的所有历史运单信息，统计出走每一条路线的历史运单数。之后根据客户新的货运需求，给出提货地点、卸货地点的情况下，在历史线路分布中找到符合需求的订单，可以知道历史上司机们是怎么安排提货、卸货顺序的，走了什么路线。同时系统根据图论中求最短路径的 Floyd 算法，通过将中间节点排列组合的方式给出不同的途经顺序，求解必须经过某些中间节点的最短路径问题，给出一条最短路径。最后结合历史上的路线和图论优化路线，给出一条推荐路线和推荐星级。推荐星级表示路线的可信程度，最小是 1，最大是 5。

3. 数据分析与建模

3.1. 运量走势

我们把 2.2.1 节中介绍的思路重述如下：统计两地之间一个月所有订单发送货物的吨数和方数之和，得到两地之间运量随时间变化的序列。使用 ARIMA 模型对上述时间序列进行

分析，预测未来两地之间运量的变化。下面我们使用伪代码来描述上述思路:

【1】 在地址数据从上到下扫描到第 i 条。

a) 若为新的运单 ID，记录该 ID，跳转到【2】

b) 否则继续向下搜索，重复运行【1】

【2】 比对提货地点，判断是否为目标提货点 Pos_A，

a) 若是，跳转到【3】

b) 否则，跳转到【1】

【3】 向下搜索第 i+1 条数据，判断是否为目标卸货点 Pos_B。

a) 若是目标卸货点，则记录本运单 ID。在运单数据中搜索该运单。

i. 判断本运单的时间点，将本条数据中包含的吨数和方数，加在预先设置好的吨数/方数时间序列上。

【4】 跳转到【1】

经过上述处理之后，我们可以得到客户（A/B）在 2018 年 1 月至 2019 年 7 月的所有运单发送货物的吨数和方数的月度数据，即时间序列。下面对客户 A 和客户 B 的数据处理结果分别进行说明。

客户 A：未经过整理的数据，地址数据共有 21472 条，运单数据共有 9142 条。

经过分析数据，我们观察到客户 A 所有订单的提货点均位于陕西咸阳，而卸货点有很多城市。但是，我们也观察到地址数据中存在同一运单下多个卸货点在不同城市的情况，为了便于按照前述思路处理数据，我们不考虑这种情况，只统计卸货点在同一城市的运单数据。最终得到的统计结果存储到附件“运量走势\数据整理\客户 A\数据\result.csv”文件中，部分结果如图 1。

Csv 文件中每一行存储的是某提货城市到某卸货城市从 2018 年 1 月到 2019 年 7 月的运量月度时间序列，数据格式为（提货省，提货市，卸货省，卸货市，方数时间序列，吨数时间序列）。共收集到 83 条数据。

因为客户 A 数据集中有 19 个月的数据，所以统计得到的时间序列长度为 19。通过观察方数时间序列和吨数时间序列，我们可以发现大部分的时间序列最后一个点为 0。分析数据集我们可以发现，最后一个月的数据只有该月 1 号的数据。为了便于分析时间序列的规律，我们在分析的时候忽略最后一个点。

	A	B	C	D	E	F	G
1	陕西	咸阳	陕西	西安	[2130.0, 360.(	[868.1799999999997,	
2	陕西	咸阳	宁夏	银川	[25.0, 0, 275.(	[24.75, 0, 306.0599999	
3	陕西	咸阳	陕西	咸阳	[65.0, 80.0, 54	[11.6, 67.69999999999	
4	陕西	咸阳	重庆	重庆	[350.0, 0, 0, 0	[322.2, 0, 0, 0, 0, 0, 0, 0	
5	陕西	咸阳	陕西	汉中	[415.0, 60.0, 2	[239.52, 144.60000000	
6	陕西	咸阳	青海	西宁	[40.0, 40.0, 30(	[24.7, 91.1, 295.19999	
7	陕西	咸阳	新疆	乌鲁木齐	[40.0, 0, 910.(	[20.0, 0, 1000.9000000	
8	陕西	咸阳	陕西	商洛	[200.0, 10.0, 1	[161.0, 10.0, 195.6, 202	
9	陕西	咸阳	陕西	宝鸡	[240.0, 90.0, 1	[74.9, 91.16, 78.4, 157.3	
10	陕西	咸阳	陕西	渭南	[160.0, 0, 157	[33.4, 0, 87.3, 254.2700	
11	陕西	咸阳	甘肃	兰州	[40.0, 0, 758.(	[14.7, 0, 612.76999999	
12	陕西	咸阳	陕西	安康	[80.0, 0, 66.0,	[6.8, 0, 72.0, 17.08, 0, 1	
13	陕西	咸阳	甘肃	武威	[0, 10.0, 45.0,	[0, 8.4, 38.5999999999	
14	陕西	咸阳	陕西	延安	[0, 0, 240.0, 1	[0, 0, 254.18, 201.3499	
15	陕西	咸阳	甘肃	天水	[0, 0, 52.0, 25	[0, 0, 48.19999999999	
16	陕西	咸阳	甘肃	白银	[0, 0, 45.0, 0,	[0, 0, 37.87, 0, 47.8, 13(	
17	陕西	咸阳	甘肃	酒泉	[0, 0, 36.0, 60	[0, 0, 36.72, 53.1, 38.3,	
18	陕西	咸阳	河南	许昌	[0, 0, 30.0, 24	[0, 0, 30.0, 60.0, 50.3, 0	
19	陕西	咸阳	甘肃	张掖	[0, 0, 65.0, 10	[0, 0, 16.9, 7.58, 0, 0, 0,	
20	陕西	咸阳	甘肃	平凉	[0, 0, 215.0, 3	[0, 0, 173.04, 319.92, 2	

图1　客户 A 部分运量时间序列

客户 B：未经过整理的数据，地址数据共有 55757 条，运单数据共有 26627 条。

经过分析数据，我们发现客户 B 的数据量比客户 A 的数据量大很多。提货点不只有一个城市，同时也存在一个运单的多个卸货点不在同一个城市的情况。如果以城市为提货点和卸货点的地点单位，计算量相当巨大，所以我们以省为地点单位统计时间序列，同时忽略多个卸货点不在同一个省份的运单。最终得到的统计结果存储到附件“运量走势\数据整理\客户 B\数据\result.csv”文件中，部分结果如图 2：

result.csv

河北	安徽	"[1480.0, 1200.0, 1283.0, 0, 290.0, 0, 110.0, 0, 100.0, 0, 0, 0, 0, 0, 0, 60.0, 0, 0, 0]"	"[360.0, 273.0, 417.0, 0, 78.3, 0, 35.0, 0, 30.0, 0, 0, 0, 0, 0, 0, 20.0, 0, 0, 0]"
河北	山东	"[40.0, 360.0, 60.0, 120.0, 100.0, 460.0, 100.0, 100.0, 100.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[19.0, 88.0, 10.0, 40.0, 30.0, 140.0, 30.0, 30.0, 30.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	浙江	"[50.0, 0, 0, 0, 100.0, 0, 0, 0, 700.0, 0, 0, 200.0, 100.0, 0, 0, 0, 0, 0, 0]"	"[9.6, 0, 0, 0, 19.2, 0, 0, 0, 238.44, 0, 0, 67.84, 35.0, 0, 0, 0, 0, 0, 0]"
河北	江苏	"[0, 180.0, 0, 0, 0, 0, 100.0, 0, 400.0, 0, 0, 100.0, 60.0, 0, 0, 0, 0, 0, 0]"	"[0, 42.0, 0, 0, 0, 0, 30.0, 0, 124.0, 0, 0, 30.0, 20.0, 0, 0, 0, 0, 0, 0]"
河北	湖北	"[0, 0, 0, 0, 100.0, 300.0, 0, 300.0, 100.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 0, 30.0, 90.0, 0, 90.0, 30.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	河南	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	河北	"[6054.0, 2402.0, 3655.0, 2101.0, 240.0, 420.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[1674.4, 477.0, 1028.0, 325.0, 34.5, 63.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	陕西	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	广西	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	天津	"[280.0, 0, 101.0, 101.0, 210.0, 0, 0, 100.0, 100.0, 0, 100.0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[60.0, 0, 44.0, 47.0, 67.5, 0, 0, 30.0, 30.0, 0, 30.0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	四川	"[0, 0, 0, 0, 100.0, 100.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 0, 30.0, 30.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	广东	"[106.0, 0, 1.0, 126.0, 600.0, 300.0, 480.0, 240.0, 240.0, 60.0, 360.0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[83.086, 0, 31.59, 188.648, 191.058, 158.672, 253.19, 127.664, 127.
河北	上海	"[11.0, 2.0, 4.0, 0, 0, 100.0, 0, 600.0, 2400.0, 3800.0, 3200.0, 4100.0, 1700.0, 0, 0, 0, 0, 0, 0]"	"[124.57, 60.0, 120.0, 0, 0, 30.0, 0, 208.78, 844.2900000000001, 135.
河北	云南	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	福建	"[0, 0, 0, 0, 0, 300.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 0, 0, 90.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	贵州	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	宁夏	"[0, 0, 0, 90.0, 120.0, 100.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 27.0, 34.0, 27.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	海南	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]"
河北	重庆	"[15.0, 0, 24.0, 125.0, 280.0, 300.0, 90.0, 170.0, 0, 180.0, 180.0, 0, 0, 0, 0, 0, 0, 0, 0]"	"[59.26, 0, 115.634, 157.17, 127.812, 93.998, 47.056, 84.394, 0, 93.736, 95.12

图2　客户 B 部分运量时间序列

Csv 文件中每一行存储的是某提货地省份到某卸货地省份从 2018 年 1 月到 2019 年 7 月运量的月度时间序列，数据格式为（提货省，卸货省，方数时间序列，吨数时间序列）。共收集到 868 条数据。

和客户 A 的数据集一样，因为客户 B 数据集中有 19 个月的数据，所以统计得到的时间序列长度为 19。通过观察序列发现大多数序列的最后一个点都是 0，原因与客户 A 相同，所以我们在进行时间序列分析的时候同样忽略最后一个点。

此外，考虑到我们统计了所有省份两两之间的运量时间序列，但存在某提货省到某卸货省没有运输记录的个例，所以部分数据存在运量时间序列全为零的情况。我们在时间序列分析的时候将这些数据视为无效数据，不予分析。

3.2. 车货匹配

3.2.1. 数据清洗

分两步对原始数据进行整理。

①重点分析运过货的车辆基础数据以及运单数据；

②用历史运单数据中的最大吨数与方数对车辆基础数据中的空数据以及不合理数据（体现在车辆能够承载的最大吨数或最大方数小于货物的吨数或方数）进行修改以及更新。

数据整理部分结果展示如下，以下对客户 A、B 的数据分别进行说明。

3.2.2. 整理结果

客户 A：未经过整理的数据，车辆基础数据共有 106340 条数据，运单数据 9142 条，经过数据整理后，将车辆基础数据表与运单数据表进行对应连接后得到 9142 条数据，与运单数据相关的车辆数据共 3136 条数据。若考虑起始点终点后，将地址数据、运单数据、整理

后的车辆数据进行全连接后，得到 21472 条数据。

由图 3 看出，源数据中存在着空数据的情况。

carID_3	license_plate_3	car_length_3	car_size_3	max_tonnage_3	max_square_3
100001	时N7**75	17.50	敞车	180	49
100002	周AB**07	(Null)	(Null)	(Null)	(Null)
100003	西B1**8E	13.00	敞车	90	40
100004	周F8**41	13.00	高栏	68	32
100005	西B3**6G	13.00	敞车	90	40
100006	周B2**32	(Null)	(Null)	(Null)	(Null)
100007	西AG**K6	4.20	厢车	19	8

图3　车辆基础数据中存在空数据

将运单数据与车辆数据进行全连接操作，得到图 4，可以看出，有不合理的数据存在，存在运单数据中的吨数大于车辆最大吨数的情况，我们假设这种情况是因为车辆数据更新不及时（或车辆数据错误）导致的。

waybill_number_2	required_delivery_time_2	required_arrival_time_2	car_ID_2	total_squares_2	max_square_3	gross_tonnage_2	max_tonnage_3	license_plate_3	car_length_3	car_size_3
20180129555363	2018-1-29 23:59:00	2018-1-30 23:59:00	100800	40	10.00	11.3	30.00	右AP**21	5.20	敞车
20180202889122	2018-2-2 23:59:00	2018-2-3 23:59:00	100800	10	10.00	8.9	30.00	右AP**21	5.20	敞车

car_ID_2	total_squares_2	max_square_3	gross_tonnage_2	max_tonnage_3
100800	40	10.00	11.3	30.00
100800	10	10.00	8.9	30.00

图4　源数据中的不合理数据

整理后的车辆基础数据表如图 5 所示，可以看出对空数据以及不合理数据进行整理更新。

carID_5	max_tonnage_5	max_square_5	car_size_5	license_plate_5	car_length_5
105344	20	15	高栏	右F5**96	10
105352	70	32	高栏	右AD**78	13
105369	70	32	高栏	中E6**86	13
105391	70	32	高栏	西FG**55	13
105409	30	15	高栏	右D9**59	7
105420	40	22	高栏	右DA**57	10
105434	40	22	高栏	中D1**62	10
105439	70	35	高栏	右F6**96	13
105448	70	32	高栏	右EA**59	13
105482	70	32	高栏	右F3**22	13
105489	70	32	高栏	右DB**95	13
105490	70	33	高栏	右KB**23	13
105491	70	32	高栏	右KB**22	13
105492	70	32	高栏	北CB**99	13
105519	20	25	高栏	右A2**G6	4
105525	13	25	厢车	右D7**96	6
105539	70	35	高栏	中E6**11	13

图5　清洗后的数据

客户 B：未经过整理的数据，车辆基础数据共有 106340 条数据，运单数据 27627 条，经过数据整理后，将车辆基础数据表与运单数据表进行对应连接后得到 27627 条数据，与运单数据相关的车辆数据共 13614 条数据。若考虑起始点终点后，将地址数据、运单数据、整理后的车辆数据进行全连接后，得到 56757 条数据。

首先同上所述，车辆基础源数据中存在着空数据的情况。

将运单数据与车辆数据进行对应连接操作，且其中可以看出有不合理的数据存在，存在运单数据中的吨数大于车辆最大吨数的情况，我们假设这种情况是因为车辆数据更新不及时（或车辆数据错误）导致的。

整理后的车辆基础数据表如图 6 所示，可以看出对车辆空数据以及不合理数据进行整理更新。

b_new_car @bigdata2 (127.0.0.1) - 表

b_2 @bigdata2 (127.0.0... × | b_new_car @bigdata2 (... × | combine123 @bigdata2 (... ×

文件 编辑 查看 窗口 帮助

导入向导 导出向导 筛选向导 网格查看 表单查看 备注 十六进制 图像

carID_3	max_tonnage_3	max_square_3	car_size_3	license_plate_3	car_length_3
37672	60	51	厢车	秒CN**56	9.6
37711	55	35	厢车	周M1**56	9.6
37809	80	75	高栏	工D9**80	13
38098	120	116.546	厢车	左DC**25	17.5
38131	55	40	厢车	周M9**17	9.6
38420	150	124.377	敞车	周N9**86	17.5
38535	60	40	高栏	工Q6**4J	9.6
38813	60	40	高栏	北C1**91	9.6
38862	70	60	高栏	西JR**51	13
38963	60	60	高栏	周C3**52	9.6
39305	60	15	厢车	秒CN**26	9.6
39390	80	70	高栏	民H5**78	13
39706	130	129.662	厢车	年AA**36	17.5
39795	120	63.28	敞车	周M9**99	17.5
39806	80	100	高栏	周A8**71	13
40235	60	69.77	高栏	上K9**3C	13
40735	160	100	敞车	周K6**99	17.5
40738	130	100	敞车	周K6**96	17.5
41513	140	100	敞车	周KJ**03	17.5
41532	120	117.015	厢车	下C1**88	17.5
41818	140	129.217	厢车	年BD**73	17.5

图6　清洗后的数据

3.2.3.　模型构建分析

为了考虑全面，还需知道当时车辆的地理坐标，因为是未来数据，此处假设车辆与起点的地理距离为

$$d_{车\text{-}起} \sim (x_{车}, x_{起})^2 + (y_{车}, y_{起})^2$$

同时还与路径损耗γ相关，即车辆当前位置-起点路径的路径情况越好，路径损耗γ越小，此处选用路径系数$\beta=e^{d_{车\text{-}起}}\times\gamma$。

即假若起终点以及吨数方数确定：

①只要车辆最大吨数和方数大于发货需求的吨数和方数，则认为该车可以作为承运车辆；

②考虑到车货匹配的反馈率要求，通过数据分析选出较频繁的备选车辆；

③并通过 XGBoost 对备选车辆进行预测，得出空闲可用车辆的优先度α。

④将同一起终点的所有空闲可用车辆的优先度α与路径系数β综合考虑，将ε*α+（1-ε）*β进行排序，选出最大值即应该匹配的车辆，具体考虑方案ε可以参考当前的情况，假如物流货运没有很紧张，一般可主要考虑空闲可用车辆的优先度α。

3.3. 线路分布

3.3.1. 统计客户历史运单数据的线路分布

这一部分使用 MySQL+Python Pandas 库的技术路线，python 与 MySQL 数据库的通信使用 pymysql 库完成。

3.3.1.1. 解析运单采用的线路

模块对客户 A 或 B，查询运单数据库中所有的运单 ID，保存在变量 yundanIDs 中。之后对每一个运单 ID，模块从地址数据库中查询该运单的所有途经节点的类型（提货还是卸货）、节点所在的省、市。然后将每一个运单 ID 查询出的所有节点串起来，形成一个途经城市轨迹序列，例如(('提货', '陕西,咸阳'), ('卸货', '陕西,西安'), ('卸货', '陕西,咸阳'), ('卸货', '甘肃,兰州'), ('卸货', '陕西,宝鸡'), ('卸货', '河南,许昌'))。这就是运单采用的线路。对于货车在一个城市不同区域多次卸货的情况，粒度只细分到城市，即只考虑车在这个城市卸过货，只保留第一次出现的记录，其余重复节点直接删除。将线路和对应的运单 ID 保存在数据帧中。将数据帧的内容保存在 csv 文件（track_A 或 B.csv）中。

3.3.1.2. 统计每条路线对应的运单数

这里使用 pandas 库的 value_counts()方法，统计每条线路对应的运单数，按照降序排列，结果保存在数据帧 trackCount 中。trackCount 中部分内容如图 8 或图 10 所示。客户 A 的所有线路和线路对应运单数见附件“线路分布”文件夹下 trackCount-A.csv，第一列 Track 代表线路，第二列 Count 代表走这条线路的运单数，第三列 Percentage 代表走这条线路的运单数占客户 A 的运单总数的比例。客户 B 有对应的 trackCount-B.csv.

3.3.1.3. 统计结果

通过 3.3.1.1 和 3.3.1.2 节整理，系统统计出客户 A 和 B 的所有运单走的不同路线，走每一条路线的运单数，按照降序排列可以给出频数直方图，如图 7 图 9 所示。直方图按照运单数大于 100，记录数不超过 20 条进行筛选以便于画图。

客户 A 在陕西咸阳提货、到陕西西安卸货的需求非常大，在总共 9142 个运单中有 3514 个走这条路线，占比 38.4%。A 的部分统计结果见图 8，表头 track 代表路线，freq 代表运单数，percentage 代表占运单总数的比例。

图 9 是客户 B 的线路分布直方图，图 10 是部分数据。可以看到，客户 B 的业务范围更加广泛，提货城市共 94 个，卸货城市共 264 个。前五大线路需求是安徽马鞍山提货、浙江

杭州卸货；广东中山提货、广东东莞卸货；陕西宝鸡提货、陕西宝鸡卸货；安徽六安提货、安徽芜湖卸货；安徽马鞍山提货、浙江湖州卸货。

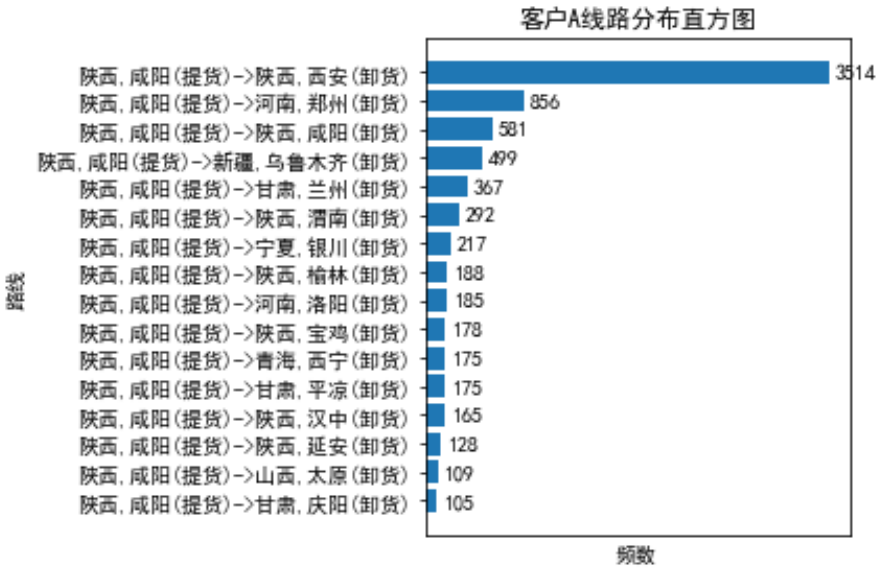


图7　客户 A 线路分布直方图

Index	track	freq	percentage
0	陕西,咸阳(提货)->陕西,西安(卸货)	3514	0.38438
1	陕西,咸阳(提货)->河南,郑州(卸货)	856	0.0936338
2	陕西,咸阳(提货)->陕西,咸阳(卸货)	581	0.0635528
3	陕西,咸阳(提货)->新疆,乌鲁木齐(卸货)	499	0.0545832
4	陕西,咸阳(提货)->甘肃,兰州(卸货)	367	0.0401444
5	陕西,咸阳(提货)->陕西,渭南(卸货)	292	0.0319405
6	陕西,咸阳(提货)->宁夏,银川(卸货)	217	0.0237366
7	陕西,咸阳(提货)->陕西,榆林(卸货)	188	0.0205644
8	陕西,咸阳(提货)->河南,洛阳(卸货)	185	0.0202363
9	陕西,咸阳(提货)->陕西,宝鸡(卸货)	178	0.0194706
10	陕西,咸阳(提货)->甘肃,平凉(卸货)	175	0.0191424
11	陕西,咸阳(提货)->青海,西宁(卸货)	175	0.0191424

图8　客户 A 部分线路频数

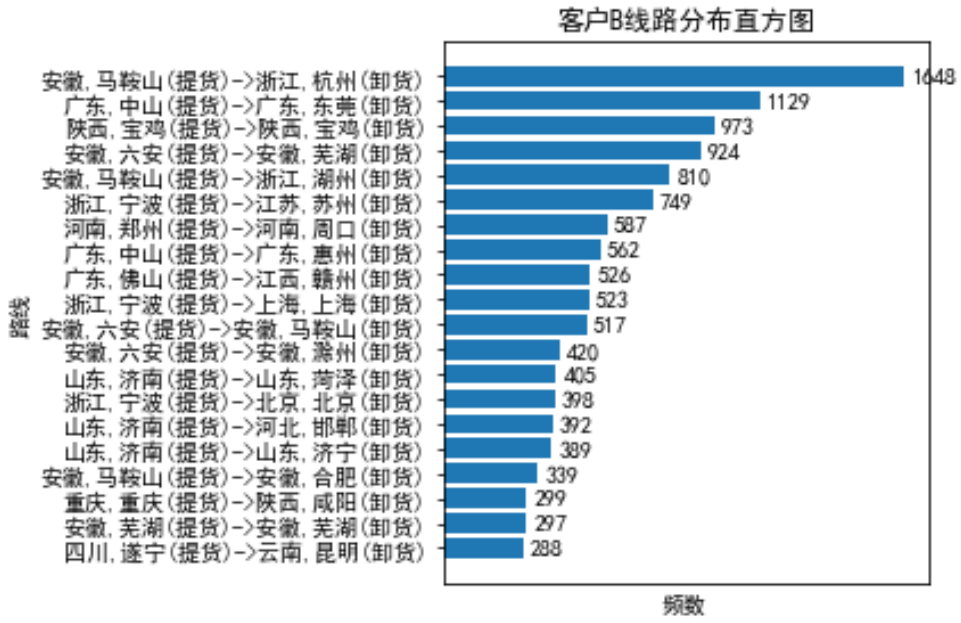


图9　客户 B 线路分布直方图

Index	track	freq	percentage
0	安徽,马鞍山(提货)->浙江,杭州(卸货)	1648	0.0618921
1	广东,中山(提货)->广东,东莞(卸货)	1129	0.0424006
2	陕西,宝鸡(提货)->陕西,宝鸡(卸货)	973	0.0365419
3	安徽,六安(提货)->安徽,芜湖(卸货)	924	0.0347016
4	安徽,马鞍山(提货)->浙江,湖州(卸货)	810	0.0304203
5	浙江,宁波(提货)->江苏,苏州(卸货)	749	0.0281293
6	河南,郑州(提货)->河南,周口(卸货)	587	0.0220453
7	广东,中山(提货)->广东,惠州(卸货)	562	0.0211064
8	广东,佛山(提货)->江西,赣州(卸货)	526	0.0197544
9	浙江,宁波(提货)->上海,上海(卸货)	523	0.0196417
10	安徽,六安(提货)->安徽,马鞍山(卸货)	517	0.0194164
11	安徽,六安(提货)->安徽,滁州(卸货)	420	0.0157735

图10　客户 B 部分线路频数

3.3.2.　最短路线模型

3.3.2.1.　使用图论算法求最短路的出发点

可以看到在运单数据中，当有多个卸货城市时，不同的运单会采取不同的顺序。不同的顺序可能导致车辆行驶的路程差距很大，导致时间和车辆资源的浪费。比如客户 A 的运单 612748，途径地点信息如图 11 所示。该运单从陕西咸阳提货，先后在陕西西安、青海西宁、陕西安康卸货。这条路径不是在这些地点卸货的最短路，从陕西到青海的路程很长，这会导致走很长的重复道路。

```
1764369,612748,提货,陕西,咸阳,礼泉县,108.48256879,34.59785379,1,0
1764370,612748,卸货,陕西,西安,灞桥区,109.08464713,34.28823220,0,0
1764371,612748,卸货,青海,西宁,城北区,101.75950917,36.74214011,0,1
1764372,612748,卸货,陕西,安康,旬阳县,109.35199354,32.83173366,1,2
```

图11　运单 612748 途径地点信息

3.3.2.2.　模型构建分析

模型假设只需要满足在某些地点提货、在另一些地点卸货的需求，不需要考虑某个城市是主节点，即必须最先或者最后经过。在这个假设下，只需要把提货地点、卸货地点全排列，对地点序列中的所有地点两两之间用最短路径连接，每一种可能都会求出一条长度尽量小的路径，所有序列中长度最小的就是最短路线。使用我们的算法，对运单 612748 的需求给出的推荐线路是陕西,咸阳(提货)->陕西,西安(卸货)->陕西,安康(卸货)->青海,西宁(卸货)，先在陕西省内卸货完毕之后再去青海省卸货，避免重复走陕西到青海的路程。

3.3.3.　专家评议法

专家评议法是出现较早且应用较广的一种评价方法。它是在定量和定性分析的基础上，以打分等方式做出定量评价，其结果具有数理统计特性。其最大的优点在于，能够在缺乏足够统计数据和原始资料的情况下，可以做出定量估计。

在模型中使用加乘评分法。模型在推荐路线时，先参考历史上满足提货卸货地点要求的线路，一种可能的查询结果如表 1，有 x 个运单采用 A 路线，y 个运单采用 B 路线，z 个运单采用 C 路线。

表1　查询满足需求的历史运单的一种可能结果

Index 索引	Track 运单路线	Freq 频数	Percentage 百分比
0	A	x	x/(x+y+z)
1	B	y	y/(x+y+z)
2	C	z	z/(x+y+z)

3.3.3.1.　评分指标

专家评分系统考察三项指标，三项得分加权求和得到总评，总评分最高的路线是系统最终推荐的路线。

1.　百分比

百分比（Percentage）是走线路 X 的运单数占满足需求的运单总数的比例。$0 \leq Percentage \leq 1$。模型考虑该指标的原因与反馈率有关，历史上较多司机选择的路线可能具有更加著名，路况更好或者更安全的特点，模型在推荐路线时要考虑司机们的选择的合理性。

2.　热门路线

模型中定义热门路线（Hot Track）的概念。热门路线就是对应运单数较多的路线，本模型认为对应运单数（表中的频数 freq）大于等于 10 的路线是热门路线。对于热门路线，该项得分 1 分；非热门路线不得分。模型考虑该指标的原因也是反馈率。

3.　最短路线算法推荐的路线

在 3.3.2 节叙述的模型会给出一个最短路线的推荐结果 suggestPath。对于图论算法推荐的最短路线，该项得分 1 分；不是最短路线的不得分。模型考虑该指标的原因是想帮助货车司机少走回头路，提高物流效率。

3.3.3.2. 评分方法

路线 X 的总分 score 满足公式

$$\text{score(X)} = a\, f_1(X) + b\ \text{Percentage}(X)\ + c\, f_2(X)$$

式中a，b和c是模型的三个参数，其中a是热门路线指标的权重，模型中设置为 0.2；b是百分比指标的比重，模型中设置为 0.3；c表示最短路径指标的权重，模型中设置为 0.5.三个参数满足关系

$$a + b + c = 1$$

f_1表示路线是否是热门路线，满足

$$f_1(X) = \begin{cases} 1, & \text{如果}X\text{是热门路线} \\ 0, & \text{其它} \end{cases}$$

f_2表示路线是否是最短路径，满足

$$f_2(X) = \begin{cases} 1, & \text{如果}X\text{是最短路径} \\ 0, & \text{其它} \end{cases}$$

如果最短路线算法求出的路线不在历史记录中，对应历史运单数为 0，$f_1(X) = 0$，$\text{Percentage}(X) = 0$.从公式可以看出，路线的总分是 1 分。

在所有路线中，取得分最高的路线作为算法推荐路线，并且给出推荐星级，路线的星级越高代表可信度越高，更推荐司机选择。得分与推荐星级对照表如表 2 所示。

表2 得分与推荐星级对照表

得分 score	推荐星级
$0.8 \leq \text{score} \leq 1$	5
$0.6 \leq \text{score} < 0.8$	4
$0.4 \leq \text{score} < 0.6$	3
$0.2 \leq \text{score} < 0.4$	2
$0 \leq \text{score} < 0.2$	1

4. 实现路径

4.1. 运量走势

运量走势部分采用 ARIMA 模型对运单的方数和吨数月度数据进行研究预测。ARIMA 模型称为差分自回归移动平均模型，其中，ARIMA(p,d,q)，AR 是自回归，p 为自回归项；MA 为移动平均，q 为移动平均项数，d 为时间序列成为平稳时所做的差分次数。所谓 ARIMA 模型，是指将非平稳时间序列转化为平稳时间序列，然后将因变量仅对它的滞后值以及随机误差项的现值和滞后值进行回归所建立的模型。

基本步骤：

Step 1: 根据时间序列的散点图、自相关图、偏自相关图，以 ADF 单位根检验其方差、趋势及其季节性变化规律，对序列的平稳性进行识别。

Step 2: 对非平稳序列进行平稳化处理。如果数据序列是非平稳的，并存在一定的增长或下降趋势，则需要对数据进行差分处理，如果数据存在异方差，则需要对数据进行技术处理，直到处理后的数据的自相关函数值和偏自相关函数值无显著地异于零。

Step 3: 根据时间序列模型的识别规则，建立相应的模型。若平稳序列的偏自相关函数是截尾的，而自相关函数是拖尾的，可断定序列适合 AR 模型；若平稳序列的偏自相关函数是拖尾的，而自相关函数是截尾的，则可断定序列适合 MA 模型；若平稳序列的偏自相关函数和自相关函数均是拖尾，则序列适合 ARMA 模型。
Step 4: 进行参数估计，检验是否有统计意义。
Step 5: 进行假设检验，诊断残差序列是否为白噪声。
Step 6: 利用已通过检验的模型进行预测分析。

4.2. 车货匹配

本车货匹配方案综合考虑供需匹配与地点，需要考虑起始点终点后，将地址数据、运单数据、整理后的车辆数据进行全连接后进行分析，可以看出其实对于一辆车来说，它的运单起终点选择是相对固定的，结合 mysql 以及 python，使用数学建模与机器学习方法 XGBoost，可以对同一辆车的历史起终点进行统计，对于同一辆车，历史频次越高则权重 α 更大，即相同需求下，认为同一辆车更倾向选择权重大的起终点进行运输。

这里有一点需要特别说明的，通过数据分析，可以看出客户 A 的发货点只有一个地方，而客户 B 的发货点和收货点都相对随机，在以下所有车货匹配方案中的推荐车辆备选中，比如客户 A 仅仅需要考虑收货地点与车辆的关系，而客户 B 则需要综合考虑起点和终点与车辆的关系。

假若起终点以及吨数方数确定，具体车货匹配如下：

①只要车辆最大吨数和方数大于运单需求的吨数和方数，则认为该车可以作为承运车辆；

②考虑到车货匹配的反馈率要求，通过数据分析选出较频繁的备选车辆（即最好起终点均在推荐方案中，可以简单理解为起点和终点的频次和），此处每辆车我们给出三种程度的推荐，如图 12 所示。

carID_5	max_tonnage_5	max_square_5	No1_choice	No1_weight	No2_choice	No2_weight	No3_choice	No3_weight	car_size_5	license_plate_5	car_length_5
108961	20	32	甘肃	6	青海	5	陕西	1	(Null)	右A9**R8	4
108962	21	33	甘肃	2	陕西	1			(Null)	右A3**D5	4
108983	80	40	河南	52	青海	6	甘肃	2	(Null)	时K7**10	13
108986	32	33	新疆	2				(Null)	(Null)	右EC**66	13
109001	100	33	陕西	2				(Null)	(Null)	右KB**47	13
109055	70	33	宁夏	1	甘肃	1			(Null)	中AB**01	13
109076	100	32	河南	1				(Null)	(Null)	时MB**13	13
109087	100	49	新疆	5				(Null)	(Null)	右EC**76	13
109102	55	20	河南	1				(Null)	(Null)	时NC**89	10

图12　推荐结果

由图 12 可以看出，No1_choice,No2_choice,No3_choice 分别对应三种程度的优先度，No1_weight,No2_weight,No3_weight 分别对应三种程度的权重（与历史运单频次相关，两地间运单越频繁，相对应的权重越大），其实这里的权重也会在后面的 XGBoost 的预测中体现出来。

③通过 XGBoost 进行预测，得出空闲可用车辆的优先度 α 。比如假如通过数据分析选出较频繁的 6 辆备选车辆，将此六辆车与之前一年半的订单的供需关系进行训练得到 XGBoost 模型，并通过 XGBoost 模型进行概率预测（类似于多分类问题），并将预测出的匹配概率进行排序，得到预测优先度 α 。比如如图 13 所示，每行是输入的订单，每列是 XGBoost 模型选择对应的概率~α ，一定程度上反映了历史的记录的影响，并具有一定的泛化能力。

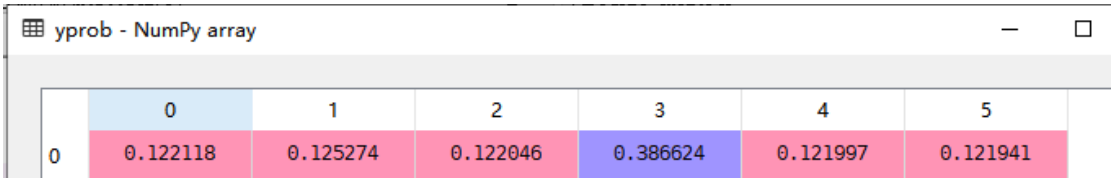


图13　XGBoost 模型预测 6 辆车对应的概率

④将同一起终点的所有空闲可用车辆的优先度α与路径系数β综合考虑，将 $\varepsilon\alpha+(1-\varepsilon)\beta$ 进行排序，选出最大值即应该匹配的车辆，具体考虑方案ε可以参考当前的情况，假如物流货运没有很紧张，一般可主要考虑空闲可用车辆的优先度α，即将ε设为 0.85 左右。

4.3. 线路分布

4.3.1. 查询满足需求的历史线路分布

模块查找历史路线中经过需求城市的记录，对于不同路线的运单数，统计占经过需求城市的运单总数的比例。以在陕西咸阳提货，在甘肃庆阳和宁夏银川卸货的需求为例，查询结果如图 14 所示，历史运单中满足需求的线路共有 2 条，有 3 个运单选择第一条路线，占满足需求的运单总数 5 的 60%；2 个运单选择第二条路线，占满足需求的运单总数 5 的 40%.

Index	track	freq	percentage
0	(('提货', '陕西,咸阳'), ('卸货', '甘肃,庆阳'), ('卸货', '宁夏,银川'))	3	0.6
1	(('提货', '陕西,咸阳'), ('卸货', '宁夏,银川'), ('卸货', '甘肃,庆阳'))	2	0.4

图14　查询满足需求的线路记录示例

4.3.2. Floyd 算法

4.3.2.1.　首先模型给全国 34 个省编号，映射关系如表 3 所示。

表3　省份名称对应的矩阵索引

省份	新疆	甘肃	内蒙古	黑龙江	吉林	辽宁	河北	北京	天津
编号	0	1	2	3	4	5	6	7	8
省份	山西	陕西	宁夏	青海	河南	山东	西藏	四川	重庆
编号	9	10	11	12	13	14	15	16	17
省份	湖北	安徽	江苏	贵州	湖南	江西	浙江	云南	广西
编号	18	19	20	21	22	23	24	25	26
省份	广东	福建	海南	香港	澳门	台湾	上海		
编号	27	28	29	30	31	32	33		

4.3.2.2.　根据 Floyd 算法，初始化图的权值矩阵和路由矩阵。

初始化权值矩阵$W^{(0)}$如图 15 所示。主对角线全是 0，表示每个省到自身的距离是 0.两省份如果接壤，矩阵对应位置是有限值，表示省会之间的距离，单位千米；如果不接壤，则

对应位置是无穷。

W - NumPy object array

	0	1	2	3	4	5	6
0	0	1625.5	inf	inf	inf	inf	inf
1	1625.5	0	870.1	inf	inf	inf	inf
2	inf	870.1	0	1314.4	1162.2	983	391.5
3	inf	inf	1314.4	0	240.6	inf	inf
4	inf	inf	1162.2	240.6	0	282.3	inf
5	inf	inf	983	inf	282.3	0	862.8
6	inf	inf	391.5	inf	inf	862.8	0
7	inf	inf	inf	inf	inf	inf	263.5
8	inf	inf	inf	inf	inf	inf	257.1
9	inf	inf	337.5	inf	inf	inf	174

图15　初始化权值矩阵$W^{(0)}$

初始化路由矩阵$R^{(0)}$如图 16 所示。$W^{(0)}$和$R^{(0)}$矩阵的值存储在 distance.npz 文件中.

R - NumPy object array

	0	1	2	3	4	5	6
0	0	1	-1	-1	-1	-1	-1
1	0	1	2	-1	-1	-1	-1
2	-1	1	2	3	4	5	6
3	-1	-1	2	3	4	-1	-1
4	-1	-1	2	3	4	5	-1
5	-1	-1	2	-1	4	5	6
6	-1	-1	2	-1	-1	5	6
7	-1	-1	-1	-1	-1	-1	6
8	-1	-1	-1	-1	-1	-1	6
9	-1	-1	2	-1	-1	-1	6

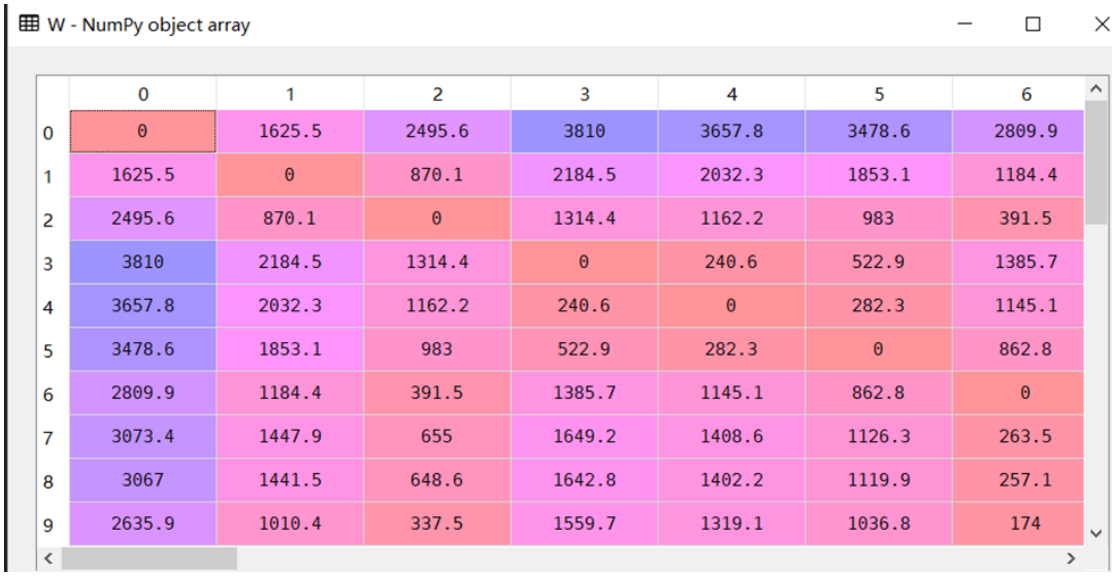
图16　初始化路由矩阵$R^{(0)}$

4.3.2.3.　运行 Floyd 算法，求任意两节点之间的最短路径和路由。

4.3.2.4.　运行结束后，得到任意两节点之间最短路径的权值矩阵$W^{(34)}$和路由矩阵$R^{(34)}$。

$W^{(34)}$和$R^{(34)}$存储在 distance_shortest.npz 文件中.

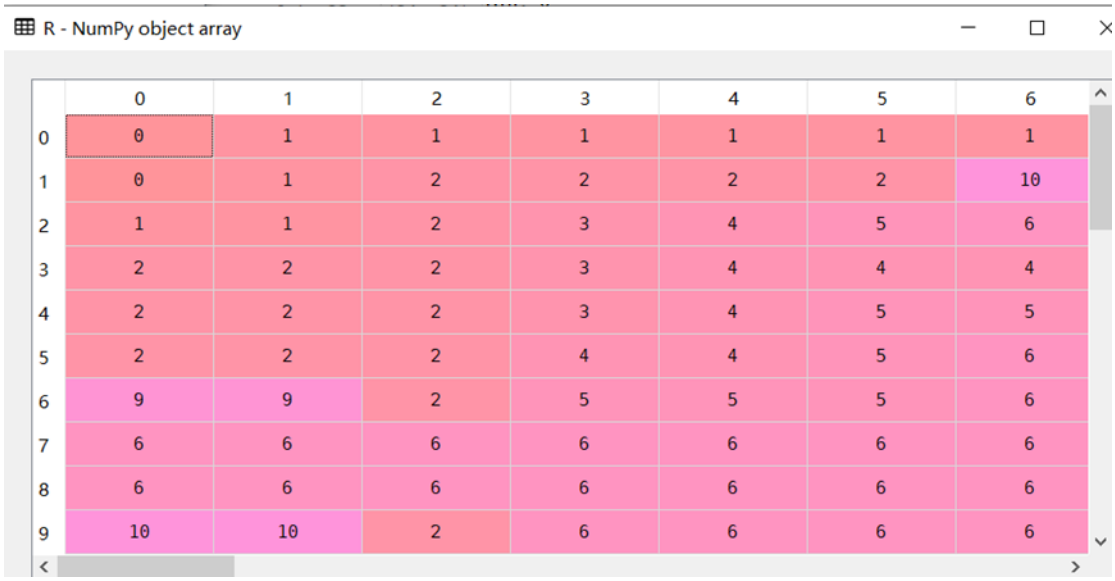
权值矩阵$W^{(34)}$如图 17 所示，矩阵中每个元素代表两省之间最短路径的长度。



	0	1	2	3	4	5	6
0	0	1625.5	2495.6	3810	3657.8	3478.6	2809.9
1	1625.5	0	870.1	2184.5	2032.3	1853.1	1184.4
2	2495.6	870.1	0	1314.4	1162.2	983	391.5
3	3810	2184.5	1314.4	0	240.6	522.9	1385.7
4	3657.8	2032.3	1162.2	240.6	0	282.3	1145.1
5	3478.6	1853.1	983	522.9	282.3	0	862.8
6	2809.9	1184.4	391.5	1385.7	1145.1	862.8	0
7	3073.4	1447.9	655	1649.2	1408.6	1126.3	263.5
8	3067	1441.5	648.6	1642.8	1402.2	1119.9	257.1
9	2635.9	1010.4	337.5	1559.7	1319.1	1036.8	174

图17　最终结果权值矩阵$W^{(34)}$

路由矩阵$R^{(34)}$如图 18 所示，采用前向路由，由矩阵可以得到任意两节点之间最短路径的路由。



	0	1	2	3	4	5	6
0	0	1	1	1	1	1	1
1	0	1	2	2	2	2	10
2	1	1	2	3	4	5	6
3	2	2	2	3	4	4	4
4	2	2	2	3	4	5	5
5	2	2	2	4	4	5	6
6	9	9	2	5	5	5	6
7	6	6	6	6	6	6	6
8	6	6	6	6	6	6	6
9	10	10	2	6	6	6	6

图18　最终结果路由矩阵$R^{(34)}$

例如，由路由矩阵得到上海到新疆的最短路径是上海->江苏->安徽->河南->陕西->甘肃->新疆，由$W^{(34)}$[33,0]得到路线近似总长度是 3444.2 千米。

4.3.3.　最短路线算法的实现

最短路线算法的伪代码如下。

----------伪代码----------

设所有的提货城市组成集合 X，卸货城市组成集合 Y。

Step1：把所有提货城市、卸货城市所在的省份记录下来,提货省份集合记为 S，卸货省份集合记为 T。设 m 是集合 S 的元素个数，n 是集合 T 的元素个数。

Step2：分情况讨论

Step2.1：如果提货省份和卸货省份的数量分别都是 1，由于图论优化算法以省作为图的

节点，所以对于两个节点的情况直接输出最短路径，对于多个城市在同一个省内的情况，模型认为城市之间距离是 0，所以同一个省内城市访问顺序不分先后。但是如果提货和卸货城市在同一个省且有重复的话，为了减少重复路程，会尽量按照回文方式安排路线，即 *x* 地（提货）->*y* 地（提货）->*z* 地（提货）->…->*z* 地(卸货)->*y* 地（卸货）->*x* 地（卸货）。

Step2.2：对于有多个提货省或者多个卸货省的情况，这时候最短路线问题转化为图论中必须经过某些中间节点的最短路问题。经过查阅资料，我们发现这是一个 NP-hard 问题，即没有好的多项式时间算法能解决这个问题，想要获得全局最优解只能靠穷举。考虑到必须先在所有的提货点提货完毕后才能进行卸货的约束条件，问题转化为必须先依次经过集合 *S* 中的所有节点之后再依次经过集合 *T* 中所有节点的最短路径。

模型采用遍历算法，即对于 *S* 中元素的每一种全排列和 *T* 中元素的每一种全排列，将提货点集合中两两节点之间用最短路径连接，卸货点集合中两两节点之间用最短路径连接，将提货路径终点与卸货路径起点用一条最短的路径连接起来。对于所有路径，找出长度最短的就是全局最优解，即满足需求的最短路线。

//遍历算法

Step 2.2.1:对 *S* 中元素，两两之间求最短路径，保存路径的长度

Step 2.2.2:对 *T* 中元素，两两之间求最短路径，保存路径的长度

Step 2.2.3:对 *S* 中任意一个元素 *x* 和 *T* 中任意一个元素 *y*，两两之间求最短路径，保存路径的长度

Step 2.2.4:对于 *S* 中元素的一种全排列$(s_{i_1}, s_{i_2}, \dots, s_{i_m})$（$i_1, i_2, \dots, i_m$是 $12 \dots m$的一种排列，就是一种顺序）和 *T* 中元素的一种全排列$(t_{j_1}, t_{j_2}, \dots, t_{j_n})$（$j_1, j_2, \dots, j_n$是 $12 \dots n$的一种排列），将两个节点序列拼接起来得到$(s_{i_1}, s_{i_2}, \dots, s_{i_m}, t_{j_1}, t_{j_2}, \dots, t_{j_n})$，求序列中相邻两节点之间的最短路径的长度，得到整个序列的长度。

Step 2.2.5:求出所有序列长度中的最小值，对应的省份序列就是最优的提货、卸货顺序 S_i 和 T_j。

Step 2.2.6:对于 S_i 中的每一个省份，在 *X* 中找到属于这个省的所有元素，顺序输出。对于 T_j 中的每一个省份，在 *Y* 中找到属于这个省的所有元素，顺序输出。

Step 2.2.6.1:如果 S_i 的最后一个节点和 T_j 的第一个节点相同，即在某省提货完毕后紧接着就卸货，则找 *X* 和 *Y* 的交集中属于该省的元素，构造回文序列以尽量减少重复路程

Step3: 输出近似最短路线和路线的长度

对最短路线算法的可行性进行简要说明。遍历算法的时间复杂度和空间复杂度都很高，均为 O(*m*!*n*!)级。如果提货或卸货省份数量非常多，算法求解的时间将非常长。不过客户 A 的数据集中涉及到的省份数（即图的节点数）最多为 4，客户 B 的数据集中涉及到的省份数最多为 3，使用该算法可以做到计算时间是微秒量级，所以针对本问题的场景是可以使用遍历算法。

5. 分析结果

5.1. 运量走势

我们任取 3.1 节中处理得到的 csv 文件中一运量时间序列，如图 19 所示。下面按照 4.1 节中介绍的算法对该序列进行分析。

```
#绝大多数情况，会从.txt/.xlsx/.csv文件中读取，在此为简化过程，以列表形
dta=[3345.0, 1905.0, 3860.0, 4492.0, 5726.0, 4992.0, 3443.0, 4053.0, 5
```

图19　任一时间序列

分析上述序列，我们可以观察到时间序列的变化非常剧烈。为了减少这种剧烈变化带来的影响，我们先将各点取对数，再进行下一步处理。绘制取对数之后的时间序列曲线图如图 20。

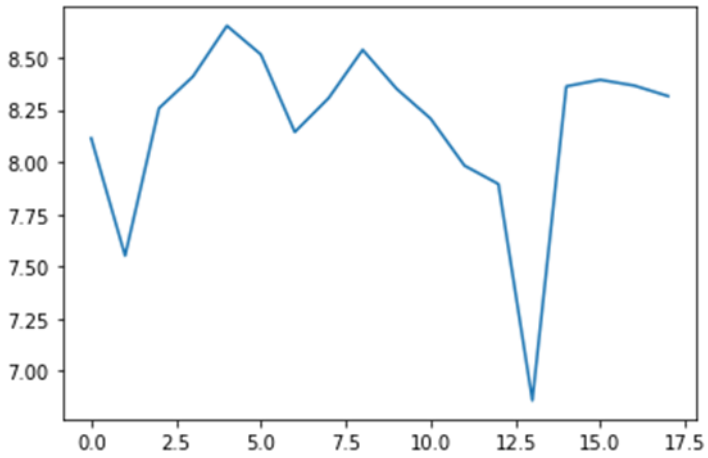


图20　取对数之后的时间序列曲线图

分析图 20，我们观察到时间序列的变化明显减弱了，这有助于后续我们分析其规律。下面依次绘制时间序列的自相关图（图 21）和偏相关图（图 22）。

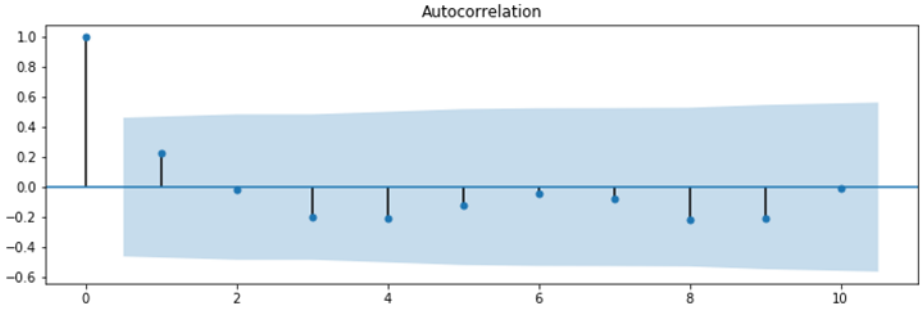


图21　时间序列的自相关图

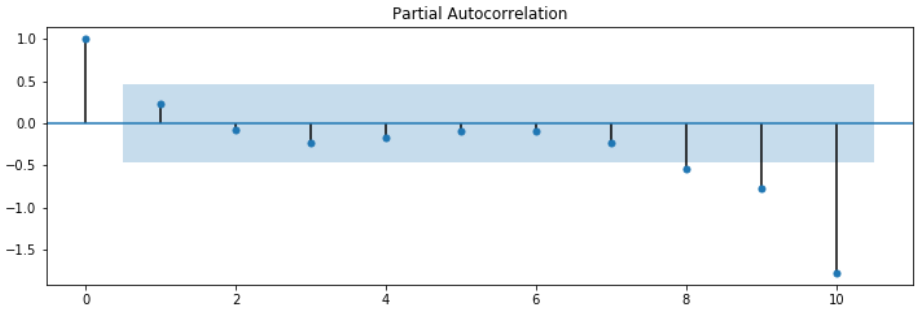


图22　时间序列的偏相关图

分析图 21 和图 22，我们可以观察到两图都没有截尾，这说明时间序列不太平稳。下面我们对时间序列进行平稳性检测（ADF 检测），结果如图 23。

	value
Test Statistic Value	-3.07328
p-value	0.0285945
Lags Used	0
Number of Observations Used	17
Critical Value(1%)	-3.88927
Critical Value(5%)	-3.05436
Critical Value(10%)	-2.66698

图23　差分前时间序列的平稳性检测结果

分析上述平稳性检测的结果，我们可以观察到 T 检验值并未小于 Critical Value(1%)，且 p 值并未远小于 0.05. 为了获得更平稳的时间序列，我们对时间序列进行差分处理。差分后的时间序列曲线图如图 24。

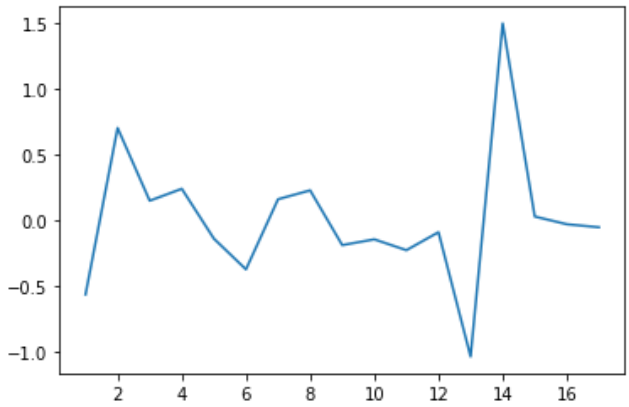


图24　差分后的时间序列曲线图

差分时间序列的自相关图和偏相关图如图 25 和图 26 所示。

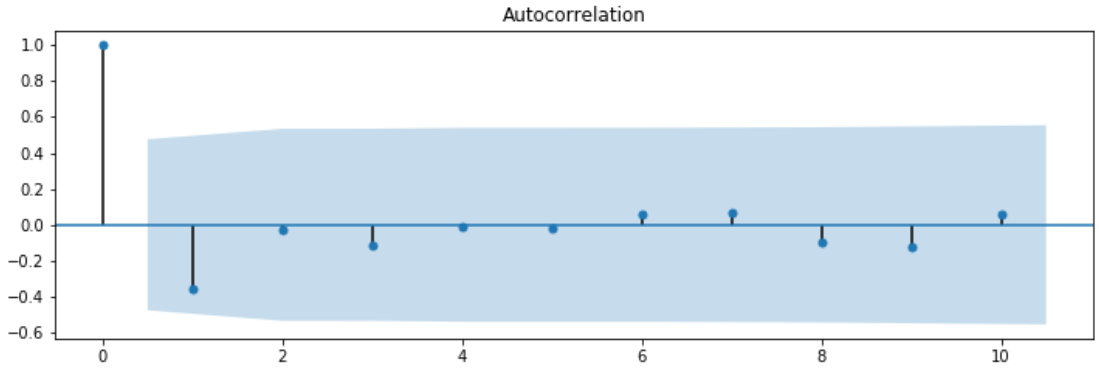


图25　差分时间序列的自相关图

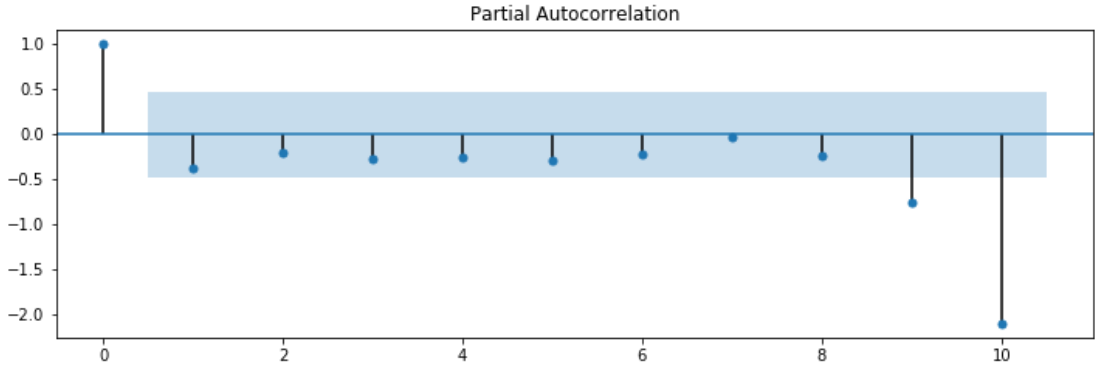


图26　差分时间序列的偏相关图

分析差分时间序列的自相关图和偏相关图，我们可以观察到两图的结果优于差分之前。这说明时间序列更加稳定。下面我们对差分时间序列进行平稳性检测（ADF 检测），如图 27。

	value
Test Statistic Value	-5.71692
p-value	7.08586e-07
Lags Used	0
Number of Observations Used	16
Critical Value(1%)	-3.92402
Critical Value(5%)	-3.0685
Critical Value(10%)	-2.67389

图27　差分时间序列平稳性检测结果

观察平稳性检测的结果，我们发现差分时间序列满足 T 检验值小于 Critical Value(1%)，且 p 值远小于 0.05.下面对一阶差分后的序列做白噪声检验，如图 28。

```
#对一阶差分后的序列做白噪声检验
#p值为第二项
from statsmodels.stats.diagnostic import acorr_ljungbox
print(u'差分序列的白噪声检验结果：',acorr_ljungbox(ddta, lags= 1)) #返回统计量和 p 值
```

差分序列的白噪声检验结果： (array([4.03499639]), array([0.04456577]))

图28　一阶差分后的序列的白噪声检验结果

检验结果是 p<0.05（图 28 中元组的最后一个元素），因此该差分序列不是白噪声序列。

下面我们通过 AIC、BIC、HQIC 准则，来判断 ARIMA 模型的阶数。对应代码如图 29。最终确定 ARIMA 模型的阶数为（*p,d,q*）=（0,1,1）

```
order=sm.tsa.stattools.arma_order_select_ic(ddta,max_ar=4,
                                max_ma=4,ic=['aic','bic','hqic'])
order
```

图29　判断 ARIMA 模型的阶数的代码段

我们使用原始序列中的前 15 个点作为训练数据，后 3 个点作为测试数据。观察 ARIMA 模型的预测效果，如图 30 所示。观察预测的数据和实际数据，我们发现基于 ARIMA 模型的时间序列分析较好的预测了未来的数据，为运量预测提供了理论支持。

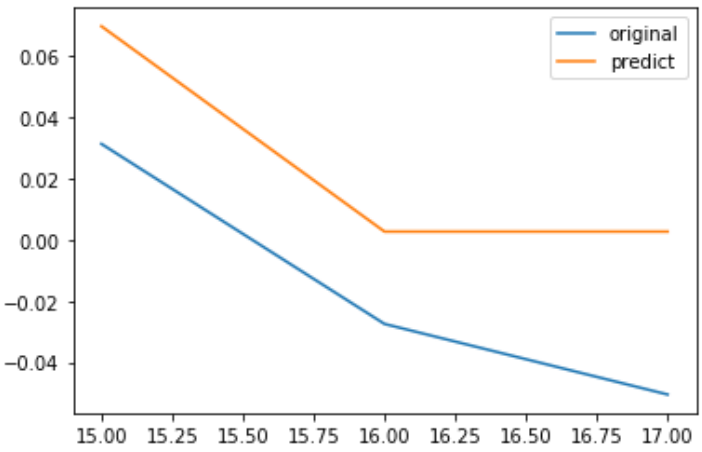


图30　ARIMA 模型在测试数据上的表现

5.2. 车货匹配

本车货匹配方案综合考虑供需匹配与地点，一共可对 3838 辆车进行进一步的权重（带有优先级权重）的推荐，即将同一起终点的所有空闲可用车辆的优先度 α 与路径系数 β 综合考虑并按比例进行排序，选出最大值即应该匹配的车辆，假如物流货运没有很紧张，一般可主要考虑空闲可用车辆的优先度 α 。

XGBoost 批量预测结果如图 31 所示，每行是某时刻同时输入的订单（以订单为单位），每列是 XGBoost 模型选择对应的概率 α ，一定程度上反映了历史的记录的影响，并具有一定的泛化能力。

需要注意的是，此处我们假定这些同起终点的订单是并发的，所以可以看出有多订单同时偏好同一辆车的情况，面对这种情况我们有以下解释：首先现实中不会有这么多同起终点运单并发；其次最终还要结合路径系数 β 综合考虑并按比例进行排序，选出最大值即应该匹配的车辆；再其次就是，假如发现选出来的车已经占用，则再进行一次预测（此时之前匹配的车已经不可用，因为它已不再空闲，所以不会进入备选名单）。

最终可以看出，本车货匹配方案结合了数学建模与机器学习 XGBoost 的方法，并最终将对每一个订单给出最匹配的车辆。

yprob - NumPy array

	0	1	2	3	4	5
70	0.107939	0.186574	0.378225	0.111648	0.107832	0.107782
71	0.116384	0.11817	0.407815	0.125149	0.116268	0.116215
72	0.121497	0.392591	0.121425	0.121788	0.121377	0.121321
73	0.410138	0.119078	0.117208	0.119306	0.117162	0.117108
74	0.396061	0.151001	0.113185	0.113524	0.11314	0.113088
75	0.118112	0.153974	0.118041	0.373939	0.117994	0.11794
76	0.129876	0.133233	0.129799	0.347656	0.129747	0.129688
77	0.103	0.239101	0.102938	0.106979	0.345132	0.10285
78	0.119524	0.121359	0.119453	0.119811	0.400502	0.119351
79	0.410856	0.119286	0.117414	0.117765	0.117367	0.117313
80	0.410856	0.119286	0.117414	0.117765	0.117367	0.117313
81	0.103386	0.239997	0.103324	0.103633	0.346425	0.103236
82	0.119524	0.121359	0.119453	0.119811	0.400502	0.119351
83	0.108138	0.186918	0.378923	0.110008	0.108031	0.107981

图31　XGBoost 批量预测结果

5.3. 线路分布

5.3.1. 历史上有一条线路但是可能绕路的推荐方案

客户 A 有在陕西咸阳提货，在陕西西安、青海西宁、甘肃兰州卸货的需求。从图 32 看到，查询到满足需求的历史运单（ID 595580）采用的路线是在陕西咸阳提货，依次在陕西西安、青海西宁、甘肃兰州卸货。但是通过查看地图可知，从陕西西安去往青海西宁的途中会经过甘肃兰州，所以这项记录所使用的路线绕路。

从图 33 看到，模型用最短路线算法推荐的路线是依次在陕西西安、甘肃兰州、青海西宁卸货。从图 34 看到，专家评议法根据评分规则，对历史路线评分 0.3，对最短路线评分 0.5，所以模型最终推荐的路线是最短路线，而且给出了路线的近似总长度是 695.8 千米和推荐星级是 3.

```
1724786,595580,提货,陕西,咸阳,礼泉县,108.48256879,34.59785379,1,0
1724787,595580,卸货,陕西,西安,未央区,108.94975291,34.33606346,0,0
1724788,595580,卸货,青海,西宁,城北区,101.75908113,36.70623786,0,1
1724789,595580,卸货,甘肃,兰州,安宁区,103.68879819,36.12210663,0,2
1724790,595580,卸货,甘肃,兰州,安宁区,103.69374355,36.11964286,0,3
1724791,595580,卸货,甘肃,兰州,永登县,103.67335175,36.53917155,1,4
```

图32　运单 595580 经过的节点信息

```
图论算法推荐路线:
陕西,咸阳(提货)->陕西,西安(卸货)->甘肃,兰州(卸货)->青海,西宁(卸货)
历史上符合查询要求的运单的路线和选择这条路线的人数
                                          track  freq  percentage
0  陕西,咸阳(提货)->陕西,西安(卸货)->青海,西宁(卸货)->甘肃,兰州(卸货)     1         1.0
系统推荐路线：陕西,咸阳(提货)->陕西,西安(卸货)->甘肃,兰州(卸货)->青海,西宁(卸货)
推荐星级：3
近似总长度:695.8 km
```

图33　案例 5.3.1 程序运行结果

Index	track	freq	percentage	score
0	陕西,咸阳(提货)->陕西,西安(卸货)->青海,西宁(卸货)->甘肃,兰州(卸货)	1	1	0.3
1	陕西,咸阳(提货)->陕西,西安(卸货)->甘肃,兰州(卸货)->青海,西宁(卸货)	0	0	0.5

图34　专家评议法对两路线评分情况

5.3.2.　历史上有多条线路给出的推荐方案

客户 A 有在陕西咸阳提货，在甘肃庆阳和宁夏银川卸货的需求。通过检索历史运单，如图 35 所示，发现有 3 个运单先在甘肃庆阳，再在宁夏银川卸货；有 2 个运单顺序相反。由于最短路径算法推荐的卸货顺序是甘肃庆阳->宁夏银川，最终根据专家评议系统评分（如图 36 所示），推荐的路线是陕西,咸阳(提货)->甘肃,庆阳(卸货)->宁夏,银川(卸货)。

```
图论算法推荐路线:
陕西,咸阳(提货)->甘肃,庆阳(卸货)->宁夏,银川(卸货)
历史上符合查询要求的运单的路线和选择这条路线的人数
                                          track  freq  percentage
0  陕西,咸阳(提货)->甘肃,庆阳(卸货)->宁夏,银川(卸货)     3         0.6
1  陕西,咸阳(提货)->宁夏,银川(卸货)->甘肃,庆阳(卸货)     2         0.4
系统推荐路线：陕西,咸阳(提货)->甘肃,庆阳(卸货)->宁夏,银川(卸货)
推荐星级：4
近似总长度:844.5 km
```

图35　案例 5.3.2 程序运行结果

Index	track	freq	percentage	score
0	陕西,咸阳(提货)->甘肃,庆阳(卸货)->宁夏,银川(卸货)	3	0.6	0.68
1	陕西,咸阳(提货)->宁夏,银川(卸货)->甘肃,庆阳(卸货)	2	0.4	0.12

图36　专家评议法对两路线的评分

5.3.3.　历史上没有运输线路给出的推荐方案

客户 B 有在浙江宁波提货，在河南郑州和广东广州卸货的需求。通过查询历史运单，发现没有符合需求的运单。系统直接推荐最短路线，线路是浙江宁波(提货)->河南郑州(卸货)->广东广州(卸货)，近似总长度 2121.1 km。因为最短路线不是历史线路，所以百分比和热门线路两项得分都是 0 分，最终总分 0.5 分，推荐星级是 3。程序运行结果如图 37 所示。

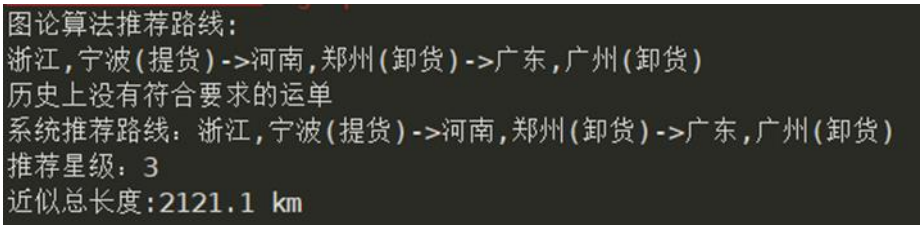


图37　案例 5.3.3 程序运行结果

6. 总结

6.1. 模型整体架构

在运量走势方面，本数据分析方案给出两地之间未来的运量预测方法。通过 python 进行数据处理，得到了运量吨数和方数的时间序列。通过基于 ARIMA 模型的时间序列分析方法，较好地实现运量预测功能。

在车货匹配方面，本数据分析方案将数学建模与机器学习相结合，通过预测结果进行加权，并考虑到车货匹配场景下的反馈率的需求以及路损，最终对每一个订单按比例对优先度 α 和路径系数 β 进行排序，最终选出最大值即推荐匹配的车辆。

在线路分布方面，本数据分析方案首先统计客户的历史线路分布，其次在给定需求的情况下能够根据历史上符合需求的路线和对应的运单数，再结合最短路线算法给出的推荐路线，最后通过对三项指标评分，给出评分最高的路线作为系统最终的推荐路线。给出推荐星级供货车司机参考，路线的星级越高代表可信度越高。

6.2. 为网络货运平台提供的参考方案

通过 ARIMA 模型我们成功预测未来客户在任意两地之间提货、卸货运量需求情况。由于比赛时间所限，本文中呈现的结果是按照月度的尺度预测运量的未来走势，但是我们会在之后的比赛中（如果有机会的话）尝试将时间粒度精确到周或者日，或者以运单为粒度预测未来何时有订单需求，例如我们预测出客户 A 有在陕西咸阳提货，陕西西安、甘肃兰州、宁夏银川卸货的需求，要求提货时间 2020 年 12 月 10 日 8：00，要求卸货时间 2020 年 12 月 11 日 23：59，方数 28，吨数 9.6。

我们将预测的运单需求输入车货匹配模型，模型会根据运单参数推荐出合适的车辆 1、2、3 等。根据反馈率等指标对车辆优先度进行排序，再根据承运时间内车辆预计的空闲情况给车辆分配运输任务，或者让车辆在运单的要求提货时间之前保持空闲，并到预测的提货地点附近等待接单。这可以大大减少车辆的空闲等待时间，提高物流资源的利用率。例如，车辆 1 的优先度 0.1，车辆 2 的优先度 0.3，车辆 3 的优先度 0.2。车辆 2 在 2020 年 12 月 8 日 12：00 到 2020 年 12 月 10 日 23：59 承担运单，无法参加预测运单的运输任务，将车辆 2 从推荐车辆列表中删除。这时推荐优先度较高的车辆 3，平台提前给车辆 3 的司机提供建议，建议其在 2020 年 12 月 10 日 8：00 之前到陕西咸阳客户 A 的提货地附近等待接单，并在运单要求提货时间之前不要接受其它运单请求。

此外根据预测的运单需求，线路分布模块通过查找历史上满足需求的运单走的线路，给出历史上较多司机选择的运输线路。结合使用最短路线算法得到的提货、卸货地点顺序，帮助货车司机更合理的安排运输路线，节约时间和能源成本。例如，前述预测需求在客户 A 的

数据库中不存在，模块根据最短路线算法的结果向车辆 3 的司机推荐的顺序是陕西咸阳(提货)->陕西西安(卸货)->甘肃兰州(卸货)->宁夏银川(卸货)，建议司机不要先去宁夏银川卸货、再去甘肃兰州卸货、最后去陕西西安卸货，帮助司机少走回头路。

综上所述，模型使用时下火热的数据挖掘和人工智能技术对客户物流数据集进行分析。通过时间序列分析预测客户的货运需求，帮助车辆匹配货运需求，实现物流资源的科学调度和合理配置，根据历史热门和最短路线原则优化运输节点顺序，缩短运输路线长度，为货运平台降本增效提供了解决方案。