

队伍编号	MC2305806
题号	D

基于 QAR 数据的实时安全分析与预警系统

摘要

当今，我国民航业快速发展，而飞行安全的重要性也不言而喻。严重的飞行事故的发生不仅会使航空公司蒙受巨大损失还会极大地威胁乘客的生命安全。那么如何强化科学管理，科学有效提高从业人员素质，及时监测和预警风险，降低飞行事故发生概率呢？为解决以上问题，本文将利用飞机上所搭载的快速存取记录器（QAR）所记录的数据进行处理分析与建模。

针对问题一，主要需要对 QAR 数据进行预处理、分析研究数据质量的可靠性、提取与飞行安全的关键性因素，并定性定量分析。由于 QAR 所记录的数据体量大、维度高，因此本文对表格以时间序列为标准进行筛选并多方面详细分析，以减少错误数据对后续建模的影响。同时，本文对附件 1 中的数据进行可靠性研究，再以主成分分析累计方差解释与层次聚类法提取关键性因素，筛选出的重要性指标见[筛选影响飞行安全的重要性指标](#)。此外，本文对上述指标进行定性及定量分析，结果见[表 5 及图 5](#)。

针对问题二，主要需要对飞行过程中机组人员对飞行的异常操作进行量化描述。为及时分析出飞机状态出现偏差的原因，本文在问题一的基础上，对 8 次飞行的数据进行筛选整理，由于着陆阶段的操纵较为关键，且较易出现超限事件，故本文重点分析 8 次航班的降落过程，并以 10 Hz 频率为间隔，选取落地瞬间前 9 秒至降落后 9 秒间重要数据，并将其可视化，进行[实时监测](#)。之后，依据时刻参量变化，本文对附件 1 的 8 次航班均的降落过程进行详细的操纵分析，结果见[模型的求解与结果分析](#)。

针对问题三，主要需要对附件 2 超限情形进行筛选处理、分析超限事件发生的各种情形、总结不同超限事件的基本特征。因此本文从“超限事件-季度”，“超限事件-飞行阶段”，“超限事件-航线”“警告级别-飞行阶段”四方面进行统计分析，从多方面分析超限的基本特征。

针对问题四，主要需要基于飞行参数对飞行员的技术进行评估。首先对附件 3 进行数据的预处理，包括但不限于缺失值处理、异常值处理、列指标分析、标签编码等。之后利用 PCA 累计解释方差确定重要程度较高的指标个数，再通过随机森林及 XGboost 模型综合得出与飞行技术重要程度较高的因素。最后建立 XGboost 多分类预测模型，预测准确率可达 85%。从而得到了基于飞行参数的飞行技术评估方法。此外，本文还对模型多项指标进行分析，效果良好，分析模型预测的合理性。

针对问题五，主要需要建立自动化智能预警机制系统，预防可能发生的安全事故。因此，本文综合上述问题的分析，建立综合多方面因素影响的自动化智能预警机制系统，且设定超限阈值，对附件 1 的 8 次航班进行全面分析，进行实时监测，最终得到仿真结果。

最后，本文对所建立的模型进行中肯评价、提出改进措施，并对模型进行一定推广。

关键词：QAR 数据；PCA-CVCR；RF-XGboost；自动化智能预警；实时监测

目录

一、 问题的提出	1
1.1 问题背景	1
1.2 问题要求	1
二、 问题的分析	1
2.1 问题的整体分析	1
2.2 问题一的分析	2
2.3 问题二的分析	2
2.4 问题三的分析	2
2.5 问题四的分析	3
2.6 问题五的分析	3
三、 模型的假设	3
四、 符号说明	3
五、 模型的建立与求解	4
5.1 问题一模型的建立与求解	4
5.1.1 附件 1 数据预处理及可靠性研究	4
5.1.2 筛选影响飞行安全的重要性指标	8
5.1.3 指标定量及定性分析	11
5.2 问题二模型的建立与求解	13
5.2.1 模型的建立	14
5.2.2 模型的求解与结果分析	14
5.2.3 飞行安全定性分析	16
5.3 问题三模型的建立与求解	16
5.3.1 多层分析模型架构的建立	17
5.3.2 不同超限基本特征统计分析	17
5.4 问题四模型的建立与求解	18
5.4.1 附件 3 数据预处理	19
5.4.2 多分类模型的建立	19
5.4.3 PCA-CVCR & RF-XGBoost 数据降维	21
5.4.4 新数据集模型的建立与求解	22
5.4.5 预测效果分析	23
5.5 问题五模型的建立与求解	25
5.5.1 附件 1 中 8 次航班各个全航段监测	26
5.5.2 附件 1 中 8 次航班综合分析	28
5.5.3 附件 1 中 8 次航班仿真	29
六、 模型的评价与推广	31
6.1 模型的评价	31

6.2 模型的推广	31
参考文献	32
附录	33

一、问题的提出

1.1 问题背景

改革开放以来，我国民航业蓬勃发展，越来越多的乘客选择乘坐飞机出行，飞行安全的重要性不言而喻。截至 2022 年 3 月 21 日，即“3.21”空难发生前，我国民航安全飞行达 1 亿零 59 万飞行小时，为我国历史最好安全记录。严重的飞行安全事故不仅会使航空公司蒙受经济损失，还威胁着乘客的生命财产安全。为科学管理，降低飞行事故发生的概率，综合现有数据进行监测并预警风险，总结出具有针对性和系统性的方案提升从业人员素质显得尤为重要。在航空安全数据分析中快速存取记录器（Quick Access Recorder, QAR）发挥着重要作用。目前我国民航业主要研究两方面：**一是**，对超限事件的研究，分析及应用；**二是**，非超限数据的统计分析及应用。其中，对于前者的分析着眼于超出阈值的部分，然而超出阈值的部分不完全是人为因素，可能为环境或飞机本身存在一定问题，若基于非人为因素对机组严以要求显然是不合理的。QAR 超限可为航空安全管理和飞行训练提供数据支撑，而少量的 QAR 超限显然不具有说服性，故挖掘 QAR 全航段数据，基于不同飞行机组，航线，机场及特殊飞行条件下的飞行记录，建立数学模型，并分析之，评估各指标风险系数，针对性开展安全培训，排除安全隐患，改进安全绩效。

1.2 问题要求

- **问题一：**由于 QAR 数据并不能保证绝对正确性，故应进行数据预处理减少错误数据干扰。在此基础上对附件 1 进行可靠性研究，提取关键数据项并分析重要程度。
- **问题二：**飞行过程往往通过一系列飞行操纵如：横滚、俯仰等以保证安全。国内航司主要以超限监控飞行动态，虽然能够快速分辨飞机状态偏差，但无法在较短时间内知道原因。为解决此问题，依据附件 1 合理量化描述飞行操纵。
- **问题三：**除人为，环境，飞机本身缺陷外等因素外，仍有一定因素会影响超限的发生。依据附件 2 分情况讨论超限并研究其基本特征。
- **问题四：**飞机运行数据研究往往由两大类组成，一类由 LOSA 获取，另一类则遵从相关学者建议，开展飞行技术评估。请依据附件 3，建立数学模型以合理分析评估飞行员飞行技术。
- **问题五：**在 QAR 实现陆空实时传输的情况下，以航司安全管理人员的身份建立实时自动化预警机制，预防可能的安全事故，并依据附件 1 给出仿真结果。

二、问题的分析

2.1 问题的整体分析

该问题是一个关于航空安全风险及飞行技术的数据分析、建立预警模型的问题。

从分析目的看，本题需要分析飞机在飞行中的各项飞行参数和航空安全风险，同时建立自动化预警机制，从而预防安全事故的发生。因此本主题需要完成两方面任务：其一，研究

飞行参数对航空安全的影响程度，并对各参数进行量化分析。其二，根据上述分析，建立合理模型，通过飞行参数对风险进行识别，改进安全绩效。确保模型的准确性、稳健性、可靠性，并有一定的泛化能力。

从数据来源、特征看，本题的数据来源于某航司 2013-2017 年随机快速存取记录器(QAR)。数据包括 2014 年 7 月 5 日至 2014 年 10 月 11 日部分时段某些航线完整飞行数据；2015 年至 2016 年间相关航线超限数据；2013 年至 2017 年间 A 机型落地主操作人员资质及相关飞行数据。飞行及超限数据量较大且影响飞行及超限数据的因素具有存在无效值、高维、复杂等特征。因此本题数据较为特殊且复杂，需对数据进行预处理，便于后续问题的分析。

从模型的选择看，本题数据量较大、维度较高，且需要分析多趟航班整体情况、基于飞行参数评估机组人员技术水平及多方面指标，因此本文选择 PCA 模型进行降维，利用层次聚类进行综合分析指标，利用随机森林、XGBoost 模型进行多分类的预测学习。

从编程软件的选择看，本题为大数据分析类，需要进行大量的数据预处理、数据分析、数据可视化，并依据各设问建立预警自动化智能预警机制，因此我们选择 Python Jupyter 对问题进行求解，其交互式的编程范式及轻量化，方便且高效。

2.2 问题一的分析

问题一的核心目的有以下几点：其一，对真实的 QAR 数据进行预处理，去伪存真；其二，**分析研究附件一数据质量的可靠性**；其三，**提取与飞行安全的关键性因素，并定性及定量分析**。对于已给的数据集，数据在真实性、完整度、指标标准等方面存在一定缺陷，这导致我们在原始数据上不可直接进行分析，因此需要对其进行相应的预处理。此外附件 1 为 8 次航班的由起飞到降落的全过程的 QAR 数据，数据体量大，因此我们需要由特殊到一般，建立普适性模型，高效分析多张数据表格；同时我们还发现，数据维度较高，为得到关键性因此，考虑累计方差解释、层次聚类分析及熵权法，确定重要度较高指标，并对筛选出的指标进行合理性分析。

2.3 问题二的分析

问题二的核心目的在于**对飞行过程中机组人员对飞行的异常操作进行量化描述**。由于在飞行过程中，航空公司只能监测到操纵动作，虽然能分辨出是否出现偏差，却无法得出原因。因此我们需要对操纵进行量化描述。通过预处理后附件一的数据，我们绘制出了八个航线飞机的状态和操纵随时间变化的曲线图，通过曲线的突变，来合理分析发生偏差的原因。

2.4 问题三的分析

问题三的核心目的有以下几点：其一，**基于附件 2 中的数据对所有超限情形进行筛选，分类处理**；其二，**分析超限事件发生的各种情形**；其三，**总结研究得出不同超限的基本特征**。对于所给定的数据集，数据多且杂乱无章，我们无法直接分析，故我们对附件 2 中的数据进行了筛选，剔除了空白和未知部分的无效数据，而后对各超限情形进行了分类，共分为 6 类。考虑到数据集提供了日期，起飞机场，目的机场等数据，故我们从季节，航线，机场等角度

进行了统计，为便于分析，我们将相关情形做出可视化处理并建立模型，而后对其进行研究以得出不同超限的基本特征。

2.5 问题四的分析

问题四的核心目的在于**基于飞行参数对飞行员的技术进行评估**。但是附件 3 与附件 1 同样在数据完整度、指标标准等方面存在一定缺陷，因此我们也需要对其进行一定的预处理；同时，我们还发现附件 3 维度更高，飞行参数较多，对于模型的效率有一定影响，因此我们沿用问题一的想法，以累计方差解释确定重要因素较高的指标的个数，再以随机森林及极端梯度提升算法综合分析出与飞行员飞行技术重要程度较高的因素。但我们也发现，附件 3 中，飞行员资质为“C”类的占比仅为整体的 0.844%，因此我们综合多方面考虑，选择将该两项数据单独分析，剩余数据视为多分类预测。最后以筛选出的重要指标为自变量，飞行员的“不同资质”，即不同技术级（除“C”类）别为因变量，建立 XGboost 多分类预测模型，从而建立出基于飞行参数的飞行技术评估方法。同时，为探讨模型效果，我们绘制出模型的分类混淆矩阵热力图、分类报告、ROC/AUC 曲线等对预测结果进行合理性分析。

2.6 问题五的分析

问题五的核心目的在于**建立自动化智能预警机制系统，预防可能的安全事故**。因此，我们综合问题一、二、三、四的分析，建立综合多方面因素影响的自动化智能预警机制系统，设定超限阈值，对附件 1 的 8 次航班进行全面分析，实时监测，最终得到仿真结果。

三、模型的假设

- **假设一：**假设问题中所涉机型为适航的普通民航机型及普通小型飞机。
- **假设二：**假设机上所载 QAR 设备处于理想环境，即处于正常工作状态。
- **假设三：**假设飞行数据已能实现陆空实时传输，延时低。

四、符号说明

符号	符号说明
μ	样本平均数
σ	样本标准差
x_{standard}	经过标准化后的数据
$R(x)_{m \times n}$	经过某项处理后的数据特征集
\hat{y}	预测值
$L^{(t)}$	目标函数
ω	权重

注：这里并未列出部分变量，这是由于它们在不同的章节处有不同的含义，因此我们在每一节中详细讨论它们。

五、模型的建立与求解

对于本题，本文模型的建立与求解部分主要分为数据的准备，模型的建立、求解、结果分析。

- **数据的准备：**对于给定的数据集进行预处理，方便后续模型的建立，以及多次航班的规范分析。
- **模型的建立、求解、结果分析：**对于给定的数据集，本文依据其特点，建立合适的模型，研究并量化分析影响飞行安全的因素。此外还需要分析飞行阶段操纵杆的过程变化情况，分析安全性。同时，还需要依据飞行参数对驾驶员飞行技术进行预测，并解释预测的合理性。最后需要结合上述问题，建立自动化智能预警机制，预防可能的安全事故的发生，给出仿真结果。

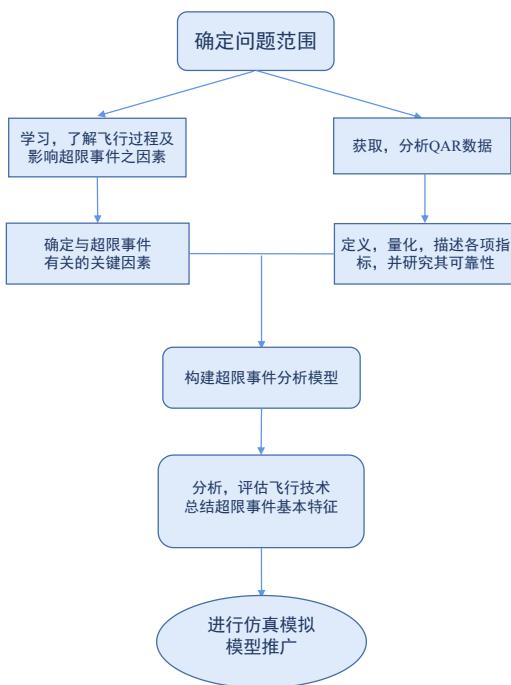


图 1 问题解决流程

5.1 问题一模型的建立与求解

对于问题一，我们首先分析数据的特点，依据时间特征及对应的各参量进行合理性分析，对错误值进行剔除，提高数据的真实性，并以此分析数据的可靠性；之后我们依据刘柳^[1]、龙海江^[2]学者的研究结成果，采用层次聚类法对多维度指标进行聚类分析，同时在此基础上利用主成分分析累计方差解释，确定影响飞行安全的关键性指标，并利用熵权法对其重要性进行量化分析。此外注重定性及定量的分析研究，结果与现实情况相近，选取结果良好。

同时为了下文叙述方便，我们对附件 1 的 8 张表格进行命名，见表 1。

5.1.1 附件 1 数据预处理及可靠性研究

通过对附件 1 的 8 个 Excel 表格依此分析，并结合快速存取记录器（Quick Access Recoder, QAR）数据的特点，我们发现以下几点可能的错误方面：

表 1 附件 1 表格命名航班对应

文件名	201404070532	201404071917	201404080617	201404081034
航班号	航班 1	航班 2	航班 3	航班 4
文件名	201404090110	201404091701	201404100843	201404101159
航班号	航班 5	航班 6	航班 7	航班 8

- 起始时刻不为飞机开始运行时刻:** 通过对 8 张表 8 次航班的全航段记录数据的逐一分析，我们发现表格“201404091701”记录器开始记录的时刻为 2014 年 4 月 9 日 14 时 23 分 51 秒，而该航班实际运行时间应该为 2014 年 4 月 9 日 17 时 01 分 50 秒，因此我们认为该表格的起始时刻不应为前者，而应是飞机开始运行的时刻。因此我们将该表格的起始时刻校正，即删除 14 时 23 分 51 秒的数据，保留 17 时 01 分 50 秒之后的数据。而对于其余表格并未发现相同错误。
- 相邻数据存在时刻上的重复且后续指标不一致:** 利用 Python 的 pandas 库，对 8 张表格统一分析，发现所有表格均存在该方面的问题，即存在某同一时刻的数据，但后续指标却不一样，如表 2 所示，这是明显的 QAR 数据记录错误，究其原因，可能是由于该时刻被 QAR 连续记录两次，但指标发生变化，但为了保证数据在时刻上的连续性，我们以这些重复时刻数据首次出现的为标准，将其保留，而另一条数据选择剔除。

表 2 表格“201404101159”时刻重复值（部分列）

月	日	具体时间	海拔高度	下降率	无线电高度	计算空速	地速
4	10	12:18:16	12751	-3144	1404	310.625	372.25
4	10	12:18:16	12804	-3154	1404	310.75	372.5
4	10	13:11:44	30103	2	1404	305.625	465.25
4	10	13:11:44	30105	-10	1404	305.625	465.25

- 出现两列一致指标，且其下所有时刻数据一致:** 我们发现附件 1 的 8 次航班数据中均出现两次“俯仰角率”列数据，为了探讨其下所有时刻数据是否一致，我们利用 pandas 对其进行分析，发现该两列数据无任何差别，造成数据冗余，因此我们将其中一列剔除。

经过上述处理后，各表格行数据剔除率及数据保留率如表 3 所示。各表格列数据均剔除一列。

表 3 附件 1 各表格数据剔除、保留率（名称省略年份 2014）

Rate	070532	071917	080617	081034	090110	091701	100843	101159
剔除率	0.000284	0.000299	0.000301	0.000286	0.000315	0.000357	0.000429	0.000268
保留率	0.999716	0.999701	0.999699	0.999714	0.999685	0.999643	0.999571	0.999732

为了再次验证数据在时刻层面的连续性，即每行记录的数据均间隔为 1 秒，我们对所有航班进行时刻点计算，均与处理后的数据记录条数一致，从而也反映出保留下的数据的合理性。但此处，我们并未对其他方面进行分析，如异常值等，也并未对其进行剔除，这是因为我们查阅相关资料，结合李瀚明^[3]学者及胡占桥^[4]老师的研究，及对实际数据的分析，给出下几点原因：

- **传感器存在一定误差**, QAR 误差可分为三种: 多报、漏报、误报。多报即为生成了额外的数据点, 造成数据冗余; 漏报即为未生成相应时刻的数据点; 误报即为某些点因 QAR 仪器, 造成数据采集有误;
 - **QAR 存在一定局限性**, 记录的数据可能会有少部分产生问题, 但浮值不会变化过大, 这是由仪器本身所决定, 而时刻数据 (上述分析的多报) 即为其局限性的一方面;
 - **部分异常值可能为机组人员误操作造成**, 即我们很难在一定程度上区分误报与机组人员误操作造成异常值记录的数据, 若我们将其剔除, 则会使原始数据在信息解释方面造成一定失真, 对数据进行了“篡改”, 从而影响后续分析。

因此在这里，我们不考虑部分异常值，而尽可能保留数据的真实性，而对于该数据，我们也将 在问题五进行详细叙述。

同时我们发现，附件 1 的 8 张表格，前三行均属于表头类型，且有部分字段重复，若将其直接利用 pandas 读取，可能会对后续处理产生一定影响，因此我们根据附件 1 的字段中文说明，将 8 张全航段表格数据表头进行处理，且保证所有表格表头一致，方便后续处理。

此外，通过读取附件 1 数据，查看其空缺值情况，发现“起落架”¹该列数据缺失值较多，这里我们以附件 1 中表格“201404101159”为例²，其空缺值情况如图 2 所示。



图 2 附件 1 航班 8 缺失值

这里，我们结合实际分析，认为空缺值均应填充为“NON-DOWN”，即此时起落架为收起状态，理由如下：

¹这里指标名称已由原数据的英文修改为中文指标。

²对于问题一，在本文正文，我们均以附件1中表格“201404101159”为例进行分析，后文简称“航线8”，其余表格的分析结果也将在正文及附录中呈现。

- “起落架”该列数据存在的均为“DOWN”，即起落架状态为放下状态，结合具体时刻，我们可以发现为飞机运行的开始阶段与将要结束阶段，因此中间的缺失值视为在空中正常飞行阶段；
- 结合“海拔”“空地电门”等多项列数据进行分析，可以验证上述我们的猜想，即中间段“起落架”的缺失值均为“NON-DOWN”。

因此在这里我们首先以该值进行填充，而其余处理也将在后续展开叙述。此外，我们发现在该列数据缺失，其余列数据均无一缺失，则无需再进行填充处理。

同时，我们还注意到数据表中“起落架”“空地电门”“是否接通了 A/T”“是否接通了任意侧的 A/P”“起飞机场”“落地机场”指标的数据均为文本类型数据，对于后续的分析及模型的建立有一定影响，因此我们这里采用字典方法，对其键值对人为对应替换，替换结果如表 4 所示。其中“起飞机场”与“落地机场”列数据，我们选择直接剔除，而将在后续与“月”“日”一起进行定性分析。

表 4 附件 1 指标替换

指标	替换
起落架	DOWN:1, NON-DOWN:0
空地电门	True:1, False:0
是否接通了 A/T	DISENGD:0, ENGAGED:1
是否接通了任意侧的 A/P	OFF:0, ON:1

最后，我们发现附件 1 的维度较高，且各维度数据量纲不一，对后续模型的建立会产生较大影响，因此这里我们对所有数据进行标准化及归一化处理，其区别、后续利用方式及计算方式如下：

- 数据标准化：**采用 Z-score 方法处理，用于主成分分析（Principal Component Analysis, PCA）累计方差贡献率（Cumulative Variance Contribution Rate, CVCR）及层次聚类（Hierarchical Clustering, HC）模型的建立。同时该标准化处理方法适合当代嘈杂的大数据场景^[5]。因此对于大样本的数据，如出现部分异常值，使用该方法对最终结果影响较小。其计算方式如下：

对于某一列数据 $x = [x_1, x_2, \dots, x_m]^T$ ，其平均值为

$$\mu = \frac{1}{m} \sum_{i=1}^m x_i \quad (1)$$

标准差为

$$\sigma = \sqrt{\frac{1}{m} \sum_{i=1}^m (x_i - \mu)^2} \quad (2)$$

则标准化后的数据为

$$(x_{Z\text{-score}})_i = \frac{x_i - \mu}{\sigma} \quad (3)$$

- **数据归一化:** 采用 **Min-Max** 方法处理, 用于熵权法 (Entropy Weight Method, EWM) 模型的建立。其处理方法计算公式如下³:

$$(x_{\text{Min-Max}})_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} \quad (4)$$

这样分开分别处理有以下几点原因:

- **Z-score** 方法, 使得原数据经过处理后, 数据集中于 0 附近, 即均值为 0, 标准差为 1。其适用于 PCA-CVCR 及 HC 模型的建立, 大大提升模型的效率。
- **Min-Max** 方法, 使得原数据经过处理后, 在设定的最大值与最小值之间, 方便 EWM 的建立, 避免对数运算的错误。

5.1.2 筛选影响飞行安全的重要性指标

对于问题一, 我们建立 PCA-CVCR、HC 及 EWM 模型, 分析及讨论过程如下。

主成分分析累计方差贡献率 (Principal Component Analysis Cumulative - Variance Contribution Rate, PCA-CVCR) : 由于附件 1 所有数据集较为复杂, 且为高维数据, 而 PCA 作为一种可以将原始高维数据转换为新的低维数据的方法, 故我们采用此方法对数据集进行处理。在此我们将对 PCA 累计方差解释率作出解释。在 PCA 过程中, 各主要成分所占方差比例的累加和, 往往以百分比的方式呈现。PCA 累计方差解释率对选择合适的主成分进行降维起到至关重要的作用。在 PCA 降维中我们往往基于 PCA 累计方差解释率的高低来选择一定数量的主成分以代表原始数据的特征, PCA 累计方差解释率越高, 说明前几个主成分所能代表的方差比例越高, 便可选择这些主成分降维以保留更多信息。其计算方式如下:

根据标准化后的数据集计算协方差矩阵

$$C = \begin{bmatrix} \text{Cov}(X_1, X_2) & \text{Cov}(X_1, X_2) & \cdots & \text{Cov}(X_1, X_n) \\ \text{Cov}(X_2, X_1) & \text{Cov}(X_2, X_2) & \cdots & \text{Cov}(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(X_n, X_1) & \text{Cov}(X_n, X_2) & \cdots & \text{Cov}(X_n, X_n) \end{bmatrix} \quad (5)$$

其中, 每一个元素定义如下^[8]

$$\text{Cov} = (X, Y) = E \{ [X - E(X)] [Y - E(Y)] \} = E(XY) - E(X)E(Y) \quad (6)$$

其中 $E(X)$ 为该列数据的均值。

之后, 计算上述协方差矩阵的特征值 λ_n 及其对应的特征向量 u_{nj} , 这里我们不妨令 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$

³此处仅为理论公式, 而在 EWM 模型建立中将进行更深层次叙述。

$\lambda_2 \geq \dots \geq \lambda_n \geq 0$, 由特征向量组成 n 和新的指标变量

$$\begin{cases} y_1 = \mathbf{u}_{11}x_1 + \mathbf{u}_{21}x_2 + \dots + \mathbf{u}_{n1}x_n \\ y_2 = \mathbf{u}_{12}x_1 + \mathbf{u}_{22}x_2 + \dots + \mathbf{u}_{n2}x_n \\ \vdots \\ y_3 = \mathbf{u}_{1n}x_1 + \mathbf{u}_{2n}x_2 + \dots + \mathbf{u}_{nn}x_n \end{cases} \quad (7)$$

其中 y_n 是第 n 个主成分, 再计算各个主成分的贡献率 γ_n 及累计贡献率 α_p

$$\gamma_n = \frac{\lambda_n}{\sum_{i=1}^n \lambda_i} \quad (8)$$

$$\alpha_p = \frac{\sum_{i=1}^p \lambda_i}{\sum_{i=1}^n \lambda_i} \quad (p \leq n) \quad (9)$$

通过上述步骤, 我们绘制出航线 8 各指标参量的累计方差解释图, 如图 3 所示。

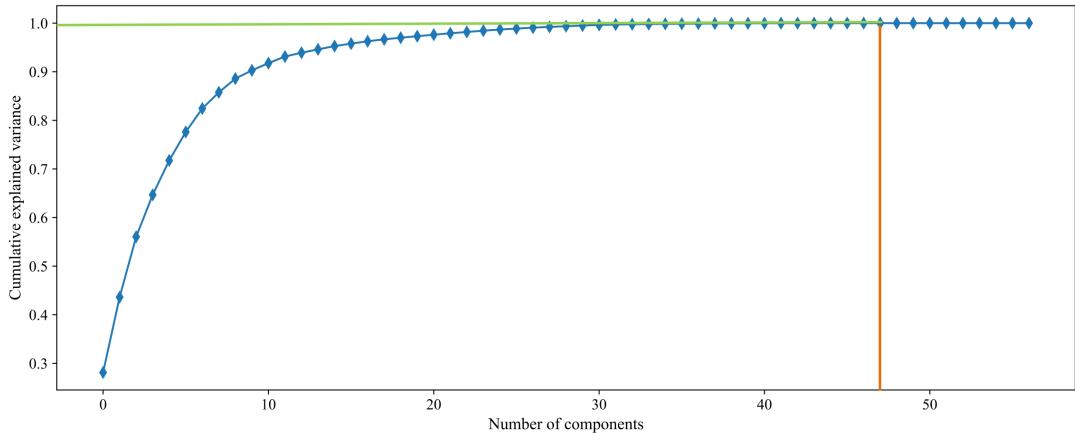


图 3 航线 8 各指标参量的累计方差解释

依据上图, 我们可以发现在指标个数为 47 个左右时, 指标对于信息的累计方差贡献率趋近于 100 %, 因此, 初步分析后, 我们将在后续选择近似 47 个指标进行综合分析。

层次聚类 (Hierarchical Clustering, HC) 是基于簇间的相似度在不同层次上分析数据, 从而形成树形的聚类结构, 采用自底向上策略。首先将每个对象作为单独的一个原子簇, 然后合并这些原子簇形成越来越大的簇, 直到所有的对象都在层次的最上层。^[9] 我们定义类与类间相似度度量: 若有两个样本 G_1 与 G_2 , 则可用 “ward” 方法, 即离差平方和法度量他们之间距离:

$$D_1 = \sum_{x_i \in G_1} (x_i - \bar{x}_1)^T (x_i - \bar{x}_1) \quad (10)$$

$$D_2 = \sum_{x_j \in G_2} (x_j - \bar{x}_2)^T (x_j - \bar{x}_2) \quad (11)$$

$$D_{12} = \sum_{x_i \in G_1 \cup G_2} (x_k - \bar{x})^T (x_k - \bar{x}) \quad (12)$$

其中

$$\bar{x}_1 = \frac{1}{n_1} \sum_{x_i \in G_1} x_i \quad (13)$$

$$\bar{x}_2 = \frac{1}{n_2} \sum_{x_j \in G_2} x_j \quad (14)$$

$$\bar{x} = \frac{1}{n_1 + n_2} \sum_{x_k \in G_1 \cup G_2} x_k \quad (15)$$

因此，可以得到

$$D(G_1, G_2) = D_{12} - D_1 - D_2 \quad (16)$$

根据上述计算方法，我们绘制出层次聚类树状图，如图 4 所示。⁴

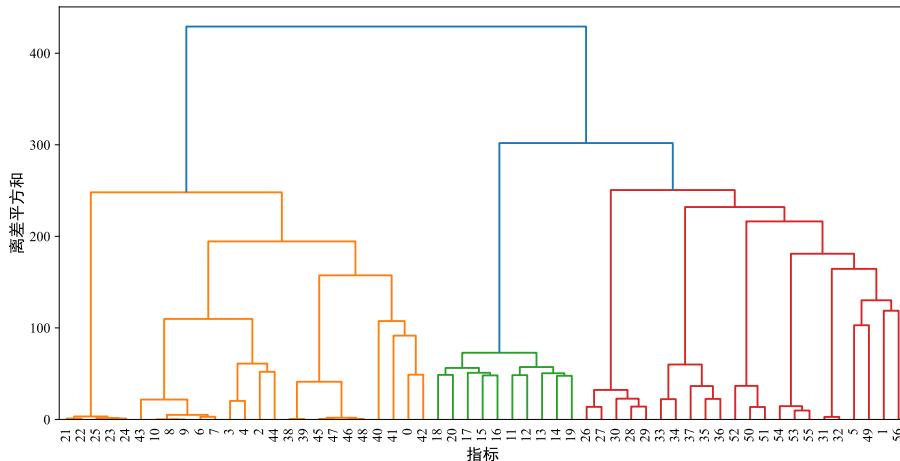


图 4 各指标参量的层次聚类树状图

根据以上结果，并结合附件 1 的其余 7 次航班分析结果，我们最终确定“人-机-环”评级体系，即三方面：人为操作、机器状态、环境干预。并依据社会实际情况，确定了最终三个层面的共计 47 个指标，但实际上，一部分指标是一种指标不同时刻的量值。因此，在这里我们将其进行合理性合并，最终得到三个层面共计 19 项与飞机飞行安全高度相关的指标，即：

- **环境干预：**风向、风速；
- **人为操作：**下降率、地速、起落架、着陆 G 值、杆量、盘量、RUDD 位置；
- **飞机状态：**姿态（俯仰角）、航向道偏差、磁航向、左侧发动机油门 N1 值、右侧发动机油门 N1 值、下滑道偏差、左发油门杆位置（角度）、右发油门杆位置（角度）、坡度（左负右正）、俯仰角率。

⁴由于数据维度过高且字段过长，我们将其列为阿拉伯数字，而最后结果也将在后文提及。

5.1.3 指标定量及定性分析

下面我们将对上述重要程度较高的因素进行量化分析，这里我们使用熵权法进行分析。此外我们还需要对前文叙述的部分因素进行定性分析。

熵权法 (Entropy Weight Method, EWM) 是一种指标客观影响程度的量化方法。当信息熵越大时，信息的无序程度越大，此时，信息价值越小，指标权重就越小^[6]。其计算步骤如下：

- **Step 1:** 指标正向化。由于数据集构成的指标类别不一，部分指标可能数值越大越好，部分指标可能越小越好，而有的可能在某一点取值最优，为方便、高效评价，我们需要进行指标正向化处理^[7]。其处理方法如下

– 越大越优指标

$$x'_{ij} = x_{ij} \quad (17)$$

– 越小越优指标

$$x'_{ij} = \max(x_{ij}) - x_{ij} \quad (18)$$

– 在 β 处取值最优指标

$$x'_{ij} = 1 - \frac{|x_{ij} - \beta|}{\max(|x_{ij} - \beta|)} \quad (19)$$

- **Step 2:** 数据标准化。由于数据集构成的指标数据数量级存在差异、量纲不一，为消除上述情况对结果的影响，我们需要将各指标进行标准化处理，这里我们使用 Min-Max 方法。处理方法计算公式为

$$r_{ij} = \frac{x'_{ij} - \min(x'_j)}{\max(x'_j) - \min(x'_j)} \quad (20)$$

- **Step 3:** 计算信息熵。进行上述处理后可得到由特征数据构成的矩阵 $R(r_{ij})_{m \times n}$ ，对于某一项指标的数据 r_j ，其信息熵为

$$E_j = -\frac{1}{\ln m} \cdot \sum_{i=1}^m p_{ij} \ln p_{ij} \quad (21)$$

其中

$$p_{ij} = \frac{r_{ij}}{\sum_{i=1}^m r_{ij}} \quad (22)$$

观察到(22) 式中分母不可为 0，且(21) 式对数真数部分不能为 0，因此，我们在进行 Step2 时，标准化区间的最小值设为 0.002，可避免计算时的不合定义。

- **Step 4:** 计算指标权重。其计算公式为

$$\omega_j = \frac{1 - E_j}{\sum_{j=1}^n (1 - E_j)} \quad (23)$$

在这里，我们需要注意，附件 1 并未给出此刻飞行的安全状态，即属于无标签类型数据。因此，我们在这里无须对其进行评价分析，而是获取其特征性数据，用于后续筛选重要性指标。从而对于上述的 Step 1 我们无须进行，即直接进行 Step 2 至 Step 4 的计算。

通过上述步骤，我们分别计算出三层方面下各自因素的权重，并对其余 7 次航班均作一致处理，得到各航班不同指标的权重，再计算平均值得到最终量化结果。此外，我们还计算出其方差与标准差，确定置信区间以及分析量化结果的合理性，结果见表 5。同时我们还绘制权重组合图，如图 5 所示。

表 5 影响飞行安全的 19 个重要指标从属于三个层级的量化值结果

层级	指标	均值	标准差	方差	量化值
环境干预	风速	0.7307	0.0643	0.004133	0.7307 ± 0.0643
	风向	0.2693	0.0643	0.004133	0.2693 ± 0.0643
人为	起落架	0.8944	0.0994	0.009880	0.8944 ± 0.0994
	地速	0.0485	0.0493	0.002432	0.0485 ± 0.0493
	着陆 G 值	0.0327	0.0029	0.000008	0.0327 ± 0.0029
	下降率	0.0087	0.0074	0.000055	0.0087 ± 0.0074
	杆量	0.0082	0.0034	0.000011	0.0082 ± 0.0034
	盘量	0.0067	0.0012	0.000002	0.0067 ± 0.0012
	RUDD 位置	0.0008	0.0012	0.000002	0.0008 ± 0.0012
	姿态（俯仰角）	0.3080	0.0120	0.000143	0.3080 ± 0.0120
飞机	航向道偏差	0.2379	0.0230	0.000529	0.2379 ± 0.0230
	磁航向	0.1480	0.0976	0.009526	0.1480 ± 0.0976
	左侧发动机油门 N1 值	0.0805	0.0215	0.000462	0.0805 ± 0.0215
	右侧发动机油门 N1 值	0.0793	0.0200	0.000400	0.0793 ± 0.0200
	下滑道偏差	0.0577	0.0127	0.000164	0.0577 ± 0.0127
	左发油门杆位置（角度）	0.0344	0.0033	0.000011	0.0344 ± 0.0033
	右发油门杆位置（角度）	0.0343	0.0033	0.000011	0.0343 ± 0.0033
	坡度（左负右正）	0.0175	0.0033	0.000011	0.0175 ± 0.0033
	俯仰角率	0.0024	0.0007	0.000000	0.0024 ± 0.0007



图 5 影响飞行安全的 19 个重要指标从属于三个层级的量化值结果

同时我们还对前文所提及的指标进行定性分析，分析如下：

- 飞机重量：**如果飞机的重量过轻，首先会导致飞机稳定性受到影响，飞机的重量对于保持稳定而言至关重要。当飞机重量过轻时，它将变得更加敏感，更容易受风等环境因素

的影响，从而减小其稳定性和控制性能。其次飞机安全受挑战，一些重要的设备如降落伞、救生设备等会增加飞机的重量，但是它们对安全至关重要。如果去除这些设备，飞机的安全性将面临严重挑战。同样，着陆过程，飞机重量过轻，它所需要的升力就会更小，因此空速相对更小，拉平着陆时，一旦收油门，飞机的减速效果就会很显著，平飚能力较弱，为了维持平稳安全的下降率触地，需要增大接地仰角。如果仰角增加不足，容易快速失去升力导致落地 G 值大，甚至是重着陆，仰角增加过快过大则会增加擦机尾的风险。因此在着陆拉平触地这一过程，对飞行员在配合使用油门和俯仰操纵上有更高的要求。在另一方面，如果客机的重量过重，也会产生相类似的问题：飞机稳定性和控制力受到影响，并增加机身疲劳的风险。因此，保持适当的客机重量是确保飞机安全和经济效益的关键一步。

- **起飞机场、落地机场：**从全球已发生的飞机事故统计数据来看，起飞和着陆阶段约占总飞行时间的 6%，但发生事故的比例却高达 63%。

起飞时易发生安全事故的原因之一是当飞机加足马力达到起飞速度时，飞机以每小时几千米的速度向前行进，面对瞬息万变的情况，无论出现什么情况，哪怕是失火，都必须继续起飞，因为剩余的跑道已不够让飞机安全停下，按照飞机高速运动状态下的惯性作用，它很容易直接冲出跑道，一旦冲出跑道就会造成不可挽回的局面，当然，这就要求起飞机场在跑道的末端安装有跑道阻拦装置，用飞机的重力作用将飞机起落架卡住，从而阻止飞行，继续向前逼停飞机。

落地机场的跑道是影响机场运行的关键因素之一，只有精确地把握飞机的跑道占用时间、着陆距离等重要参数，结合相应的管制规则，才能真实地反映飞机在跑道上的运行动态，让飞机安全降落。飞机的标准着陆过程可以分为 5 个阶段：进近拉平段、第一过渡段、减速段、第二过渡段、脱离段。当落地机场跑道长度较短时，容错率低，对机长的操作技术要求高，易于发生安全事故。同时，落地机场的班次也对落地安全也有一定的影响，对于班次多的机场，飞机在降落过程中更容易受到其他飞机的干扰，更易于发生安全事故。

- **时间：**在不同时间，如一年中的不同季度，当日时间等都在一定程度上会影响到飞行的安全。这是由于不同的环境可能对机组人员的操作产生一定影响。对于该方面的分析，我们将在问题三中详细叙述。

5.2 问题二模型的建立与求解

飞机飞行过程通常划分为：起飞、爬升、巡航、下降、进近和着陆六个阶段^[15]。上述过程中，进近与着陆是整个飞行过程中最关键也是最容易出现不安全事件的阶段。有以下几点原因：

- **环境干预：**此时飞行易受到该降落机场所在区域的环境，具体分析已在前文叙述，此处不再赘述，见[指标定量及定性分析](#)。
- **人为操作：**由于环境的影响，人为操作可能受到一定影响，如飞行员的精神状态、操作技术等。同时在进近过程中，机组人员需要将机身与降落跑道对齐，使得飞机此时需要

减速，修正方向等，从而多方面的一系列操作会对飞行安全产生一定影响。此外在降落过程中， G 值显得尤为重要，若 G 值过大，会造成明显的重着陆，这样不仅对飞机设备产生影响，也会对全机人员安全产生不良影响。

- **飞机状态：**在上述两个过程中，此时飞机需要调整的指标过多，其状态不稳定，同时飞机状态会受到环境、人为操作的影响，此外，飞机状态又会反作用于人为操作，因此该三方面是相互关联，相互制约的。

因此，对于问题二，我们在问题一的基础上，以 10 Hz 为频率，选取接地前 9 秒及接地后 9 秒主要相关数据，共计 19 秒数据，精确到 0.1 秒。共 8 组数据，即 8 次航班，依此建立“着陆监控预警”模型，依据模型对降落过程中的飞行操纵进行合理性定性与定量描述。

5.2.1 模型的建立

首先我们依据 QAR 数据特点，以时间序列为自变量，并结合问题一的重要性指标，选取“杆量”“盘量”“着陆 G 值”“姿态（俯仰角）”及“坡度”为主要因素建立了“着陆监控预警”模型并将各航班着陆过程量值可视化。这里我们以“空地电门”“降落杆”等因素来判断飞机落地瞬间。因此在后学绘制可视化监控图时，我们以降落时刻所在秒为 0 时刻，则落地前为负时刻，滑行阶段为正时刻。

5.2.2 模型的求解与结果分析

由于 8 次航班数据过多，绘制图形篇幅过大，因此在正文部分我们展示具有代表性的航班 1、航班 3、航班 7、航班 8，其余图示可在附录中查看。

- 针对“航线 1”，预警可视化图见图 6。我们发现在航线 1 中着陆的前 3 秒和后 1 秒盘量有明显下降，坡度发生较大变化，我们分析认为是在降落过程中遇到突发情况，如遇到大风、雨雪等恶劣天气，使得飞行机组被迫做出调整以防止偏离航线。

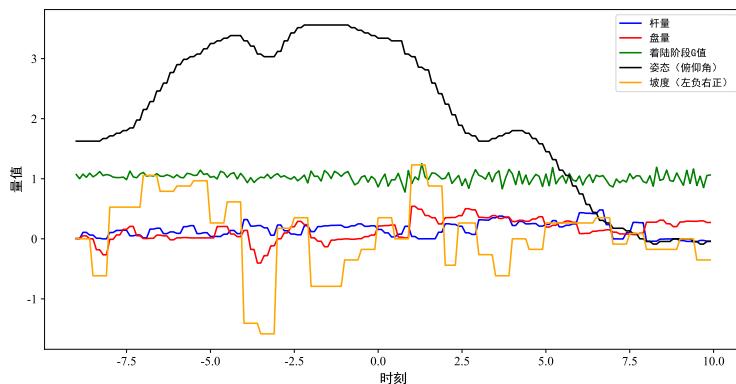


图 6 航班 1 着陆过程指标量值实时监视

- 针对“航线 2”，预警可视化图见图 31。此模型中着陆过程中杆量总体变化较小，但其姿态变化较快，我们分析认为，此次降落过程中可能出现了降落速度过小或顺风降落致使升力不足的情况。

- 针对“航线 3”，预警可视化图见图 7。此模型在落地前飞行基本处于平稳状态，但在落地后，杆量先发生较小的变化，在之后 4 秒内，杆量发生连续突变，我们分析其原因是飞机发生了弹跳，机组为了控制飞机而不断调节杆量。

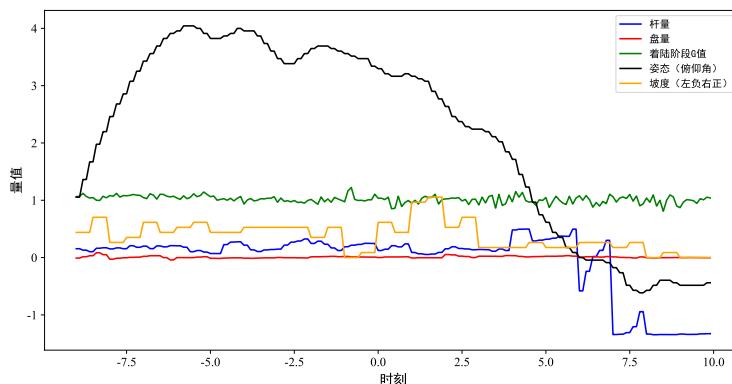


图 7 航班 3 着陆过程指标量值实时监视

- 针对“航线 4”，预警可视化图见图 32。航线 4 杆量变化较为平缓，高度下降较为缓和，但在降落前 7.5s 至降落后 2s 内，其飞行姿态始终保持向左偏转的状态，我们认为其遭遇了较强横风或类似恶劣天气，破坏了飞机的稳定状态。
- 针对“航线 5”，预警可视化图见图 33。此次降落在 0-2.5s 的时间段内总体下降较为平缓，但在 2.5s 至 5.0s 这一时间段内下降突然加快，我们分析认为这可能与人为干预有关，如：50ft 至接地距离过远，飞行员须尽快使飞机接地，并脱离跑道。
- 针对“航线 6”，预警可视化图见图 34。此模型从杆量和盘量的角度来看变化幅度较小，总体较为平稳。但我们发现，其坡度变化较为明显，我们分析认为其在降落过程中飞行速度较慢同时遭遇大风或顺风情形，致使飞机摆动，在 7.5s 之后，即起落架完全接地后杆量发生了较大突变，我们认为此次着陆过程中接地偏重。

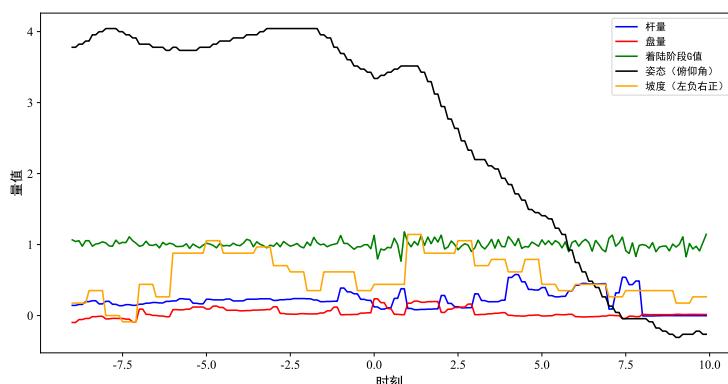


图 8 航班 7 着陆过程指标量值实时监视

- 针对“航线 7”，预警可视化图见图 8。在此次着陆过程中，我们发现在起落架完全接地后杆量并未发生明显变化，此次降落过程较为平缓。值得注意的是，此次航班降落过程

中整体向右偏转，我们认为飞机在降落过程中遭遇了较强横风，一定程度上影响了飞机的坡度，干扰飞行员正常的飞行操作。

- 针对“航线 8”，预警可视化图见图 9。在此次着陆过程中，飞行员选择了保持相对稳定的姿态（俯仰角）着陆，从杆量变化来看，在飞机完全接地时，杆量变化量极小，可见此次降落的接地过程较为平稳，进近过程受环境影响大，飞机摆动较大。

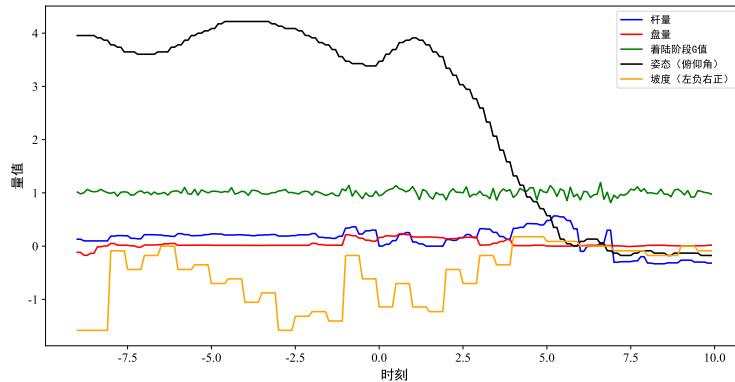


图 9 航班 8 着陆过程指标量值实时监视

5.2.3 飞行安全定性分析

在这一部分中我们将就现有数据分析可能促使超限事件发生的原因。

我们比较了 2015 及 2016 年某地区航线中发生超限事件总数，如下图所示。显然，两者差别并不明显，而若将其细化至各季度，我们发现两者一二季度超限事件总数均明显高于三四季度超限事件总数。我们认为该地区冬春季节气候较为复杂，可能存在如：天气多变；降雨、降雪多；温度偏低等情况。而这些情况往往会导致一些威胁飞行安全的情况发生，如：风切变；降雨云团阻挡进近路线，机翼和尾翼在穿越积雨云或厚积云后发生结霜现象，显著影响飞机的空气动力学特性，升力减小，据研究表明在上述情况中总阻力值增加 70 % ~ 80 %；风挡模糊或结冰，飞行员目视距离不足；跑道湿滑等。这些往往会导致“50 英尺对接地距离远”“进近速度/坡度大”“着陆垂直载荷大”“离地速度大”等现象。对数据进一步筛选，我们发现超限事件往往发生在起飞、降落、进近这三个飞行速度较慢的过程，上文所提及可能的现象也恰巧属于这三类之内。据现有数据，此三类事件占超限事件总数的 74.14 %，一二季度占其中 53.49 %。我们不妨做一个比较：2015 年一二季度较三四季度多出 827 起⁵，2016 年一二季度较三四季度更是多出了 1338 起，这些现象足以证明不同的季度会对飞行员的飞行操作造成一定的影响，造成超限事件的发生。综上所述，除人为因素外，时间（季度）因素也是影响飞行安全的重要原因之一。

5.3 问题三模型的建立与求解

对于问题三，我们建立了“超限事件-季度”，“超限事件-飞行阶段”，“超限事件-航线”“警报级别-飞行阶段”分析模型架构，尽可能多维分析超限的不同情况及基本特征。

⁵该处数据来源于附件 2，对于该部分的分析，在问题三中，我们也将进行详细叙述

5.3.1 多层分析模型架构的建立

附件 2 中主要提供了“警告级别”“机号”“目的机场”“时间日期”“起飞机场”“超限名称”“飞行阶段”以及“月份”的数据，首先我们对数据进行了预处理，剔除超限名称中“未知”及空白部分的无用数据集，而后我们比较了各因素之间的关联性，发现超限事件与季度（时间）、飞行阶段、航线存在一定联系，警告级别与飞行阶段存在一定联系，故我们剔除其余关联度较低的指标“机号”“月份”，其余因素视为关键因素，建立相关模型。

5.3.2 不同超限基本特征统计分析

对于“超限事件-季度”，我们首先分别选取 2015 年及 2016 年的第一季度、第二季度、第三季度和第四季度，分别统计两年四个季度的超限事件总数，建立“超限事件-季度”模型并将其可视化，见图 10。我们分析得出 2015 年和 2016 年存在一定的共性：两年均出现一二季度超限事件总数远远多于三四季度，故我们认为当地一二季度存在较强影响飞行安全的因素，超限事件的发生与季度存在一定关联。

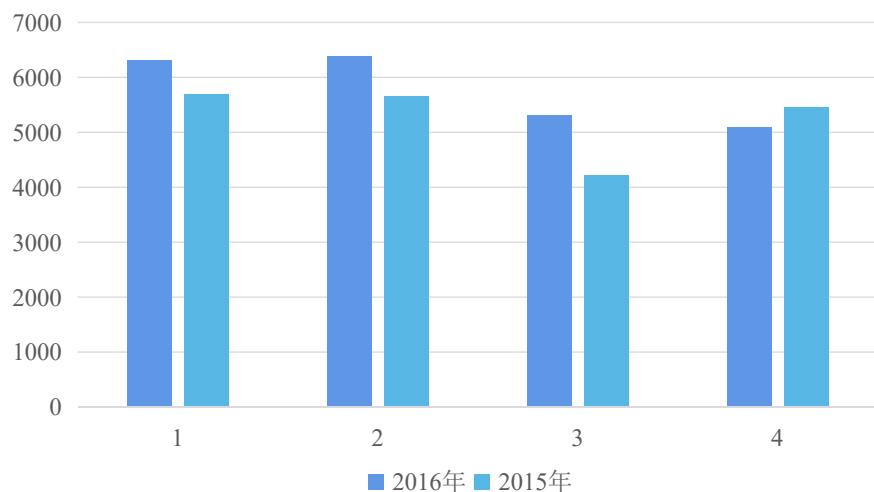


图 10 超限事件-季度统计分析结果

针对“超限事件-飞行阶段”，我们分别统计了着陆、空中、进近、地面（起飞及着陆）阶段的超限事件总数，其中着陆 26406 次，空中 11119 次，进近 5591 次，（着陆）地面 398 次，（起飞）地面 158 次，起飞 148 次，见图 11。分别占比 60.2 %, 25.38 %, 12.77 %, 0.92 %, 0.37 %, 0.36 %，通过该模型我们发现超限事件主要发生在着陆、空中、进近过程中，显然超限事件与飞行阶段有较强的关联。同时我们还绘制空中、进近及着陆阶段超限事件词云图，如图 13~图 15 所示。

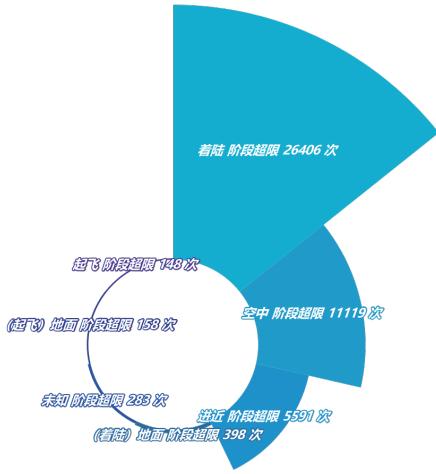


图 11 超限事件-飞行阶段统计分析结果

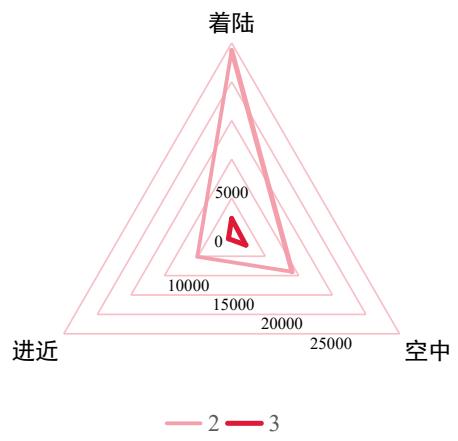


图 12 警告级别-飞行阶段统计分析结果

对于“超限事件-航线”，我们分别统计了每个航线发生超限的次数，发现机场 68, 机场 89, 机场 27 所在的航线中发生超限的次数远超其他航线，次数都超过了 1000，机场 68 发生的超限次数更是达到了 20000 次。我们由此分析出个别航线对超限发生的次数有一定的影响，这与航线中目的机场和起飞机场的安全管理，环境，航班密度有关。



图 13 空中超限事件



图 14 进近超限事件



图 15 着陆超限事件

针对“警告级别-飞行阶段”，我们对发生超限最多的三个飞行阶段的警告等级占比进行了绘图，如图 12 所示。我们发现对于不同飞行阶段，各警告等级占比不同，其中空中阶段的高级别警告占比最大，达到了 19%，这反映了在空中发生安全事故的危险程度更高，空中发生的超限更容易产生严重的后果，同时空中发生的超限次数占比较大，所以对航空安全来说，保护空中飞机平稳飞行格外重要。

5.4 问题四模型的建立与求解

对于问题四，我们首先附件 3 的数据特点，我们发现附件 3 由 2367 行数据与 197 列指标组成，是基于飞行参数对机组人员飞行技术探讨的大体量、高维度的数据。首先，我们需要对原始数据表进行预处理，包括空缺值、异常值、重复值、标签编码等操作，从而获得纯净数据。之后观察数据，进行初步降维，剔除方差较小的列数据，提升模型的预测效率。之后对数据进行标准化处理，通过 PCA-CVCR 的分析，确定保留指标的个数，再通过随机森林 (Random Forest, RF) 及极端梯度提升 (eXtreme Gradient Boosting, XGBoost) 模型，对数据建立多分类预测模型，并分析各指标的重要程度，进行第二次降维，选择合理指标，建立最终 XGBoost 模型，并分析模型的效果，计算模型的准确率、平均绝对误差、均方误差以及均

方根误差，同时进行 5 折交叉验证，以及绘制模型的分类报告、混淆矩阵热力图、ROC/AUC 曲线等，综合评价模型的效果。

5.4.1 附件 3 数据预处理

对于附件 3，我们通过以下步骤对数据进行预处理：

- **空缺值处理：**通过观察，我们发现该数据集存在多列数据大部分缺失，综合分析，我们确定 20% 阈值进行剔除列数据，即若该列数据缺失率超过 20%，则剔除该列数据，经过统计，共计 52 个列数据；同时，若该列数据缺失率小于 20%，采用众数填充的方式进行。
- **部分列数据剔除（初步降维）：**通过观察，我们发现该数据集存在多列其下各自数据一致的情况，分别是“机型”“TO Gate 1”“TO Gate 2”“TD Gate 3”“TD Gate 2”“TD Gate 1-1”“TO_Vr”，同时包括“落地主操控”，这些指标对我们预测机组飞行技术影响很小，趋近于 0，因此，我们将其剔除。
- **标签编码：**为了让模型更多地挖掘数据隐含的信息，我们需要对数据形式为字符型数据进行便签编码，分别是“V2_Method”“Vref_Method”“RoD_Method”“MACH_Method”“落地主操控人员资质”，这里我们利用 Python 中 sklearn 库 preprocessing 模块的 LabelEncoder 进行处理。
- **数据标准化：**为了避免数据量纲对模型的影响，我们需要对数据进行 Z-score 标准化，处理过程同前文分析，见[附件 1 数据预处理及可靠性研究](#)。

5.4.2 多分类模型的建立

这里我们建立随机森林 (Random Forest, RF) 及极端梯度提升 (eXtreme Gradient Boosting, XGBoost) 模型，过程如下分析。

随机森林 (Random Forest, RF) 是由多棵决策树 (Decision Tree) 进行组合后对预测结果投票或取均值的一种算法^[10]。其有分类和回归两种模型，对于本题，我们选择分类模型。算法简图如图 16 所示。

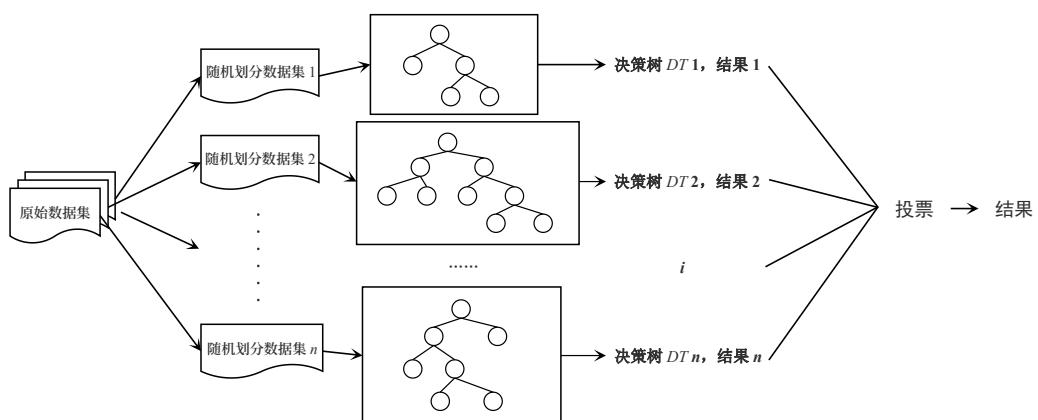


图 16 随机森林算法简图

极端梯度提升 (eXtreme Gradient Boosting, XGBoost)。XGBoost 算法是一种基于树模型的优化模型，其将弱分类器组合，训练出一个较强的分类器。该算法通过多次迭代，生成一个新的树模型用于优化前一棵树模型，随着迭代次数的增多，该模型的预测精度也会相应提高^[11]。

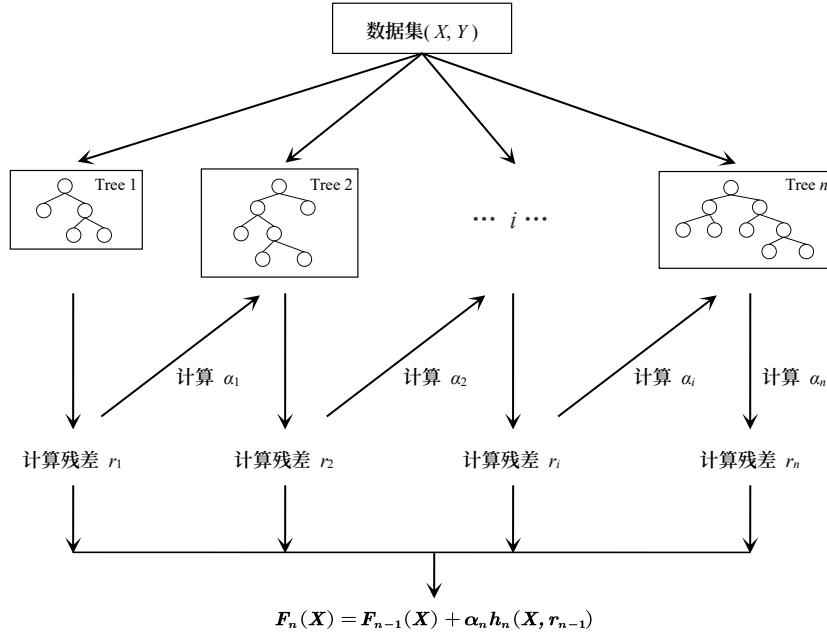


图 17 XGBoost 算法简图

记通过数据处理后的数据集特征为 $R(x_{ij})_{m \times n}$ ，表示其包含 m 个用户， n 个特征，在训练中形成的 CART 树的集合记为 $F = \{f(x) = w_{q(x)}, q : \mathbf{R}^n \rightarrow T, w \in \mathbf{R}^T\}$ ，其中 q 为树模型的叶节点决策规划， T 为某一树模型叶节点数量， w 为叶节点对应的得分^[12]。对于预测的 y 值，其计算公式为

$$\hat{y} = \varphi(x_i) = \sum_{k=1}^K f_k(x_i) \quad (24)$$

XGBoost 算法在每一次迭代过程中会保存前面所学习的模型，会将这些模型加入到新一轮迭代过程中，因此我们记第 i 个模型为预测结果为

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i) \quad (25)$$

XGBoost 算法的目标函数计算公式如下

$$L^{(t)} = \sum_{i=1}^n l \left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i) \right) + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2 + \text{const} \quad (26)$$

上述公式中， l 为模型误差损失，描述在该模型下预测值与实际值之间的出差异损失， Ω 为模型叶节点的正则项惩罚系数， γ 与 λ 为模型的超参数^[12]。通常情况下，我们难以用枚举法得到在模型中所训练出来的树结构，因此这里采用贪婪算法，从单叶子节点开始，通过迭代方

法，将其加入到树结构中，从而得到最优解，其计算公式^[13]如下

$$\mathcal{L}_{split} = \frac{1}{2} \left[\frac{\left(\sum_{i \in I_L} g_i \right)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{\left(\sum_{i \in I_R} g_i \right)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{\left(\sum_{i \in I} g_i \right)^2}{\sum_{i \in I} h_i + \lambda} \right] - \gamma \quad (27)$$

其中 $I_j = \{i | q(x_i) = j\}$ 为叶节点 j 上的样本集合^[12]，且有

$$g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)}) \quad (28)$$

$$h_i = \partial_{\hat{y}^{(t-1)}}^2 l(y_i, \hat{y}_i^{(t-1)}) \quad (29)$$

通过上述分析，我们可以得到 XGBoost 算法简图，如图 17 所示。

对于该数据集，我们利用得到的预处理后的数据进行训练、测试⁶，上述模型效果见表 6。

表 6 RF 与 XGBoost 初次学习效果

模型	准确率
RF	0.6893
XGBoost	0.8049

5.4.3 PCA-CVCR & RF-XGBoost 数据降维

为了提升模型的效果，提升学习、预测效率，我们采用 PCA-CVCR&RF-XGBoost 数据降维进行特征工程，具体过程如下：

- **PCA-CVCR:** 该模型用于对原数据进行累计方差贡献率解释，确定指标个数，在上文已经提及，这里不再赘述，见[筛选影响飞行安全的重要性指标](#)。
- **RF-XGBoost:** 该组合模型综合上述两模型，绘制重要程度排序及量化重要程度，从而确定影响机组飞行技术评估的重要性因素。

经过上述操作，我们首先得到 PCA-CVCR 图示，见图 18。

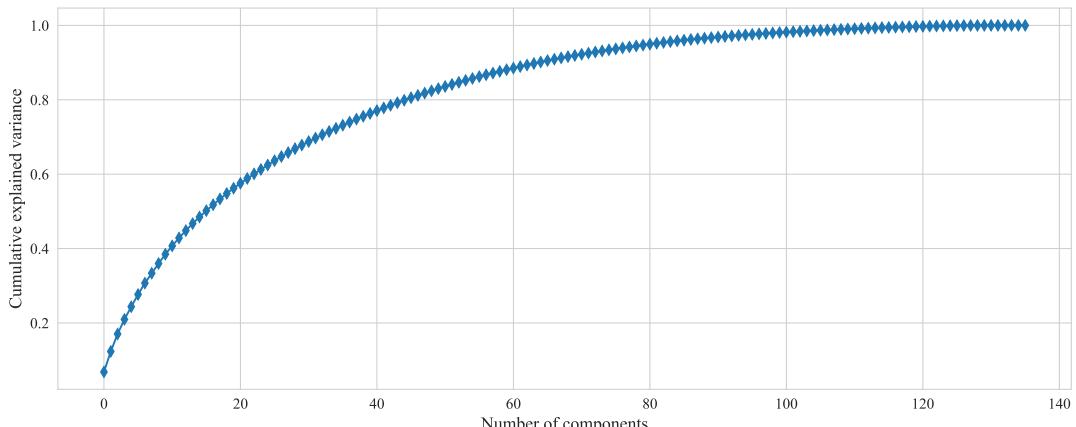


图 18 影响机组飞行技术评估的指标累计方差解释

⁶这里我们划分训练与测试集比例为 9 : 1。

通过观察上图，我们可以发现，该数据集维度高，在指标个数为 120 左右时，累计解释方差才逐渐收敛于 1.0，则对于该数据，大多对飞行参数指标均对机组飞行技术的评估有一定影响，因此仅从该方面，我们还尚且不能确定保留下的指标数及对应指标，因此这里我们再依据[多分类模型的建立](#)，得到各指标对评估的影响量化程度，对于 RF 排序见图 19，对于 XGBoost 排序见图 58。⁷

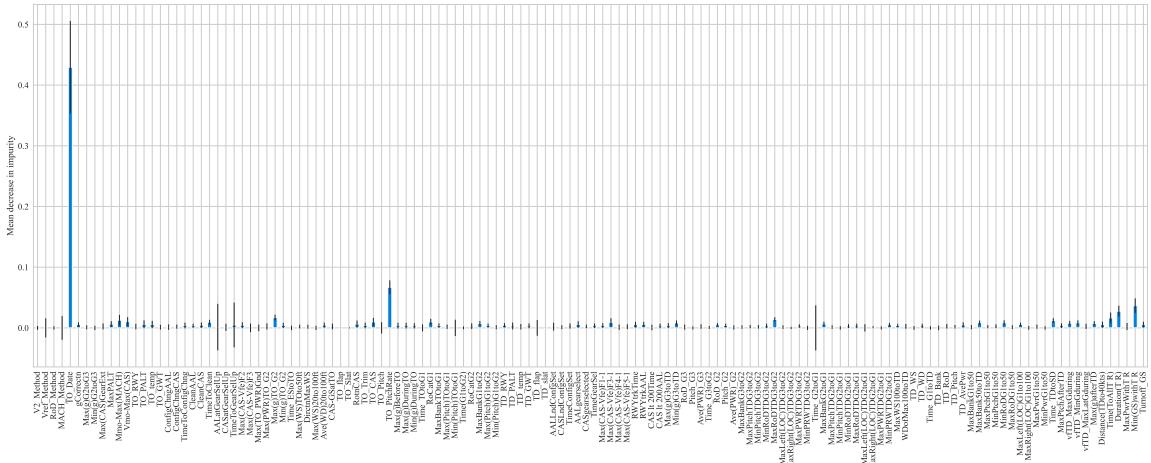


图 19 RF 重要度排序

根据上述两图结果，我们发现对于大多数指标其对机组飞行技术的评估影响均在一致范围内，仅有少数重要性程度非常低，因此我们着重分析 RF 与 XGBoost 的排名后 10 个指标，综合分析，最终选择剔除 5 列指标，分别为“CASgearselected”“MinPRWTDG2toG1”“MaxLeft-(LOC)TDG2toG1”“V2_Method”“MaxRight(LOC)TDG3toG2”。

通过上述处理后，原数据集从 196 维度降维至 127，这与我们先前利用 PCA-CVCR 分析的保留指标在误差范围内一致。

5.4.4 新数据集模型的建立与求解

通过上述降维处理后，我们再次利用 RF 及 XGBoost 模型，对新数据集进行训练、测试，这里我们比例划分依旧为 9 : 1。但我们还发现对于“不同资质”这一列资质为 C 的落地主操控人员只有两人，仅占附件 3 所给人员的 0.0844%，数据量过少，且两条数据时间间隔非常久，利用价值较低，若只根据这两个数据建立出的资质的评价标准模型，得到的结果准确性低。因此为平衡多分类样本，我们将 C 类剔除，保留 A、F、J、M、T 类别，利用 XGBoost 模型进行多分类预测学习。

通过分析降维处理后，模型精度有一定提升，较原先提升 5.8915%。同时，我们计算出模型的准确率（Accuracy）、平均绝对误差（Mean Absolute Error, MAE）、均方误差（Mean Square Error, MSE），最终结果见表 7。对于多分类模型，该三项指标，计算公式如下：

• 准确率

$$\text{Accuracy} = \frac{N_{\text{TruePredict}}}{N_{\text{Sample}}} \quad (30)$$

其中， $N_{\text{TruePredict}}$ 为预测正确的样本数， N_{Sample} 为被预测的样本总数；

⁷由于 XGBoost 重要度排序图示篇幅过大，因此我们将其放置于附录中。

- 平均绝对误差

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (31)$$

- 均方误差

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (32)$$

- 均方根误差

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (33)$$

表 7 XGBoost 模型最终效果

评价指标	初始	最终	提升 (较原先)
准确率	0.8049	0.8523	5.8915 %
平均绝对误差	/	0.2954	/
均方误差	/	0.7173	/
均方根误差	/	0.8469	/

但仅仅通过上述指标来判定模型的效果显然有片面的方面，因此在下文我们对模型进行五折交叉验证、绘制模型的分类报告、混淆矩阵热力图、ROC/AUC 曲线，综合评估模型效果。

5.4.5 预测效果分析

五折交叉验证 (Cross-Validation, CV)。具体结果见表 8。

表 8 XGboost 模型五折交叉验证结果

CV=5, 准确率							均值	方差
0.8099	0.8052	0.7981	0.8380	0.8122	0.8127	0.0136		

分类报告。 分类报告包括每一类别的精确率 (Precision)，召回率 (Recall)，F1 分数值 (F1-Score)。对于这三项值，其计算公式如下：

- 精确率

$$Precision = \frac{TP}{TP + FP} \quad (34)$$

- 召回率

$$Recall = \frac{TP}{TP + FN} \quad (35)$$

- F1 分数值

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \quad (36)$$

根据上述(34)式、(35)式、(36)式，我们可以计算出模型对于每一类别的三项指标值，并绘制分类报告图，见图 20。



图 20 XGBoost 分类报告

混淆矩阵热力图。该图示的主对角线数据之和为模型预测准确的样本数。对于多分类模型，我们可以随机指定一类为正类，而其余就为对应的负类。这里我们需要引入四项值，分别为 TP 、 FN 、 FP 、 TN ，其中 T 为 True， F 为 False，这两个字母表示预测值与实际值是否相同； P 为 Positive， N 为 Negative，这两个字母表示预测出的是属于正类（阳性）还是负类（阴性）。而混淆矩阵热力图即为这些值组成，该图示可以直观地观察到预测准确与错误的情况，以及模型对于每一类别的区分程度，见图 21。

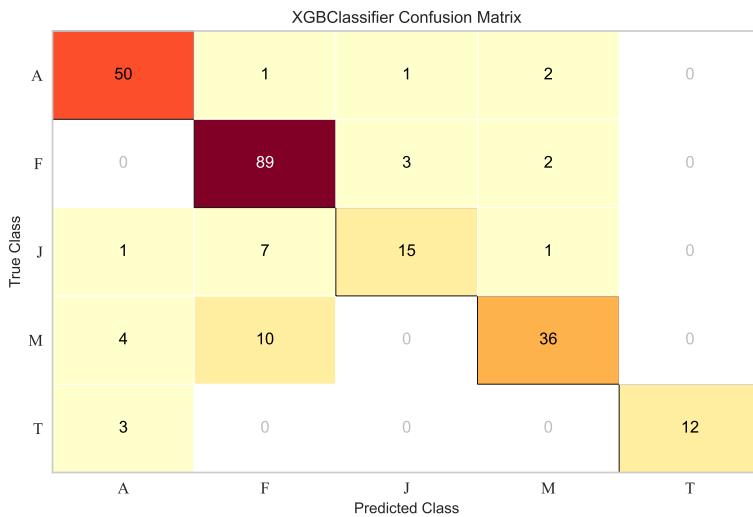


图 21 XGBoost 混淆矩阵热力图

ROC/AUC 曲线。在分析特征曲线及曲线下面积(Receiver Operating Characteristic/Area Under the Curve, ROC/AUC) 图之前，我们需要了解相关参数：

- **灵敏度 (Sensitivity)**。灵敏度又被称为真阳性率，即 TP 率，定义为：

$$\text{Sensitivity} = TPR = \frac{TP}{TP + FN} \quad (37)$$

- 特异性 (Specificity)。特异性又被称为真阴性率, 即 TN 率, 定义为:

$$\text{Specificity} = TNR = \frac{TN}{TN + FP} \quad (38)$$

- 1-Specificity。称为假阳性率 (False Positive Rate, FPR), 定义为:

$$FPR = 1 - \text{Specificity} = \frac{FP}{FP + TN} \quad (39)$$

- 1-Sensitivity。称为假阴性率 (False Negative Rate, FNR), 定义为:

$$FNR = 1 - \text{Sensitivity} = \frac{FN}{FN + TP} \quad (40)$$

FPR 和 FNR 均对数据分布的变化不敏感^[14], 因此这两个指标可以用于在不平衡的数据上建立的模型效果的评价。

该可视化以每一类别的 1 – Specificity 即 FPR 为横坐标, 以 Sensitivity 即 TPR 为纵坐标, 其可体现出模型的灵敏度与特异性之间的关系与差异。因此, 该图的理想点位于左上角, 即 $FPR = 0$ 且 $TPR = 1$, 换言之, 当曲线越靠近左上角, 模型效果就越优。从而, 我们可以得到另一项指标, 即曲线下面积 (Area Under the Curve, AUC), 由上述分析可知, AUC 值越高, 模型的整体效果也就越优。见图 22。

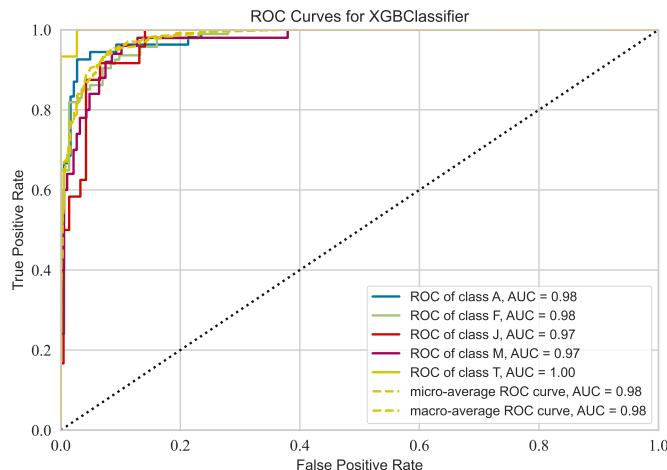


图 22 XGBoost ROC/AUC 曲线

综合分析上述评价结果, 我们可以发现基于飞行数据对机组飞行技术的评估模型效果良好, 无论是模型精度、平均绝对误差、均方误差还是均方根误差都较优秀, 模型泛化能力较强, 分类效果优秀。

5.5 问题五模型的建立与求解

对于问题五, 我们需要综合问题一、二、三、四分析的结果。

- 问题一: 得到的关键性指标因素;

- **问题二：**对横滚操纵、俯仰操纵量化分析结果；
- **问题三：**对超限事件的统计分析；
- **问题四：**基于飞行参数对机组人员飞行技术的评价的结果。

5.5.1 附件 1 中 8 次航班各个全航段监测

在问题二中，我们分析了在一次航班中，最容易出现事故的是在着陆阶段，因此在问题二中，我们着重分析飞行的着陆阶段，绘制飞行参数时刻变化监测图。而这里，我们需要对所有航班实施自动化监测，因此，在这里，我们需要绘制各飞行参数全航段变化量值，这里我们以航班 8 可视化为例，其对应部分参数如图 23 ~ 图 28 所示。

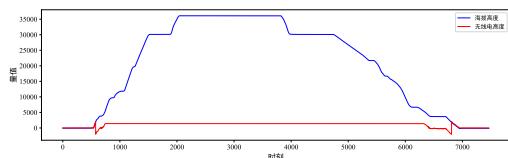


图 23 第 8 次航班海拔高度与无线电高度

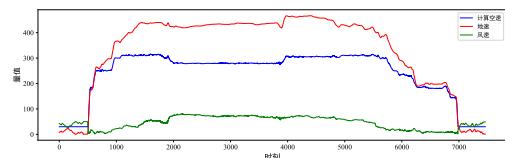


图 24 第 8 次航班空速、地速、风速

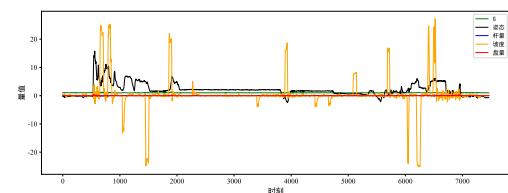


图 25 第 8 次航班飞行参量

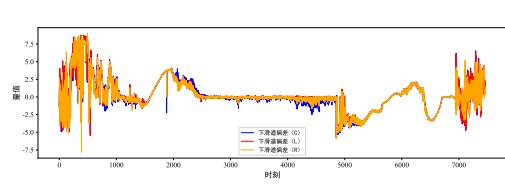


图 26 第 8 次航班下滑道偏差

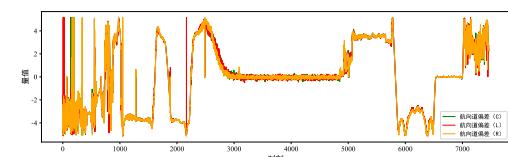


图 27 第 8 次航班航向道偏差

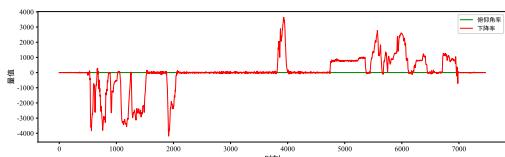


图 28 第 8 次航班俯仰角率、下降率

通过对所有航班的全航段参量实时监测分析，我们可以发现：

- **航班 1：**在飞机启动后重量从 340000 到 260000 平稳下降，在起飞和降落过程中下降率发生波动，同时也产生的航向道和下滑道的偏差，但在飞行中也出现了一次航向道与下滑道的偏差，我们分析其原因是机组的误操作。在整个过程中机组对盘量的改变次数较多，使得坡度不断发生变化，同时，G 值基本稳定不变，说明此航班对杆量的控制较好，未出现重着陆等情况。速度和海拔在飞行过程中基本不变。
- **航班 2：**在飞行过程中重量平稳下降，起飞和降落时下降率发生变化，飞机在中途飞行过程未出现航向道和下滑道的偏差。相比航班 1，航班 2 在飞行过程中盘量改变次数较少且幅度较小，坡度未发生较大变化，G 值也一直处于平稳状态。速度与海拔同样一直很稳定。

- **航班 3:** 在飞行时重量同样稳定下降，但起飞时发生了两次下降率的波动，我们推测是飞机在起飞后遇到了强气流。此航班的航向道和下滑道偏差十分严重，我们分析其原因是飞机在空中遭遇突发事件，如原航道有恶劣天气，机组不得不暂时偏离航线。G 值在飞行过程中保持平稳，说明机组人员在操作中没有严重的失误。速度与海拔保持平稳。
- **航班 4:** 其重量在飞行过程中稳定下降，在起飞后也遇到了如强气流等突发情况，使飞机的下降率产生持续波动，由于飞机受到突发情况影响较多，使飞机不断调整飞行状态，这也导致了在过程中航向道和下滑道发生了严重的偏差，但 G 值一直很稳定，我们分析其原因是机组人员应对突发情况的应变能力强，及时做出了调整。速度和海拔发生了轻微的波动。
- **下降率，俯仰角率:** 航线 5 滑行时间较短，除起飞，降落的下降率变化外，在巡航阶段下降率发生了较大突变，我们认为飞机可能遭遇较强乱流，或为 QAR 数据记录出现问题。

航线 6 与航线 5 一致，经短暂滑行后起飞，而后在巡航阶段我们发现有两处下降率突变，第一处较第二处变化小，分析认为，机组共遇到两次乱流或恶劣天气，其中第二次乱流较为明显，且在下降率明显变化后发生了持续抖动，机长可能做出了解除自动驾驶操作，手动接管飞机的操作，待飞机飞行平稳后，恢复自动驾驶。

航线 7 通过分析时间，我们认为这是一架小型飞机，其起飞和降落时间较普通民航客机时间长，巡航时间较短。

航线 8 与航线 7 一致，由普通小型客机执飞。其在巡航阶段做出降低高度操作，我们认为其为避让恶劣天气如：暴雨云团，强风等。

其中航线 5, 6, 7, 8 在降落阶段，观察其下降率曲线，我们发现出现了多处峰值，我们认为机长普遍采取了分阶段降落的措施。

- **航向道偏差:** 航线 5 由于普通民航客机飞行速度普遍较快且重量较重，起飞后转弯前往航路点，或在飞行中遇到转弯情形时，往往为了安全和旅客舒适度不会做出急转弯操作，故起飞后一段时间内时往往回偏离航道而后按航线飞行。这便解释了为何航线 5 会在起飞后一段时间后出现偏离航向道。

航线 6，我们认为此条航线为某机场至另一机场的直飞航线，其在起飞后航线道偏离极小。

航线 7，由于小型飞机飞行速度较慢，其需要花较长时间飞抵预定航向道，同样需要较普通民航客机提前脱离巡航阶段，这便解释了为何其巡航时间较短。

航线 8，与航线 7 一致。

我们发现航线 5, 6, 7, 8 飞行过程中在初始阶段和降落阶段有较大的航向道偏差，此为跑道滑行部分。

- **下滑道偏差:** 对于航线 5, 6: 如今普通民航客机的电气化程度较高，多为机器或中央电脑控制，其下滑道偏差较小。

航线 7：该小型机型为电气化程度较高机型，其起飞和降落阶段主要由飞行员控制，巡航阶段由中央电脑控制。

航线 8：全程由飞行员控制，全航程下滑道偏差较为明显。

- **飞行参量五因素：**航线 5：飞行过程中杆量、盘量和 G 值变化较小，此次飞行较为平缓，该飞机起飞后即向右大幅度转弯而后在巡航阶段向右和向左各做出微调，在降落过程，飞机出现了向左大角度偏转，我们认为机长进行短五边进近或遭遇了恶劣天气，其姿态总体保持恒定。

航线 6：其巡航阶段与航线 5 一致，起飞阶段坡度发生了较大改变，且与起飞姿态变化相重合，我们认为其在起飞阶段受到天气状况影响，起飞过程较为摇晃。将进近过程中坡度变化量与航线 5 相比较，我们认为其进近方向与航线 5 相悖。

航线 7 与航线 8：我们认为这两条航线为两座机场互通航线，其相关数据对应相反。飞行过程中，飞行员在巡航阶段多次修正航线。降落过程较为平缓，姿态变化量较小。速度。

航线 5, 6, 7, 8 计算空速均与地速呈正比，与风速呈反比。

- **海拔及无线电高度：**航线 5, 6, 7, 8 的无线电高度基本保持不变，海拔高度反映全航段的飞行过程。

5.5.2 附件 1 中 8 次航班综合分析

为得到附件 1 中所有航班的一致性特点，我们首先将经过预处理后的表格（这里不包括标签编码及数据标准化或归一化处理）进行全汇总，将 8 次航班信息导入至一张表格，分析在问题一中的得出的结果中的与飞行安全相关性大的所有指标，观察数据分布情况。但在此之前，我们发现大多指标概率频率不为 1 Hz，即 1 秒内记录多个值，因此我们这里选择对应指标其在该时刻（1 秒内）出现的最大值为对应指标，需要这样处理的指标有：“G 值”“姿态”“杆量”“坡度”“盘量”“左油门杆位置”“右油门杆位置”。经过上述处理后，我们对附件指标进行筛选，选择出我们需要的指标，进行综合分析，所选指标为：“具体时间”“计算空速”“G 值”“姿态”“杆量”“坡度”“盘量”“下滑道偏差 (C)”“下滑道偏差 (L)”“下滑道偏差 (R)”“航向道偏差 (C)”“航向道偏差 (L)”“航向道偏差 (R)”“俯仰角率”“下降率”“RUDD 位置”“左油门杆位置”“右油门杆位置”。

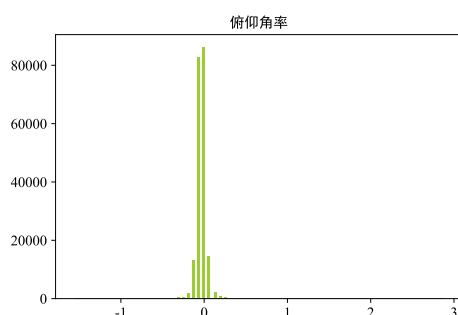


图 29 俯仰角率数据分布直方图

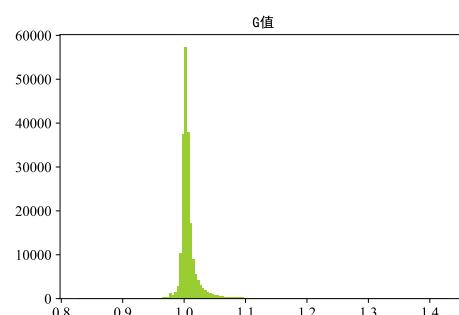


图 30 G 值数据分布直方图

这里我们以“俯仰角率”及“G 值”为例，绘制数据分布直方图，如图 29与图 30所示。

从图示上我们可以直观发现，其均服从正态分布。因此我们通过对飞行 QAR 数据的大样本统计分析：大部分飞行性能参数在较大的样本空间内都近似呈现正态分布^[16]。对于其余指标的检验，由于指标过多，篇幅过大，我们不将在正文中展现，而将其放置于附录中，读者可自行查阅，见图 35 ~ 图 49。因此这里，我们采用 3σ 方法对各指标进行时刻识别，分析异常值。在这里，我们也解释了，为何在问题一中不对异常值进行剔除，因为我们认为，异常值的存在还很大可能来源于此刻飞机出现某些危险性状态、环境干预加强、认为操作失误等，若将其剔除，有一定可能将会影响我们对飞行安全的判断。

对于上述指标，我们还需要对其进行分类，部分指标需要计算其时间变化率，从变化率才可较为准确地反映此刻飞行的参量，具体划分如表 9所示。

表 9 参量划分及处理方法

处理方法	指标名称
变化率处理	姿态、坡度、RUDD 位置、左油门杆位置、右油门杆位置
正常取值	G 值、杆量、盘量、下滑道偏差 (C)、下滑道偏差 (L)、下滑道偏差 (R)、航向道偏差 (C)、航向道偏差 (L)、航向道偏差 (R)、俯仰角率、下降率

经过上述处理后，我们即可开始计算各参数的均值、标准差，再以 3σ 方法，设定参量超限阈值，最终阈值见表 10。

表 10 十六项指标阈值设定

阈值	姿态	坡度	RUDD 位置	左油门杆位置
最小值	-0.1824	-0.6674	-0.0669	-0.7398
最大值	0.1824	0.6674	0.0669	0.7398
阈值	右油门杆位置	G 值	杆量	盘量
最小值	0.7365	0.9537	-0.0455	-0.0617
最大值	0.7365	1.0636	0.0441	0.0550
阈值	下滑道偏差 (C)	下滑道偏差 (L)	下滑道偏差 (R)	航向道偏差 (C)
最小值	-3.2725	-3.0885	-3.0608	-4.9660
最大值	3.3470	3.1475	3.1096	4.7401
阈值	航向道偏差 (L)	航向道偏差 (R)	俯仰角率	下降率
最小值	-4.9525	-4.9612	-0.2695	-1660.3728
最大值	4.7258	4.7562	0.2102	1660.7590

5.5.3 附件 1 中 8 次航班仿真

由于附件 1 中存在 8 次航班数据，数据体量大，手动分析较为困难，且在实际应用场景中，每天全球各地航班统计困难，同时我们也难以有足够的力量去逐一分析。因此，在这里，我们利用 Python 中 def 字段定义函数，建立自动化智能预警机制系统，将各指标阈值输入，传入已处理完成的数据，即可完成实时自动化预警，旨在提高分析效率，预防可能的安全事故。我们采用 1 Hz 记录频率，逐秒进行实时监测，最终可以得到关于“飞行姿态变化”“坡度变化”“RUDD 位置变化”“左油门杆位置变化”“右油门杆位置变化”“失重超重情况”“杆量情况”“盘量情况”“下滑道偏差 (C) 情况”“下滑道偏差 (L) 情况”“下滑道偏差 (R) 情

况”“航向道偏差（C）情况”“航向道偏差（L）情况”“航向道偏差（R）情况”“俯仰角率情况”“下降率情况”的情况。同对每秒实行实时监测，即可快速对异常进行分析，尽可能地预防安全事故的发生，对于航线 8 的仿真结果，我们随机选取 10 行展示如下，如表 11 所示。⁸而对于该航班其余时刻以及其余航班所有时刻仿真结果，我们将会放于“计算结果”及“支撑材料”中，供读者查阅。

表 11 航班 8 仿真结果，随机截取部分结果

具体时间	飞行姿态变化	坡度变化	RUDD 位置变化	左油门杆位置变化
12:00:59	正常	正常	正常	正常
12:01:05	正常	正常	正常	正常
12:08:53	异常	正常	正常	正常
12:08:54	异常	异常	正常	正常
12:08:56	异常	正常	正常	正常
12:17:50	正常	异常	正常	正常
12:17:51	异常	异常	正常	正常
13:08:31	正常	正常	正常	正常
13:48:41	正常	异常	正常	异常
13:48:42	正常	正常	正常	正常
具体时间	下滑道偏差（C）情况	下滑道偏差（L）情况	下滑道偏差（R）情况	航向道偏差（C）情况
12:00:59	正常	正常	正常	正常
12:01:05	正常	正常	异常 R	正常
12:08:53	正常	异常 L	正常	正常
12:08:54	正常	正常	正常	正常
12:08:56	正常	正常	正常	正常
12:17:50	正常	正常	正常	正常
12:17:51	正常	正常	正常	正常
13:08:31	正常	正常	正常	正常
13:48:41	异常 C	异常 L	异常 R	正常
13:48:42	正常	异常 L	异常 R	正常
具体时间	右油门杆位置变化	失重超重情况	杆量情况	盘量情况
12:00:59	正常	正常	正常	正常
12:01:05	正常	正常	正常	正常
12:08:53	正常	超重	松杆（推杆）过度	右旋过度
12:08:54	正常	超重	松杆（推杆）过度	正常
12:08:56	正常	超重	松杆（推杆）过度	正常
12:17:50	正常	正常	正常	左旋过度
12:17:51	正常	正常	正常	左旋过度
13:08:31	正常	正常	正常	正常
13:48:41	异常	超重	正常	左旋过度
13:48:42	正常	超重	正常	正常
具体时间	航向道偏差（L）情况	航向道偏差（R）情况	俯仰角率情况	下降率情况
12:00:59	正常	正常	正常	正常
12:01:05	正常	正常	正常	正常
12:08:53	正常	正常	异常	正常
12:08:54	正常	正常	异常	正常
12:08:56	正常	正常	异常	正常
12:17:50	正常	正常	正常	正常
12:17:51	正常	正常	正常	正常
13:08:31	正常	正常	正常	正常
13:48:41	正常	正常	正常	正常
13:48:42	正常	正常	正常	正常

⁸由于列指标过多，因此我们省略中间计算指标，而仅显示时刻与各指标是否异常情况。

六、模型的评价与推广

6.1 模型的评价

- **模型的优点:**

1. 由于 QAR 数据维度较高，我们通过 PCA 累计解释方差模型进行降维，确定了具有代表性指标的个数，结果更加符合现实。
2. 综合随机森林模型及 XGBoost 模型，对指标重要度进行排序，并选择合理性质变，定性且定量分析。
3. 利用熵权法计算飞行参数对飞行安全的权重，保证评价的客观性。
4. 在建立模型前我们对 QAR 数据进行预处理，使数据更真实完整，得到的结果更符合实际。
5. 对 QAR 数据以时间序列为特点，进行数据合理性分析，保证后续分析的数据来源可靠。
6. 在对超限的分析中我们从不同的方面分析，使得分析的结果更加全面。
7. 建立自动化智能预警机制系统，我们根据数据特点，合理设定阈值，尽可能地保证预测结果的准确，预防可能的安全事故。

- **模型的缺点及改进:**

1. 层次聚类模型在计算过程中需要计算每个样本间的距离，计算复杂度较高。在未来，我们可以选用计算复杂度低的模型来对样本进行分类。
2. 用 PCA 确定影响飞行员飞行技术的指标时，主成分各个特征维度的含义具有一定的模糊性，其特征的解释性没有原始样本强。
3. 仅通过设定阈值判定超限，仍存在一定局限性。在未来，可以构建风险判断矩阵，引入贝叶斯网络，对飞行数据进行更深层次分析，提升对未知事件预测的准确性。

6.2 模型的推广

此模型有利于处理高维复杂数据，因此除了监控飞行安全外，我们也可将此模型运用于我国高铁铁路运行监测中，虽然火车作为一种较为安全的出行方式，但并不代表其不会发生事故，此模型可以协助铁路公司分析列车的运行情况，对列车运行做出监测，及时就相关可能威胁到列车运行安全的操作做出预警。

以此类推，除了交通方面，此模型适用于需要实时监测运行状态的企业，如发电厂、高危化工企业等。在一定程度上完善了当前的预警体系，为减少相关行业恶性事件的发生做出了贡献。

参考文献

- [1] 刘柳. 基于 QAR 数据的着陆阶段飞行风险研究 [D]. 重庆大学,2018.
- [2] 龙海江. 基于 QAR 数据的重着陆分析研究 [D]. 中国民用航空飞行学院,2020.DOI:10.27722/d.cnki.gzgmc.2020.000089.
- [3] QAR 数据为什么不能简单的清洗和修正? [EB/OL].<http://news.carnoc.com/list/593/593309.html>.
- [4] 使用 QAR 实现进近着陆指标评估设计思路浅析.[EB/OL].<http://news.carnoc.com/list/593/593265.html>.
- [5] CSDN. 【数据预处理】sklearn 实现数据预处理（归一化、标准化）[EB/OL].
https://blog.csdn.net/weixin_44109827/article/details/124786873.
- [6] 姚文字, 李杰, 李岩峰, 高娜, 王涛. 基于熵权法的呼吸机质量综合评价研究 [C]//. 中国医学装备大会暨 2022 医学装备展览会论文汇编 (下册) .[出版者不详],2022:162-167.DOI:10.26914/c.cnkihy.2022.042155.
- [7] 谢赤, 钟赞. 熵权法在银行经营绩效综合评价中的应用 [J]. 中国软科学,2002(09):109-111+108.
- [8] 刘建新, 史志仙. 概率论与数理统计 [M]. 北京: 高等教育出版社,2016:115.
- [9] 司守奎, 孙玺菁. 数学建模算法与应用 [M]. 北京: 国防工业出版社,2022:264.
- [10] 饶雷, 冉军, 陶建权, 胡号朋, 吴沁, 熊圣新. 基于随机森林的海上风电机组发电机轴承异常状态监测方法 [J]. 船舶工程,2022,44(S2):27-31.DOI:10.13788/j.cnki.cbgc.2022.S2.06.
- [11] 陈振宇, 刘金波, 李晨, 季晓慧, 李大鹏, 黄运豪, 狄方春, 高兴宇, 徐立中. 基于 LSTM 与 XGBoost 组合模型的超短期电力负荷预测 [J]. 电网技术,2020,44(02):614-620.DOI:10.13335/j.1000-3673.pst.2019.1566.
- [12] 杨贵军, 徐雪, 赵富强. 基于 XGBoost 算法的用户评分预测模型及应用 [J]. 数据分析与知识发现,2019,3(01):118-126.
- [13] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). Association for Computing Machinery, New York, NY, USA, 785-794. <https://doi.org/10.1145/2939672.2939785>.
- [14] A.Tharwat, Applied Computing and Informatics (2018). <https://doi.org/10.1016/j.aci.2018.08.003>.
- [15] 郑薇. 基于 QAR 数据的重着陆风险评估及预测研究 [D]. 中国民航大学,2014.
- [16] 汪磊, 孙瑞山, 吴昌旭, 崔振新, 陆正. 基于飞行 QAR 数据的重着陆风险定量评价模型 [J]. 中国安全科学学报,2014,24(02):88-92.DOI:10.16265/j.cnki.issn1003-3033.2014.02.016.

附录

[A] 图示

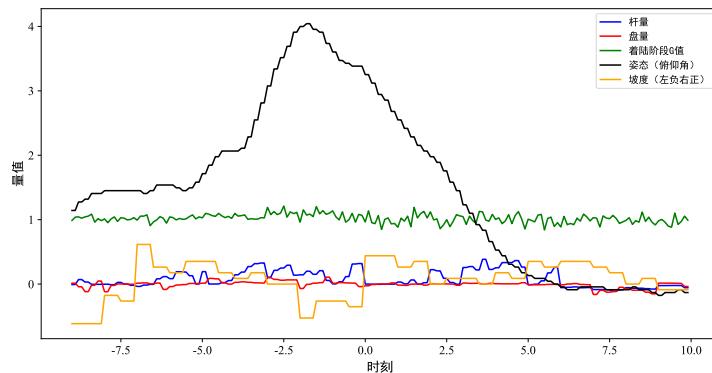


图 31 航班 2 着陆过程指标量值实时监视

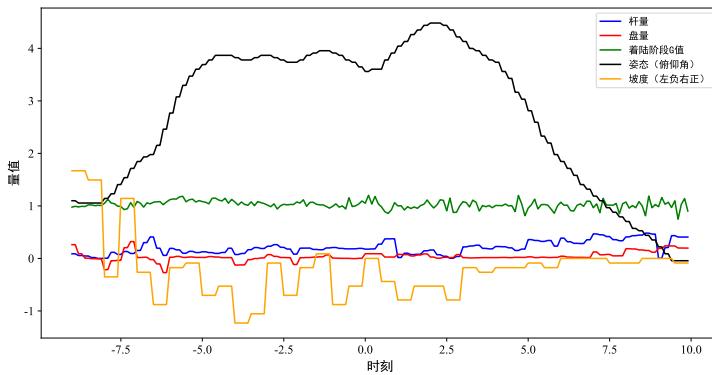


图 32 航班 4 着陆过程指标量值实时监视

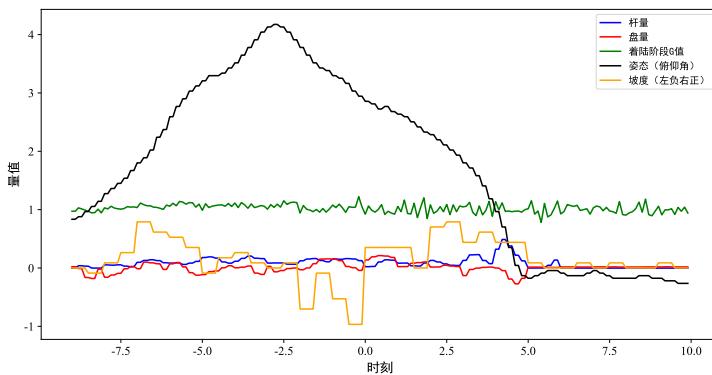


图 33 航班 5 着陆过程指标量值实时监视

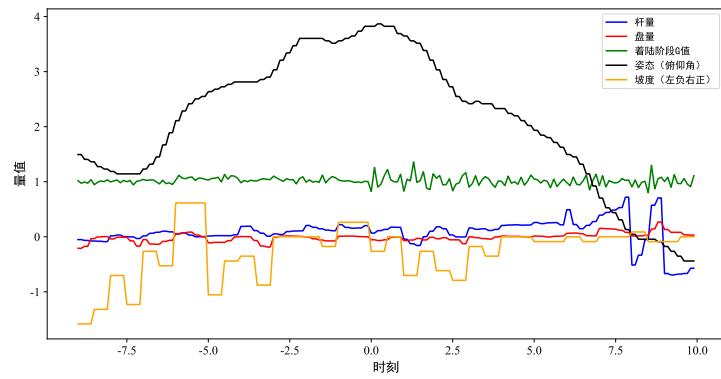


图 34 航班 6 着陆过程指标量值实时监视

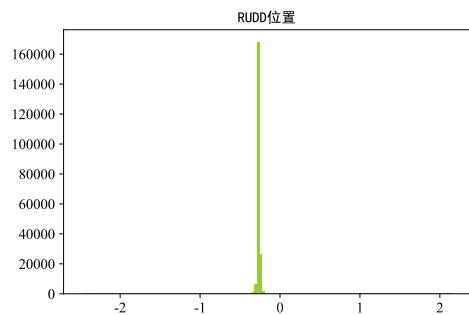


图 35 RUDD 位置数据分布直方图

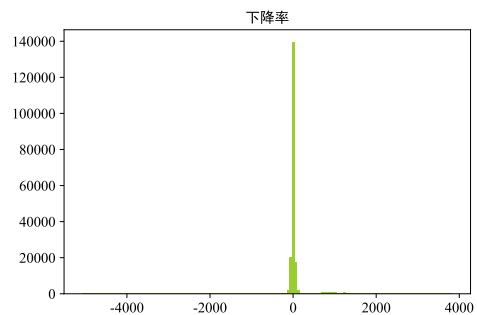


图 36 下降率数据分布直方图

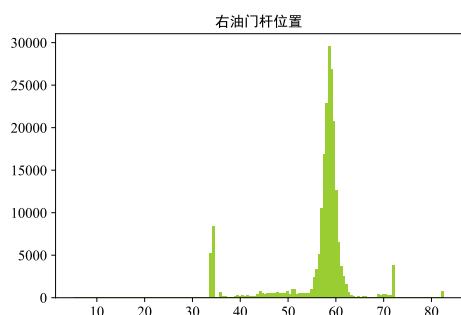


图 37 右油门杆位置数据分布直方图

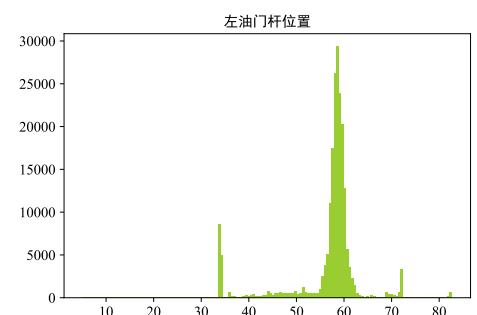


图 38 左油门杆位置数据分布直方图

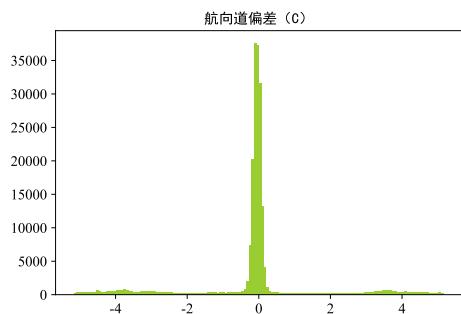


图 39 航向道偏差 (C) 数据分布直方图

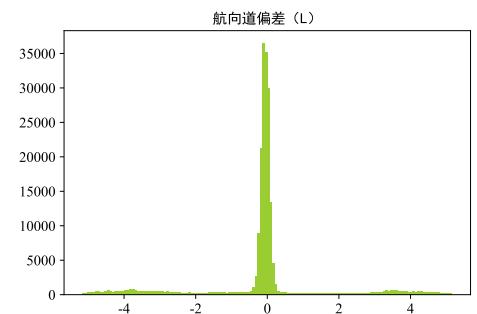


图 40 航向道偏差 (L) 数据分布直方图

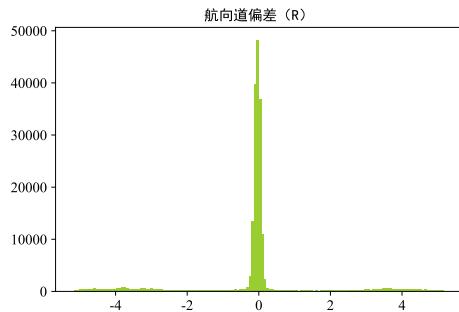


图 41 航向道偏差 (R) 数据分布直方图

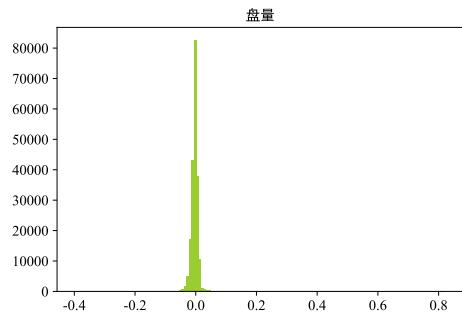


图 42 盘量数据分布直方图

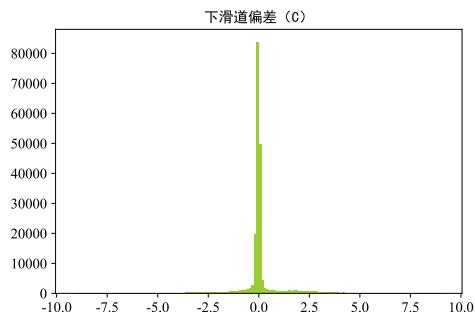


图 43 下滑道偏差 (C) 数据分布直方图

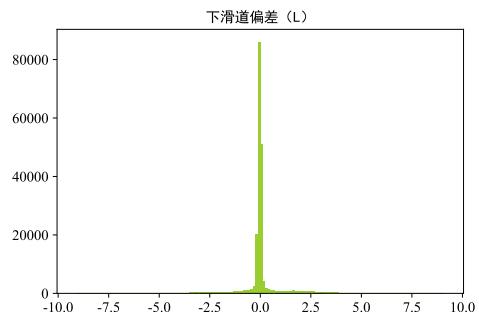


图 44 下滑道偏差 (L) 数据分布直方图

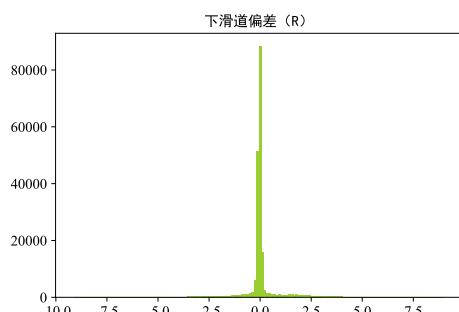


图 45 下滑道偏差 (R) 数据分布直方图

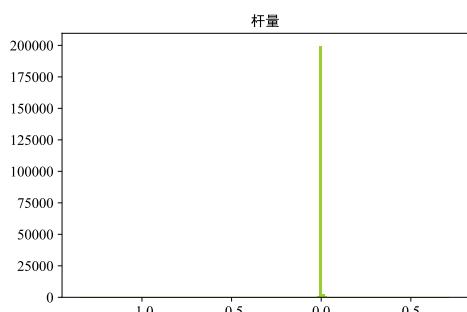


图 46 杆量数据分布直方图

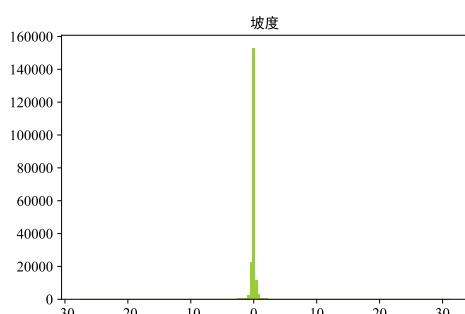


图 47 坡度数据分布直方图

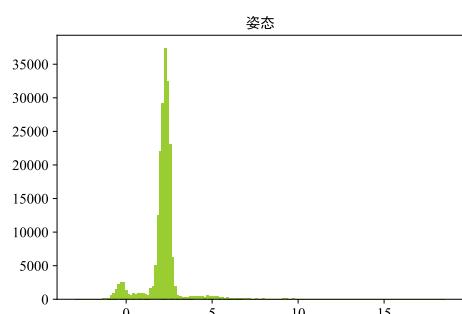


图 48 姿态数据分布直方图

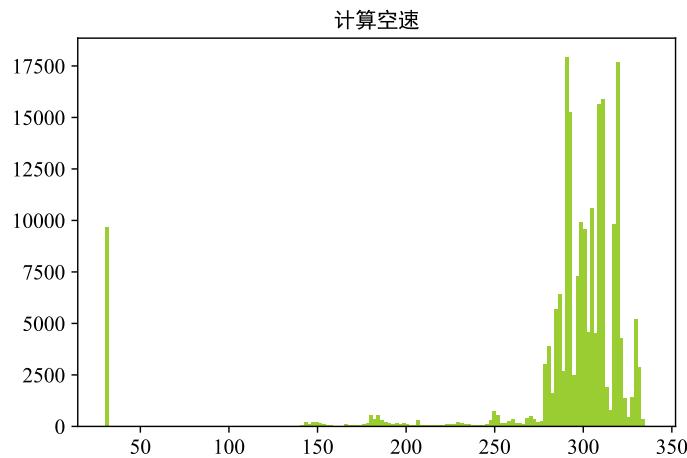


图 49 计算空速数据分布直方图

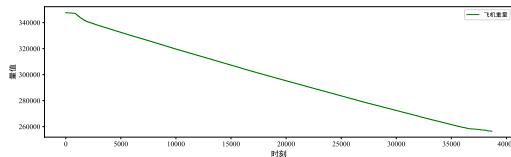


图 50 第 1 次航班飞机重量

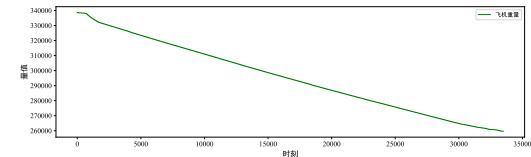


图 51 第 2 次航班飞机重量

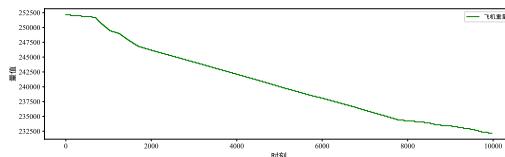


图 52 第 3 次航班飞机重量

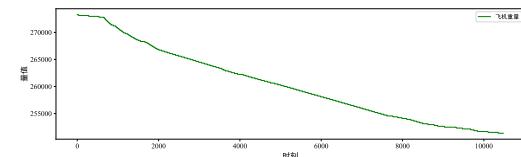


图 53 第 4 次航班飞机重量

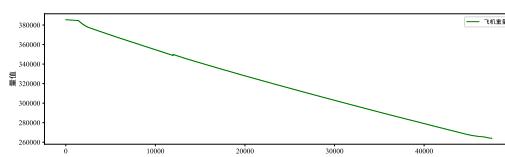


图 54 第 5 次航班飞机重量

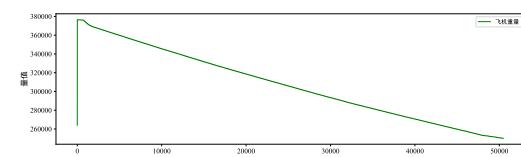


图 55 第 6 次航班飞机重量

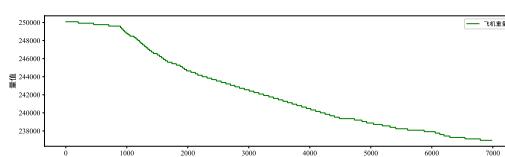


图 56 第 7 次航班飞机重量

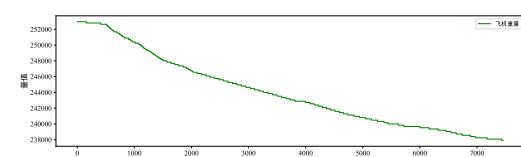


图 57 第 8 次航班飞机重量



图 58 XGBoost 重要度排序

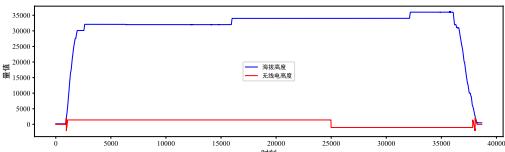


图 59 第 1 次航班海拔高度与无线电高度

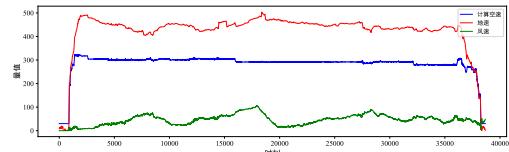


图 60 第 1 次航班空速、地速、风速

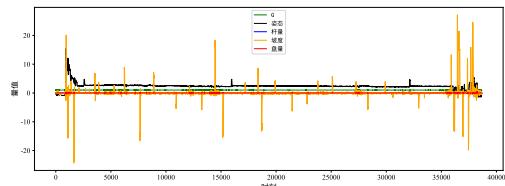


图 61 第 1 次航班飞行参量

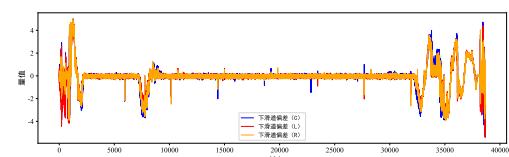


图 62 第 1 次航班下滑道偏差

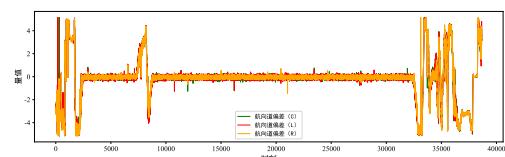


图 63 第 1 次航班航向道偏差

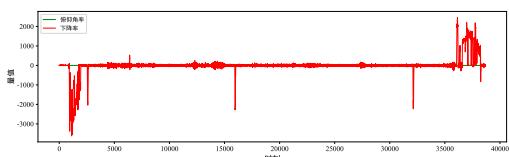


图 64 第 1 次航班俯仰角率、下降率

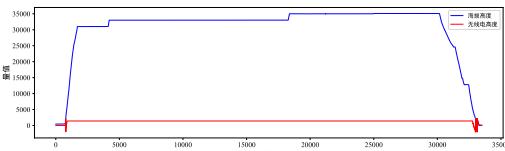


图 65 第 2 次航班海拔高度与无线电高度

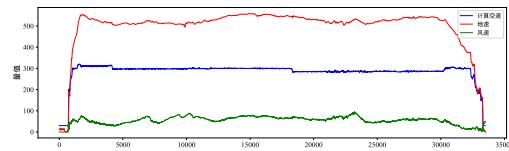


图 66 第 2 次航班空速、地速、风速

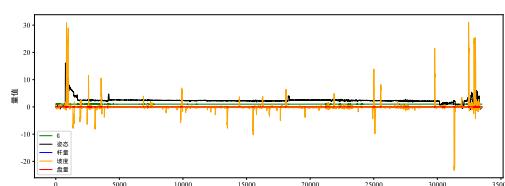


图 67 第 2 次航班飞行参量

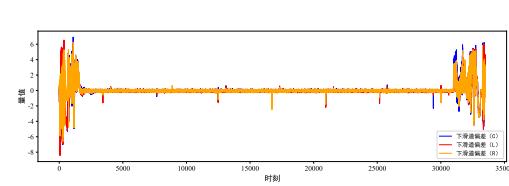


图 68 第 2 次航班下滑道偏差

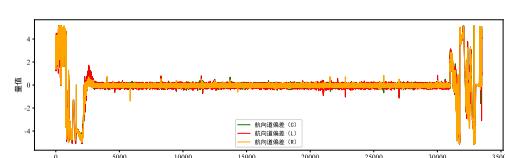


图 69 第 2 次航班航向道偏差

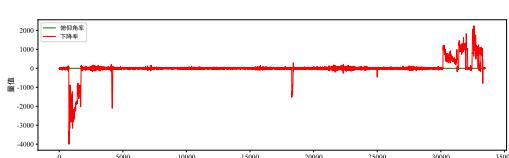


图 70 第 2 次航班俯仰角率、下降率

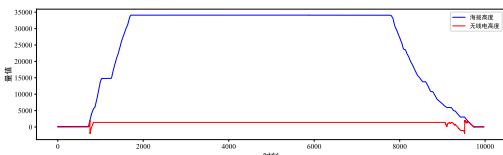


图 71 第 3 次航班海拔高度与无线电高度

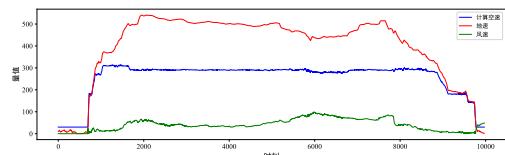


图 72 第 3 次航班空速、地速、风速

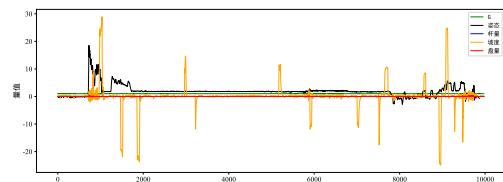


图 73 第 3 次航班飞行参量

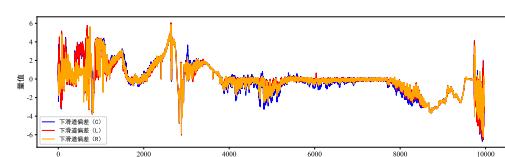


图 74 第 3 次航班下滑道偏差

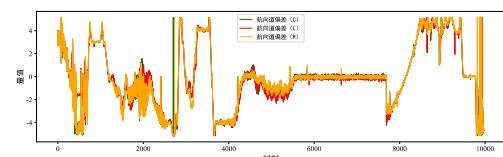


图 75 第 3 次航班航向道偏差

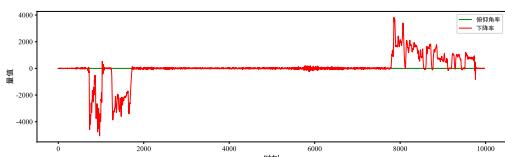


图 76 第 3 次航班俯仰角率、下降率

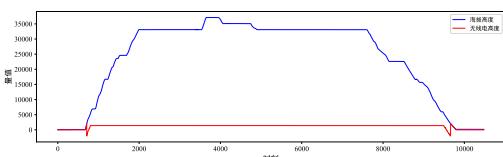


图 77 第 4 次航班海拔高度与无线电高度

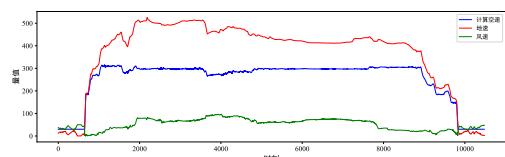


图 78 第 4 次航班空速、地速、风速

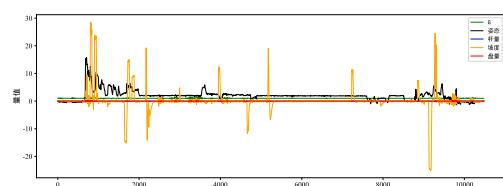


图 79 第 4 次航班飞行参量

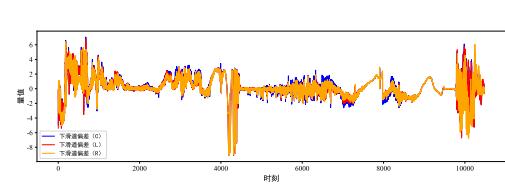


图 80 第 4 次航班下滑道偏差

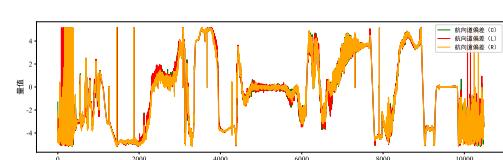


图 81 第 4 次航班航向道偏差

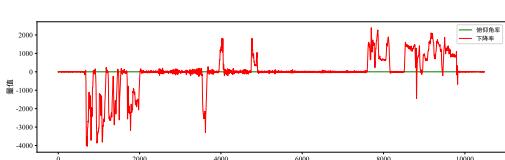


图 82 第 4 次航班俯仰角率、下降率

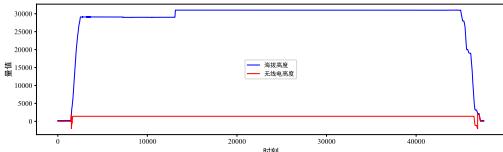


图 83 第 5 次航班海拔高度与无线电高度

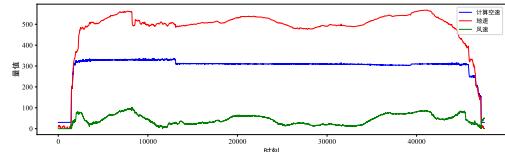


图 84 第 5 次航班空速、地速、风速

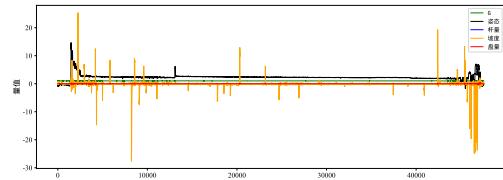


图 85 第 5 次航班飞行参量

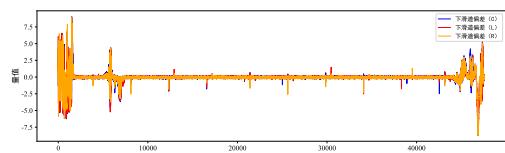


图 86 第 5 次航班下滑道偏差

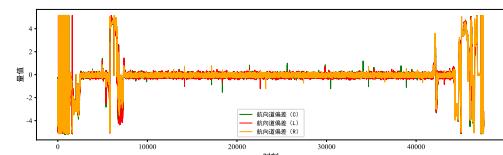


图 87 第 5 次航班航向道偏差

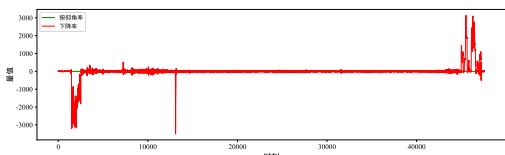


图 88 第 5 次航班俯仰角率、下降率

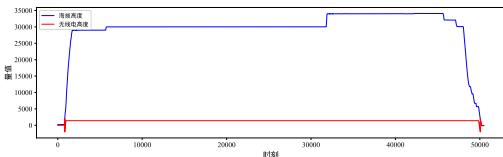


图 89 第 6 次航班海拔高度与无线电高度

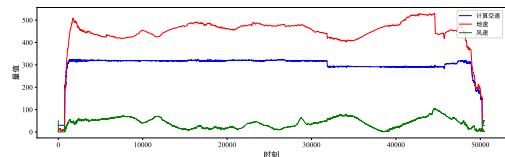


图 90 第 6 次航班空速、地速、风速

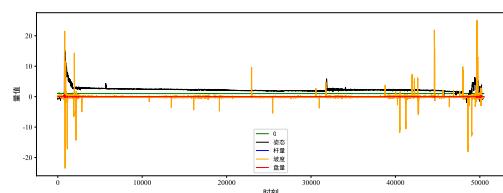


图 91 第 6 次航班飞行参量

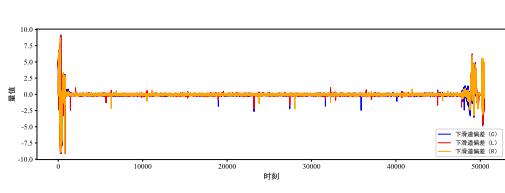


图 92 第 6 次航班下滑道偏差

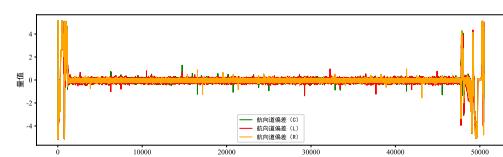


图 93 第 6 次航班航向道偏差

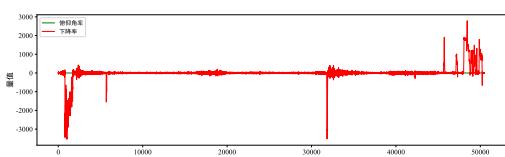


图 94 第 6 次航班俯仰角率、下降率

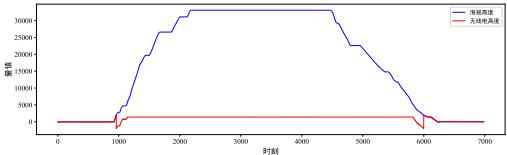


图 95 第 7 次航班海拔高度与无线电高度

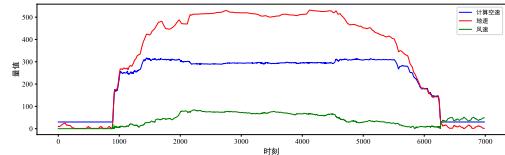


图 96 第 7 次航班空速、地速、风速

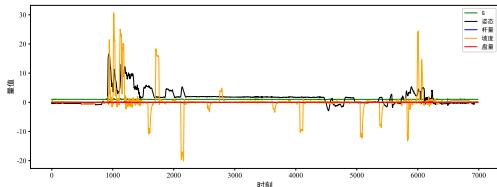


图 97 第 7 次航班飞行参量

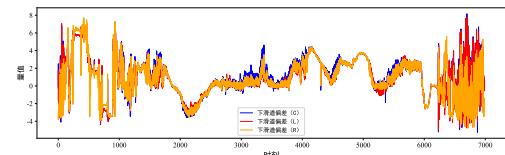


图 98 第 7 次航班下滑道偏差

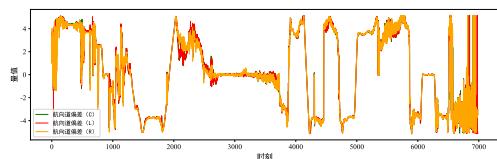


图 99 第 7 次航班航向道偏差

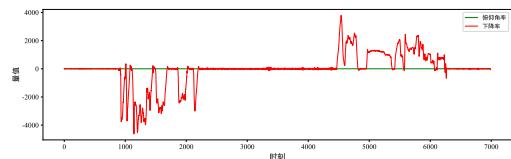


图 100 第 7 次航班俯仰角率、下降率

[B] 支撑文件列表

支撑文件列表如下（列表中不包含原始数据集以及中途产生的临时数据文件）：

文件夹名	描述
html 文件	包括所有解决问题的源程序运行结果
ipynb 文件	包括所有解决问题的源程序源代码
py 文件	包括所有解决问题的源程序输出 python 文件
仿真结果	包括附件 1 的 8 次航班全时刻的飞行状态预警

[C] 使用的软件、环境

C.1: 为解决该问题，我们所使用的主要软件有：

- TeX Live 2022
- Visual Studio Code 1.77.3
- WPS Office 2023 春季更新 (14036)
- Python 3.10.4
- Pycharm 2023.1 (Professional Edition)

C.2: Python 环境下所用使用到的库及其版本如下：

库	版本	库	版本
copy	内置库	matplotlib	3.5.2
jupyter	1.0.0	numpy	1.22.4+mkl
jupyter-client	7.3.1	openpyxl	3.0.10
jupyter-console	6.4.3	pandas	1.4.2
jupyter-contrib-core	0.4.0	pyecharts	1.9.1
jupyter-contrib-nbextensions	0.5.1	scikit-learn	0.22.2 psot1
jupyter-highlight-selected-word	0.2.0	sklearn	0.0
jupyterlab-pygments	0.2.2	snapshot_phantomjs	0.0.3
jupyterlab-widgets	1.1.0	xgboost	1.6.1
jupyter-latex-envs	1.4.6	yellowbrick	1.4
jupyter-nbextensions-configurator	0.5.0		

[D] 问题解决源程序

D.1 航班 1 数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # # 附件一预处理（不包括异常值剔除及标准化）
5
6 # In[1]:
7
8
9 import pandas as pd
10 import numpy as np
11
12
13 # In[2]:
14
15
16 data=pd.read_excel("Data\\First\\New\\201404070532.xlsx",sheet_name='201404070532')
17 data
18
19
20 # In[3]:
21
22
23 data.info()
24
25
26 # In[4]:
27
28
29 import missingno
30 import matplotlib.pyplot as plt
31 plt.rcParams['font.sans-serif']=['SimHei']
32 plt.rcParams['axes.unicode_minus']=False
33 missingno.bar(data, color=(190/255,190/255,190/255))
34 plt.tight_layout()
35
36
37 # In[5]:
38
39
40 data.replace({"起落架":{'DOWN':1},
41                 "空地电门0.2秒":{True:1,False:0},
42                 "空地电门0.4秒":{True:1,False:0},
```

```

43     "空地电门0.6秒":{True:1,False:0},
44     "空地电门0.8秒":{True:1,False:0},
45     "空地电门1秒":{True:1,False:0},
46     "是否接通了A/T":{'DISENGD':0,'ENGAGED':1},
47     "是否接通了任意侧的A/P":{'OFF':0,'ON':1},
48 }, inplace=True)
49 data
50
51
52 # In[6]:
53
54
55 data=data.fillna(0)
56 data
57
58
59 # In[7]:
60
61
62 data.drop(labels=[‘月’,‘日’,‘起飞机场’,‘落地机场’,‘飞机重量’],axis=1,inplace=True)
63 data
64
65
66 # # QAR异常判断，剔除
67
68 # In[8]:
69
70
71 dup_row = data.duplicated(subset=[‘具体时间’], keep=False)
72 data.insert(0, ‘is_dup’, dup_row)
73 data[data[‘is_dup’] == True]
74
75
76 # In[9]:
77
78
79 data=data.drop_duplicates(subset=[‘具体时间’],keep='first')
80 data
81
82
83 # In[10]:
84
85
86 dup_row = data.duplicated(subset=[‘具体时间’], keep=False)

```

```

87 data.insert(0, 'is_dup_N', dup_row)
88 data[data['is_dup_N'] == True]
89
90
91 # In[11]:
92
93
94 def function(a, b):
95     if a == b:
96         return 1
97     else:
98         return 0
99
100
101 data['bool'] = data.apply(lambda x : function(x['俯仰角率'],x['俯仰角率.1']),axis = 1)
102 data
103
104
105 # In[12]:
106
107
108 data[data['bool']==0]
109
110
111 # In[13]:
112
113
114 data=data.drop(labels=['is_dup','is_dup_N','bool','具体时间','俯仰角率.1'],axis=1)
115 data


---


1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data=pd.read_csv('Data\\First\\NNew\\1-1D.csv')

```

```
15 data
16
17
18 # # 数据标准化
19
20 # In[3]:
21
22
23 import sklearn.preprocessing as sp
24 StandardTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
25 StandardTransform
26
27
28 # In[4]:
29
30
31 ListColumns=list(StandardTransform.columns)
32 ListColumns
33
34
35 # In[5]:
36
37
38 StandardTransformScaler = sp.StandardScaler()
39 StandardTransformScaler = StandardTransformScaler.fit(StandardTransform)
40 StandardTransform = StandardTransformScaler.transform(StandardTransform)
41 StandardTransform = pd.DataFrame(StandardTransform)
42 StandardTransform.columns = ListColumns
43 StandardTransform
44
45
46 # # 数据归一化
47
48 # In[6]:
49
50
51 import sklearn.preprocessing as sp
52 MinMaxTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
53 MinMaxTransform
54
55
56 # In[7]:
57
58
```

```

59 MinMaxTransformScaler=spMinMaxScaler(feature_range=(0.002, 1))
60 MinMaxTransformScaler=MinMaxTransformScaler.fit(MinMaxTransform)
61 MinMaxTransform=MinMaxTransformScaler.transform(MinMaxTransform)
62 MinMaxTransform=pd.DataFrame(MinMaxTransform)
63 MinMaxTransform.columns=ListColumns
64 MinMaxTransform
65
66
67 # # PCA&层次聚类
68
69 # In[8]:
70
71
72 import matplotlib.pyplot as plt
73
74
75 # In[9]:
76
77
78 from sklearn.decomposition import PCA
79
80 pca = PCA()
81 pca.fit(StandardTransform)
82 evr = pca.explained_variance_ratio_
83
84 plt.figure(figsize=(12, 5))
85 plt.plot(range(0, len(evr)), evr.cumsum(), marker="d", linestyle="--")
86 plt.xlabel("Number of components",font='Times New Roman',fontsize=13)
87 plt.ylabel("Cumulative explained variance",font='Times New Roman',fontsize=13)
88 plt.xticks(font='Times New Roman',fontsize=12)
89 plt.yticks(font='Times New Roman',fontsize=12)
90 plt.tight_layout()
91 plt.savefig("Figures\\1-1 PCA Cumulative explained variance.pdf")
92
93
94 # In[10]:
95
96
97 import scipy.cluster.hierarchy as sch
98 plt.figure(figsize=(12, 6))
99 dendrogram = sch.dendrogram(sch.linkage(StandardTransform.T, method = 'ward'))
100 plt.rcParams['font.sans-serif']=['SimHei']
101 plt.rcParams['axes.unicode_minus']=False
102 plt.xlabel('指标',fontsize=13)

```

```

103 plt.ylabel('离差平方和', fontsize=13)
104 plt.xticks(font='Times New Roman', fontsize=11)
105 plt.yticks(font='Times New Roman', fontsize=11)
106 plt.savefig("Figures\\1-1 层次聚类树状图.pdf")
107
108
109 # 环境: 风向、风速
110 # 人为操作: 下降率、地速、起落架、G值、杆量、盘量、RUDD位置
111 # 飞机状态: 姿态(俯仰角)、坡度、发动机N1值、磁航向、油门杆位置、下滑道偏差、航向道偏差、
112     俯仰角率
113 # In[11]:
114
115
116 IFactor=MinMaxTransform.loc[:,['风向','风速','下降率','地速','起落架',
117     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','
118     '着陆G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒',
119     '着陆G值0.9秒','着陆G值1秒','姿态(俯仰角)','姿态(俯仰角).1',
120     '姿态(俯仰角).2','姿态(俯仰角).3','姿态(俯仰角).4','杆量',
121     '杆量.1','杆量.2','杆量.3','杆量.4','坡度(左负右正)',
122     '坡度(左负右正).1','盘量','盘量.1','盘量.2','盘量.3',
123     '盘量.4','左侧发动机油门N1值','右侧发动机油门N1值','磁航向',
124     '左发油门杆位置(角度)','左发油门杆位置(角度).1','右发油门杆位置',
125     '右发油门杆位置(角度)','右发油门杆位置(角度).1','RUDD位置',
126     '下滑道偏差(C)','下滑道偏差(L)','下滑道偏差(R)','航向道偏差(C)',
127     '航向道偏差(L)','航向道偏差(R)','俯仰角率']]
128 IFactor
129
130
131 IFactorEnv=IFactor.loc[:,['风向','风速']]
132 IFactorEnv
133
134
135 IFactorPeo=IFactor.loc[:,['下降率','地速','起落架',
136     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','
137     '着陆G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','
138     '着陆G值0.9秒','着陆G值1秒'],

```

```

133             ,杆量', '杆量.1', '杆量.2', '杆量.3', '杆量.4',
134             ,盘量', '盘量.1', '盘量.2', '盘量.3', '盘量.4',
135             ,RUDD位置', ])
136 IFactorPeo
137
138
139 # In[14]:
140
141
142 IFactorMac=IFactor.loc[:,~IFactor.columns.isin(['风向', '风速', '下降率', '地速', '起落架',
143                         ,着陆G值0.1秒', '着陆G值0.2秒', '着陆G值0.3秒', '着陆G值0.4秒', '着陆
144                         G值0.5秒', '着陆G值0.6秒', '着陆G值0.7秒', '着陆G值0.8秒', '着陆G
145                         值0.9秒', '着陆G值1秒',
146                         ,杆量', '杆量.1', '杆量.2', '杆量.3', '杆量.4',
147                         ,盘量', '盘量.1', '盘量.2', '盘量.3', '盘量.4', 'RUDD位置', ])]
148 IFactorMac
149
150
151
152 import copy
153 def ewm(data):
154     label_need = data.keys()[:]
155     data1 = data[label_need].values
156     data2 = data1
157     [m, n] = data2.shape
158     data3 = copy.deepcopy(data2)
159     p = copy.deepcopy(data3)
160     for j in range(0, n):
161         p[:, j] = data3[:, j] / sum(data3[:, j])
162     e = copy.deepcopy(data3[0, :])
163     for j in range(0, n):
164         e[j] = -1 / np.log(m) * sum(p[:, j] * np.log(p[:, j]))
165     w = (1 - e) / sum(1 - e)
166     total = 0
167     for sum_w in range(0, len(w)):
168         total = total + w[sum_w]
169     print(f'权重为: {w}, 权重之和为: {total}')
170
171
172 # In[16]:
173
174

```

```
175  ewm(IFactorEnv)
176
177
178 # In[17]:
179
180
181 ewm(IFactorPeo)
182
183
184 # In[18]:
185
186
187 ewm(IFactorMac)
188
189
190 # In[19]:
191
192
193 ListEnv=list(IFactorEnv.columns)
194 ListPeo=list(IFactorPeo.columns)
195 ListMac=list(IFactorMac.columns)
196 print(ListEnv)
197 print(ListPeo)
198 print(ListMac)
```

D.2 航班 2 数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # # 附件一预处理（不包括异常值剔除及标准化）
5
6 # In[1]:
7
8
9 import pandas as pd
10 import numpy as np
11
12
13 # In[2]:
14
15
16 data=pd.read_excel("Data\\First\\New\\201404071917.xlsx",sheet_name='201404071917')
17 data
18
19
20 # In[3]:
21
22
23 data.info()
24
25
26 # In[4]:
27
28
29 import missingno
30 import matplotlib.pyplot as plt
31 plt.rcParams['font.sans-serif']=['SimHei']
32 plt.rcParams['axes.unicode_minus']=False
33 missingno.bar(data, color=(190/255,190/255,190/255))
34 plt.tight_layout()
35
36
37 # In[5]:
38
39
40 data.replace({"起落架":{'DOWN':1},
41                 "空地电门0.2秒":{True:1,False:0},
42                 "空地电门0.4秒":{True:1,False:0},
43                 "空地电门0.6秒":{True:1,False:0},
```

```

44     "空地电门0.8秒":{True:1,False:0},
45     "空地电门1秒":{True:1,False:0},
46     "是否接通了A/T":{'DISENGD':0,'ENGAGED':1},
47     "是否接通了任意侧的A/P":{'OFF':0,'ON':1},
48 }, inplace=True)
49 data
50
51
52 # In[6]:
53
54
55 data=data.fillna(0)
56 data
57
58
59 # In[7]:
60
61
62 data.drop(labels=['月','日','起飞机场','落地机场','飞机重量'],axis=1,inplace=True)
63 data
64
65
66 # # QAR异常判断，剔除
67
68 # In[8]:
69
70
71 dup_row = data.duplicated(subset=['具体时间'], keep=False)
72 data.insert(0, 'is_dup', dup_row)
73 data[data['is_dup'] == True]
74
75
76 # In[9]:
77
78
79 data=data.drop_duplicates(subset=['具体时间'],keep='first')
80 data
81
82
83 # In[10]:
84
85
86 dup_row = data.duplicated(subset=['具体时间'], keep=False)
87 data.insert(0, 'is_dup_N', dup_row)

```

```

88 data[data['is_dup_N'] == True]
89
90
91 # In[11]:
92
93
94 def function(a, b):
95     if a == b:
96         return 1
97     else:
98         return 0
99
100
101 data['bool'] = data.apply(lambda x : function(x['俯仰角率'],x['俯仰角率.1']),axis = 1)
102 data
103
104
105 # In[12]:
106
107
108 data[data['bool']==0]
109
110
111 # In[13]:
112
113
114 data=data.drop(labels=['is_dup','is_dup_N','bool','具体时间','俯仰角率.1'],axis=1)
115 data


---


1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data=pd.read_csv('Data\\First\\NNew\\1-2D.csv')
15 data

```

```
16
17
18 # # 数据标准化
19
20 # In[3]:
21
22
23 import sklearn.preprocessing as sp
24 StandardTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
25 StandardTransform
26
27
28 # In[4]:
29
30
31 ListColumns=list(StandardTransform.columns)
32 ListColumns
33
34
35 # In[5]:
36
37
38 StandardTransformScaler = sp.StandardScaler()
39 StandardTransformScaler = StandardTransformScaler.fit(StandardTransform)
40 StandardTransform = StandardTransformScaler.transform(StandardTransform)
41 StandardTransform = pd.DataFrame(StandardTransform)
42 StandardTransform.columns = ListColumns
43 StandardTransform
44
45
46 # # 数据归一化
47
48 # In[6]:
49
50
51 import sklearn.preprocessing as sp
52 MinMaxTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
53 MinMaxTransform
54
55
56 # In[7]:
57
58
59 MinMaxTransformScaler=sp.MinMaxScaler(feature_range=(0.002, 1))
```

```

60 MinMaxTransformScaler=MinMaxTransformScaler.fit(MinMaxTransform)
61 MinMaxTransform=MinMaxTransformScaler.transform(MinMaxTransform)
62 MinMaxTransform=pd.DataFrame(MinMaxTransform)
63 MinMaxTransform.columns=ListColumns
64 MinMaxTransform
65
66
67 # # PCA&层次聚类
68
69 # In[8]:
70
71
72 import matplotlib.pyplot as plt
73
74
75 # In[9]:
76
77
78 from sklearn.decomposition import PCA
79
80 pca = PCA()
81 pca.fit(StandardTransform)
82 evr = pca.explained_variance_ratio_
83
84 plt.figure(figsize=(12, 5))
85 plt.plot(range(0, len(evr)), evr.cumsum(), marker="d", linestyle="--")
86 plt.xlabel("Number of components",font='Times New Roman',fontsize=13)
87 plt.ylabel("Cumulative explained variance",font='Times New Roman',fontsize=13)
88 plt.xticks(font='Times New Roman',fontsize=12)
89 plt.yticks(font='Times New Roman',fontsize=12)
90 plt.tight_layout()
91 plt.savefig("Figures\\1-2 PCA Cumulative explained variance.pdf")
92
93
94 # In[10]:
95
96
97 import scipy.cluster.hierarchy as sch
98 plt.figure(figsize=(12, 6))
99 dendrogram = sch.dendrogram(sch.linkage(StandardTransform.T, method = 'ward'))
100 plt.rcParams['font.sans-serif']=['SimHei']
101 plt.rcParams['axes.unicode_minus']=False
102 plt.xlabel('指标',fontsize=13)
103 plt.ylabel('离差平方和',fontsize=13)

```

```

104 plt.xticks(font='Times New Roman', fontsize=11)
105 plt.yticks(font='Times New Roman', fontsize=11)
106 plt.savefig("Figures\\1-2 层次聚类树状图.pdf")
107
108
109 # 环境: 风向、风速
110 # 人为操作: 下降率、地速、起落架、G值、杆量、盘量、RUDD位置
111 # 飞机状态: 姿态(俯仰角)、坡度、发动机N1值、磁航向、油门杆位置、下滑道偏差、航向道偏差、
112     俯仰角率
113
114
115
116 IFactor=MinMaxTransform.loc[:,['风向','风速','下降率','地速','起落架',
117     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒',
118     '着陆G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒',
119     '着陆G值0.9秒','着陆G值1秒',
120     '姿态(俯仰角)','姿态(俯仰角).1','姿态(俯仰角).2','姿态
121     (俯仰角).3','姿态(俯仰角).4',
122     '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
123     '坡度(左负右正)','坡度(左负右正).1',
124     '盘量','盘量.1','盘量.2','盘量.3','盘量.4',
125     '左侧发动机油门N1值','右侧发动机油门N1值','磁航向',
126     '左发油门杆位置(角度)','左发油门杆位置(角度).1','右发油门
127     杆位置(角度)','右发油门杆位置(角度).1',
128     'RUDD位置','下滑道偏差(C)','下滑道偏差(L)','下滑道偏差(R
129     )',
130     '航向道偏差(C)','航向道偏差(L)','航向道偏差(R),
131     俯仰角率']]]

132
133 IFactorEnv=IFactor.loc[:,['风向','风速']]
134 IFactorEnv
135
136
137 # In[13]:
138
139
140 IFactorPeo=IFactor.loc[:,['下降率','地速','起落架',
141     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆

```

```

G值0.5秒，'，着陆G值0.6秒，'，着陆G值0.7秒，'，着陆G
142 值0.8秒，'，着陆G
143 值0.9秒，'，着陆G值1秒，'
144 '杆量，'，杆量.1'，'杆量.2'，'杆量.3'，'杆量.4'，
145 '盘量，'，盘量.1'，'盘量.2'，'盘量.3'，'盘量.4'，'RUDD位置，]]]

146 IFactorPeo
147
148 # In[14]:
149
150
151 IFactorMac=IFactor.loc[:,~IFactor.columns.isin(['风向','风速','下降率','地速','起落架',
152 '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆
153 G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','着陆G
154 值0.9秒','着陆G值1秒'],
155 '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
156 '盘量','盘量.1','盘量.2','盘量.3','盘量.4','RUDD位置',)])]
157 IFactorMac
158
159
160
161 import copy
162 def ewm(data):
163     label_need = data.keys()[:]
164     data1 = data[label_need].values
165     data2 = data1
166     [m, n] = data2.shape
167     data3 = copy.deepcopy(data2)
168     p = copy.deepcopy(data3)
169     for j in range(0, n):
170         p[:, j] = data3[:, j] / sum(data3[:, j])
171     e = copy.deepcopy(data3[0, :])
172     for j in range(0, n):
173         e[j] = -1 / np.log(m) * sum(p[:, j] * np.log(p[:, j]))
174     w = (1 - e) / sum(1 - e)
175     total = 0
176     for sum_w in range(0, len(w)):
177         total = total + w[sum_w]
178     print(f'权重为: {w}, 权重之和为: {total}')
179
180
181 # In[16]:

```

```
182  
183  
184 ewm(IFactorEnv)  
185  
186  
187 # In[17]:  
188  
189  
190 ewm(IFactorPeo)  
191  
192  
193 # In[18]:  
194  
195  
196 ewm(IFactorMac)
```

D.3 航班 3 数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # # 附件一预处理（不包括异常值剔除及标准化）
5
6 # In[1]:
7
8
9 import pandas as pd
10 import numpy as np
11
12
13 # In[2]:
14
15
16 data=pd.read_excel("Data\\First\\New\\201404080617.xlsx",sheet_name='201404080617')
17 data
18
19
20 # In[3]:
21
22
23 data.info()
24
25
26 # In[4]:
27
28
29 import missingno
30 import matplotlib.pyplot as plt
31 plt.rcParams['font.sans-serif']=['SimHei']
32 plt.rcParams['axes.unicode_minus']=False
33 missingno.bar(data, color=(190/255,190/255,190/255))
34 plt.tight_layout()
35
36
37 # In[5]:
38
39
40 data.replace({"起落架":{'DOWN':1},
41                 "空地电门0.2秒":{True:1,False:0},
42                 "空地电门0.4秒":{True:1,False:0},
43                 "空地电门0.6秒":{True:1,False:0},
```

```

44     "空地电门0.8秒":{True:1,False:0},
45     "空地电门1秒":{True:1,False:0},
46     "是否接通了A/T":{'DISENGD':0,'ENGAGED':1},
47     "是否接通了任意侧的A/P":{'OFF':0,'ON':1},
48 }, inplace=True)
49 data
50
51
52 # In[6]:
53
54
55 data=data.fillna(0)
56 data
57
58
59 # In[7]:
60
61
62 data.drop(labels=['月','日','起飞机场','落地机场','飞机重量'],axis=1,inplace=True)
63 data
64
65
66 # # QAR异常判断，剔除
67
68 # In[8]:
69
70
71 dup_row = data.duplicated(subset=['具体时间'], keep=False)
72 data.insert(0, 'is_dup', dup_row)
73 data[data['is_dup'] == True]
74
75
76 # In[9]:
77
78
79 data=data.drop_duplicates(subset=['具体时间'],keep='first')
80 data
81
82
83 # In[10]:
84
85
86 dup_row = data.duplicated(subset=['具体时间'], keep=False)
87 data.insert(0, 'is_dup_N', dup_row)

```

```

88 data[data['is_dup_N'] == True]
89
90
91 # In[11]:
92
93
94 def function(a, b):
95     if a == b:
96         return 1
97     else:
98         return 0
99
100
101 data['bool'] = data.apply(lambda x : function(x['俯仰角率'],x['俯仰角率.1']),axis = 1)
102 data
103
104
105 # In[12]:
106
107
108 data[data['bool']==0]
109
110
111 # In[13]:
112
113
114 data=data.drop(labels=['is_dup','is_dup_N','bool','具体时间','俯仰角率.1'],axis=1)
115 data


---


1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data=pd.read_csv('Data\\First\\NNew\\1-3D.csv')
15 data

```

```
16
17
18 # # 数据标准化
19
20 # In[3]:
21
22
23 import sklearn.preprocessing as sp
24 StandardTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
25 StandardTransform
26
27
28 # In[4]:
29
30
31 ListColumns=list(StandardTransform.columns)
32 ListColumns
33
34
35 # In[5]:
36
37
38 StandardTransformScaler = sp.StandardScaler()
39 StandardTransformScaler = StandardTransformScaler.fit(StandardTransform)
40 StandardTransform = StandardTransformScaler.transform(StandardTransform)
41 StandardTransform = pd.DataFrame(StandardTransform)
42 StandardTransform.columns = ListColumns
43 StandardTransform
44
45
46 # # 数据归一化
47
48 # In[6]:
49
50
51 import sklearn.preprocessing as sp
52 MinMaxTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
53 MinMaxTransform
54
55
56 # In[7]:
57
58
59 MinMaxTransformScaler=sp.MinMaxScaler(feature_range=(0.002, 1))
```

```

60 MinMaxTransformScaler=MinMaxTransformScaler.fit(MinMaxTransform)
61 MinMaxTransform=MinMaxTransformScaler.transform(MinMaxTransform)
62 MinMaxTransform=pd.DataFrame(MinMaxTransform)
63 MinMaxTransform.columns=ListColumns
64 MinMaxTransform
65
66
67 # # PCA&层次聚类
68
69 # In[8]:
70
71
72 import matplotlib.pyplot as plt
73
74
75 # In[9]:
76
77
78 from sklearn.decomposition import PCA
79
80 pca = PCA()
81 pca.fit(StandardTransform)
82 evr = pca.explained_variance_ratio_
83
84 plt.figure(figsize=(12, 5))
85 plt.plot(range(0, len(evr)), evr.cumsum(), marker="d", linestyle="--")
86 plt.xlabel("Number of components",font='Times New Roman',fontsize=13)
87 plt.ylabel("Cumulative explained variance",font='Times New Roman',fontsize=13)
88 plt.xticks(font='Times New Roman',fontsize=12)
89 plt.yticks(font='Times New Roman',fontsize=12)
90 plt.tight_layout()
91 plt.savefig("Figures\\1-3 PCA Cumulative explained variance.pdf")
92
93
94 # In[10]:
95
96
97 import scipy.cluster.hierarchy as sch
98 plt.figure(figsize=(12, 6))
99 dendrogram = sch.dendrogram(sch.linkage(StandardTransform.T, method = 'ward'))
100 plt.rcParams['font.sans-serif']=['SimHei']
101 plt.rcParams['axes.unicode_minus']=False
102 plt.xlabel('指标',fontsize=13)
103 plt.ylabel('离差平方和',fontsize=13)

```

```

104 plt.xticks(font='Times New Roman', fontsize=11)
105 plt.yticks(font='Times New Roman', fontsize=11)
106 plt.savefig("Figures\\1-3 层次聚类树状图.pdf")
107
108
109 # 环境: 风向、风速
110 # 人为操作: 下降率、地速、起落架、G值、杆量、盘量、RUDD位置
111 # 飞机状态: 姿态(俯仰角)、坡度、发动机N1值、磁航向、油门杆位置、下滑道偏差、航向道偏差、
112     俯仰角率
113
114
115
116 IFactor=MinMaxTransform.loc[:,['风向','风速','下降率','地速','起落架',
117     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒',
118     '着陆G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒',
119     '着陆G值0.9秒','着陆G值1秒',
120     '姿态(俯仰角)','姿态(俯仰角).1','姿态(俯仰角).2','姿态
121     (俯仰角).3','姿态(俯仰角).4',
122     '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
123     '坡度(左负右正)','坡度(左负右正).1',
124     '盘量','盘量.1','盘量.2','盘量.3','盘量.4',
125     '左侧发动机油门N1值','右侧发动机油门N1值','磁航向',
126     '左发油门杆位置(角度)','左发油门杆位置(角度).1','右发油门
127     杆位置(角度)','右发油门杆位置(角度).1',
128     'RUDD位置','下滑道偏差(C)','下滑道偏差(L)','下滑道偏差(R
129     )',
130     '航向道偏差(C)','航向道偏差(L)','航向道偏差(R),
131     俯仰角率']]]

132
133 IFactorEnv=IFactor.loc[:,['风向','风速']]
134 IFactorEnv
135
136
137 # In[13]:
138
139
140 IFactorPeo=IFactor.loc[:,['下降率','地速','起落架',
141     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆

```

```

G值0.5秒，'，着陆G值0.6秒，'，着陆G值0.7秒，'，着陆G
142 值0.8秒，'，着陆G
143 值0.9秒，'，着陆G值1秒，'
144 '杆量，'，杆量.1'，'杆量.2'，'杆量.3'，'杆量.4'，
145 '盘量，'，盘量.1'，'盘量.2'，'盘量.3'，'盘量.4'，'
146 'RUDD位置，]]]
147
148 # In[14]:
149
150
151 IFactorMac=IFactor.loc[:,~IFactor.columns.isin(['风向','风速','下降率','地速','起落架',
152 '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆
153 G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','着陆G
154 值0.9秒','着陆G值1秒'],
155 '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
156 '盘量','盘量.1','盘量.2','盘量.3','盘量.4','RUDD位置',)])]
157
158 IFactorMac
159
160
161 # In[15]:
162
163 import copy
164 def ewm(data):
165     label_need = data.keys()[:]
166     data1 = data[label_need].values
167     data2 = data1
168     [m, n] = data2.shape
169     data3 = copy.deepcopy(data2)
170     p = copy.deepcopy(data3)
171     for j in range(0, n):
172         p[:, j] = data3[:, j] / sum(data3[:, j])
173     e = copy.deepcopy(data3[0, :])
174     for j in range(0, n):
175         e[j] = -1 / np.log(m) * sum(p[:, j] * np.log(p[:, j]))
176     w = (1 - e) / sum(1 - e)
177     total = 0
178     for sum_w in range(0, len(w)):
179         total = total + w[sum_w]
180     print(f'权重为: {w}, 权重之和为: {total}')
181 # In[16]:

```

```
182  
183  
184 ewm(IFactorEnv)  
185  
186  
187 # In[17]:  
188  
189  
190 ewm(IFactorPeo)  
191  
192  
193 # In[18]:  
194  
195  
196 ewm(IFactorMac)
```

D.4 航班 4 数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # # 附件一预处理（不包括异常值剔除及标准化）
5
6 # In[1]:
7
8
9 import pandas as pd
10 import numpy as np
11
12
13 # In[2]:
14
15
16 data=pd.read_excel("Data\\First\\New\\201404081034.xlsx",sheet_name='201404081034')
17 data
18
19
20 # In[3]:
21
22
23 data.info()
24
25
26 # In[4]:
27
28
29 import missingno
30 import matplotlib.pyplot as plt
31 plt.rcParams['font.sans-serif']=['SimHei']
32 plt.rcParams['axes.unicode_minus']=False
33 missingno.bar(data, color=(190/255,190/255,190/255))
34 plt.tight_layout()
35
36
37 # In[5]:
38
39
40 data.replace({"起落架":{'DOWN':1},
41                 "空地电门0.2秒":{True:1,False:0},
42                 "空地电门0.4秒":{True:1,False:0},
43                 "空地电门0.6秒":{True:1,False:0},
```

```
44     "空地电门0.8秒":{True:1,False:0},
45     "空地电门1秒":{True:1,False:0},
46     "是否接通了A/T":{'DISENGD':0,'ENGAGED':1},
47     "是否接通了任意侧的A/P":{'OFF':0,'ON':1},
48 }, inplace=True)
49 data
50
51
52 # In[6]:
53
54
55 data=data.fillna(0)
56 data
57
58
59 # In[7]:
60
61
62 data.drop(labels=['月','日','起飞机场','落地机场','飞机重量'],axis=1,inplace=True)
63 data
64
65
66 # # QAR异常判断，剔除
67
68 # In[8]:
69
70
71 dup_row = data.duplicated(subset=['具体时间'], keep=False)
72 data.insert(0, 'is_dup', dup_row)
73 data[data['is_dup'] == True]
74
75
76 # In[9]:
77
78
79 data=data.drop_duplicates(subset=['具体时间'],keep='first')
80 data
81
82
83 # In[10]:
84
85
86 dup_row = data.duplicated(subset=['具体时间'], keep=False)
87 data.insert(0, 'is_dup_N', dup_row)
```

```

88 data[data['is_dup_N'] == True]
89
90
91 # In[11]:
92
93
94 def function(a, b):
95     if a == b:
96         return 1
97     else:
98         return 0
99
100
101 data['bool'] = data.apply(lambda x : function(x['俯仰角率'],x['俯仰角率.1']),axis = 1)
102 data
103
104
105 # In[12]:
106
107
108 data[data['bool']==0]
109
110
111 # In[13]:
112
113
114 data=data.drop(labels=['is_dup','is_dup_N','bool','具体时间','俯仰角率.1'],axis=1)
115 data


---


1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data=pd.read_csv('Data\\First\\NNew\\1-4D.csv')
15 data

```

```
16
17
18 # # 数据标准化
19
20 # In[3]:
21
22
23 import sklearn.preprocessing as sp
24 StandardTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
25 StandardTransform
26
27
28 # In[4]:
29
30
31 ListColumns=list(StandardTransform.columns)
32 ListColumns
33
34
35 # In[5]:
36
37
38 StandardTransformScaler = sp.StandardScaler()
39 StandardTransformScaler = StandardTransformScaler.fit(StandardTransform)
40 StandardTransform = StandardTransformScaler.transform(StandardTransform)
41 StandardTransform = pd.DataFrame(StandardTransform)
42 StandardTransform.columns = ListColumns
43 StandardTransform
44
45
46 # # 数据归一化
47
48 # In[6]:
49
50
51 import sklearn.preprocessing as sp
52 MinMaxTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
53 MinMaxTransform
54
55
56 # In[7]:
57
58
59 MinMaxTransformScaler=sp.MinMaxScaler(feature_range=(0.002, 1))
```

```

60 MinMaxTransformScaler=MinMaxTransformScaler.fit(MinMaxTransform)
61 MinMaxTransform=MinMaxTransformScaler.transform(MinMaxTransform)
62 MinMaxTransform=pd.DataFrame(MinMaxTransform)
63 MinMaxTransform.columns=ListColumns
64 MinMaxTransform
65
66
67 # # PCA&层次聚类
68
69 # In[8]:
70
71
72 import matplotlib.pyplot as plt
73
74
75 # In[9]:
76
77
78 from sklearn.decomposition import PCA
79
80 pca = PCA()
81 pca.fit(StandardTransform)
82 evr = pca.explained_variance_ratio_
83
84 plt.figure(figsize=(12, 5))
85 plt.plot(range(0, len(evr)), evr.cumsum(), marker="d", linestyle="--")
86 plt.xlabel("Number of components",font='Times New Roman',fontsize=13)
87 plt.ylabel("Cumulative explained variance",font='Times New Roman',fontsize=13)
88 plt.xticks(font='Times New Roman',fontsize=12)
89 plt.yticks(font='Times New Roman',fontsize=12)
90 plt.tight_layout()
91 plt.savefig("Figures\\1-4 PCA Cumulative explained variance.pdf")
92
93
94 # In[10]:
95
96
97 import scipy.cluster.hierarchy as sch
98 plt.figure(figsize=(12, 6))
99 dendrogram = sch.dendrogram(sch.linkage(StandardTransform.T, method = 'ward'))
100 plt.rcParams['font.sans-serif']=['SimHei']
101 plt.rcParams['axes.unicode_minus']=False
102 plt.xlabel('指标',fontsize=13)
103 plt.ylabel('离差平方和',fontsize=13)

```

```

104 plt.xticks(font='Times New Roman', fontsize=11)
105 plt.yticks(font='Times New Roman', fontsize=11)
106 plt.savefig("Figures\\1-4 层次聚类树状图.pdf")
107
108
109 # 环境: 风向、风速
110 # 人为操作: 下降率、地速、起落架、G值、杆量、盘量、RUDD位置
111 # 飞机状态: 姿态(俯仰角)、坡度、发动机N1值、磁航向、油门杆位置、下滑道偏差、航向道偏差、
112     俯仰角率
113
114
115
116 IFactor=MinMaxTransform.loc[:,['风向','风速','下降率','地速','起落架',
117     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒',
118     '着陆G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒',
119     '着陆G值0.9秒','着陆G值1秒',
120     '姿态(俯仰角)','姿态(俯仰角).1','姿态(俯仰角).2','姿态
121     (俯仰角).3','姿态(俯仰角).4',
122     '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
123     '坡度(左负右正)','坡度(左负右正).1',
124     '盘量','盘量.1','盘量.2','盘量.3','盘量.4',
125     '左侧发动机油门N1值','右侧发动机油门N1值','磁航向',
126     '左发油门杆位置(角度)','左发油门杆位置(角度).1','右发油门
127     杆位置(角度)','右发油门杆位置(角度).1',
128     'RUDD位置','下滑道偏差(C)','下滑道偏差(L)','下滑道偏差(R
129     )',
130     '航向道偏差(C)','航向道偏差(L)','航向道偏差(R),
131     俯仰角率']]]

132
133 IFactorEnv=IFactor.loc[:,['风向','风速']]
134 IFactorEnv
135
136
137 # In[13]:
138
139
140 IFactorPeo=IFactor.loc[:,['下降率','地速','起落架',
141     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆

```

```

G值0.5秒，'，着陆G值0.6秒，'，着陆G值0.7秒，'，着陆G
142 值0.8秒，'，着陆G
143 值0.9秒，'，着陆G值1秒，'
144 '杆量，'，杆量.1'，'杆量.2'，'杆量.3'，'杆量.4'，
145 '盘量，'，盘量.1'，'盘量.2'，'盘量.3'，'盘量.4'，'
146 'RUDD位置，]]]
147
148 # In[14]:
149
150
151 IFactorMac=IFactor.loc[:,~IFactor.columns.isin(['风向','风速','下降率','地速','起落架',
152 '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆
153 G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','着陆G
154 值0.9秒','着陆G值1秒'],
155 '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
156 '盘量','盘量.1','盘量.2','盘量.3','盘量.4','RUDD位置',)])]
157
158 IFactorMac
159
160
161 # In[15]:
162 import copy
163 def ewm(data):
164     label_need = data.keys()[:]
165     data1 = data[label_need].values
166     data2 = data1
167     [m, n] = data2.shape
168     data3 = copy.deepcopy(data2)
169     p = copy.deepcopy(data3)
170     for j in range(0, n):
171         p[:, j] = data3[:, j] / sum(data3[:, j])
172     e = copy.deepcopy(data3[0, :])
173     for j in range(0, n):
174         e[j] = -1 / np.log(m) * sum(p[:, j] * np.log(p[:, j]))
175     w = (1 - e) / sum(1 - e)
176     total = 0
177     for sum_w in range(0, len(w)):
178         total = total + w[sum_w]
179     print(f'权重为: {w}, 权重之和为: {total}')
180
181 # In[16]:

```

```
182  
183  
184 ewm(IFactorEnv)  
185  
186  
187 # In[17]:  
188  
189  
190 ewm(IFactorPeo)  
191  
192  
193 # In[18]:  
194  
195  
196 ewm(IFactorMac)
```

D.5 航班 5 数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # # 附件一预处理（不包括异常值剔除及标准化）
5
6 # In[1]:
7
8
9 import pandas as pd
10 import numpy as np
11
12
13 # In[2]:
14
15
16 data=pd.read_excel("Data\\First\\New\\201404090110.xlsx",sheet_name='201404090110')
17 data
18
19
20 # In[3]:
21
22
23 data.info()
24
25
26 # In[4]:
27
28
29 import missingno
30 import matplotlib.pyplot as plt
31 plt.rcParams['font.sans-serif']=['SimHei']
32 plt.rcParams['axes.unicode_minus']=False
33 missingno.bar(data, color=(190/255,190/255,190/255))
34 plt.tight_layout()
35
36
37 # In[5]:
38
39
40 data.replace({"起落架":{'DOWN':1},
41                 "空地电门0.2秒":{True:1,False:0},
42                 "空地电门0.4秒":{True:1,False:0},
43                 "空地电门0.6秒":{True:1,False:0},
```

```

44     "空地电门0.8秒":{True:1,False:0},
45     "空地电门1秒":{True:1,False:0},
46     "是否接通了A/T":{'DISENGD':0,'ENGAGED':1},
47     "是否接通了任意侧的A/P":{'OFF':0,'ON':1},
48 }, inplace=True)
49 data
50
51
52 # In[6]:
53
54
55 data=data.fillna(0)
56 data
57
58
59 # In[7]:
60
61
62 data.drop(labels=['月','日','起飞机场','落地机场','飞机重量'],axis=1,inplace=True)
63 data
64
65
66 # # QAR异常判断，剔除
67
68 # In[8]:
69
70
71 dup_row = data.duplicated(subset=['具体时间'], keep=False)
72 data.insert(0, 'is_dup', dup_row)
73 data[data['is_dup'] == True]
74
75
76 # In[9]:
77
78
79 data=data.drop_duplicates(subset=['具体时间'],keep='first')
80 data
81
82
83 # In[10]:
84
85
86 dup_row = data.duplicated(subset=['具体时间'], keep=False)
87 data.insert(0, 'is_dup_N', dup_row)

```

```

88 data[data['is_dup_N'] == True]
89
90
91 # In[11]:
92
93
94 def function(a, b):
95     if a == b:
96         return 1
97     else:
98         return 0
99
100
101 data['bool'] = data.apply(lambda x : function(x['俯仰角率'],x['俯仰角率.1']),axis = 1)
102 data
103
104
105 # In[12]:
106
107
108 data[data['bool']==0]
109
110
111 # In[13]:
112
113
114 data=data.drop(labels=['is_dup','is_dup_N','bool','具体时间','俯仰角率.1'],axis=1)
115 data


---


1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data=pd.read_csv('Data\\First\\NNew\\1-5D.csv')
15 data

```

```
16
17
18 # # 数据标准化
19
20 # In[3]:
21
22
23 import sklearn.preprocessing as sp
24 StandardTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
25 StandardTransform
26
27
28 # In[4]:
29
30
31 ListColumns=list(StandardTransform.columns)
32 ListColumns
33
34
35 # In[5]:
36
37
38 StandardTransformScaler = sp.StandardScaler()
39 StandardTransformScaler = StandardTransformScaler.fit(StandardTransform)
40 StandardTransform = StandardTransformScaler.transform(StandardTransform)
41 StandardTransform = pd.DataFrame(StandardTransform)
42 StandardTransform.columns = ListColumns
43 StandardTransform
44
45
46 # # 数据归一化
47
48 # In[6]:
49
50
51 import sklearn.preprocessing as sp
52 MinMaxTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
53 MinMaxTransform
54
55
56 # In[7]:
57
58
59 MinMaxTransformScaler=sp.MinMaxScaler(feature_range=(0.002, 1))
```

```

60 MinMaxTransformScaler=MinMaxTransformScaler.fit(MinMaxTransform)
61 MinMaxTransform=MinMaxTransformScaler.transform(MinMaxTransform)
62 MinMaxTransform=pd.DataFrame(MinMaxTransform)
63 MinMaxTransform.columns=ListColumns
64 MinMaxTransform
65
66
67 # # PCA&层次聚类
68
69 # In[8]:
70
71
72 import matplotlib.pyplot as plt
73
74
75 # In[9]:
76
77
78 from sklearn.decomposition import PCA
79
80 pca = PCA()
81 pca.fit(StandardTransform)
82 evr = pca.explained_variance_ratio_
83
84 plt.figure(figsize=(12, 5))
85 plt.plot(range(0, len(evr)), evr.cumsum(), marker="d", linestyle="--")
86 plt.xlabel("Number of components",font='Times New Roman',fontsize=13)
87 plt.ylabel("Cumulative explained variance",font='Times New Roman',fontsize=13)
88 plt.xticks(font='Times New Roman',fontsize=12)
89 plt.yticks(font='Times New Roman',fontsize=12)
90 plt.tight_layout()
91 plt.savefig("Figures\\1-5 PCA Cumulative explained variance.pdf")
92
93
94 # In[10]:
95
96
97 import scipy.cluster.hierarchy as sch
98 plt.figure(figsize=(12, 6))
99 dendrogram = sch.dendrogram(sch.linkage(StandardTransform.T, method = 'ward'))
100 plt.rcParams['font.sans-serif']=['SimHei']
101 plt.rcParams['axes.unicode_minus']=False
102 plt.xlabel('指标',fontsize=13)
103 plt.ylabel('离差平方和',fontsize=13)

```

```

104 plt.xticks(font='Times New Roman', fontsize=11)
105 plt.yticks(font='Times New Roman', fontsize=11)
106 plt.savefig("Figures\\1-5 层次聚类树状图.pdf")
107
108
109 # 环境: 风向、风速
110 # 人为操作: 下降率、地速、起落架、G值、杆量、盘量、RUDD位置
111 # 飞机状态: 姿态(俯仰角)、坡度、发动机N1值、磁航向、油门杆位置、下滑道偏差、航向道偏差、
112     俯仰角率
113
114
115
116 IFactor=MinMaxTransform.loc[:,['风向','风速','下降率','地速','起落架',
117     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒',
118     '着陆G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒',
119     '着陆G值0.9秒','着陆G值1秒',
120     '姿态(俯仰角)','姿态(俯仰角).1','姿态(俯仰角).2','姿态
121     (俯仰角).3','姿态(俯仰角).4',
122     '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
123     '坡度(左负右正)','坡度(左负右正).1',
124     '盘量','盘量.1','盘量.2','盘量.3','盘量.4',
125     '左侧发动机油门N1值','右侧发动机油门N1值','磁航向',
126     '左发油门杆位置(角度)','左发油门杆位置(角度).1','右发油门
127     杆位置(角度)','右发油门杆位置(角度).1',
128     'RUDD位置','下滑道偏差(C)','下滑道偏差(L)','下滑道偏差(R
129     )',
130     '航向道偏差(C)','航向道偏差(L)','航向道偏差(R),
131     俯仰角率']]]

132
133 IFactorEnv=IFactor.loc[:,['风向','风速']]
134 IFactorEnv
135
136
137 # In[13]:
138
139
140 IFactorPeo=IFactor.loc[:,['下降率','地速','起落架',
141     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆

```

```

G值0.5秒，'，着陆G值0.6秒，'，着陆G值0.7秒，'，着陆G
142 值0.8秒，'，着陆G
143 值0.9秒，'，着陆G值1秒，'
144 '杆量，'，杆量.1'，'杆量.2'，'杆量.3'，'杆量.4'，
145 '盘量，'，盘量.1'，'盘量.2'，'盘量.3'，'盘量.4'，'
146 'RUDD位置，]]]
147
148 # In[14]:
149
150
151 IFactorMac=IFactor.loc[:,~IFactor.columns.isin(['风向','风速','下降率','地速','起落架',
152 '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆
153 G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','着陆G
154 值0.9秒','着陆G值1秒'],
155 '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
156 '盘量','盘量.1','盘量.2','盘量.3','盘量.4','RUDD位置',)])]
157
158 # In[15]:
159
160
161 import copy
162 def ewm(data):
163     label_need = data.keys()[:]
164     data1 = data[label_need].values
165     data2 = data1
166     [m, n] = data2.shape
167     data3 = copy.deepcopy(data2)
168     p = copy.deepcopy(data3)
169     for j in range(0, n):
170         p[:, j] = data3[:, j] / sum(data3[:, j])
171     e = copy.deepcopy(data3[0, :])
172     for j in range(0, n):
173         e[j] = -1 / np.log(m) * sum(p[:, j] * np.log(p[:, j]))
174     w = (1 - e) / sum(1 - e)
175     total = 0
176     for sum_w in range(0, len(w)):
177         total = total + w[sum_w]
178     print(f'权重为: {w}, 权重之和为: {total}')
179
180
181 # In[16]:

```

```
182  
183  
184 ewm(IFactorEnv)  
185  
186  
187 # In[17]:  
188  
189  
190 ewm(IFactorPeo)  
191  
192  
193 # In[18]:  
194  
195  
196 ewm(IFactorMac)
```

D.6 航班 6 数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # # 附件一预处理（不包括异常值剔除及标准化）
5
6 # In[1]:
7
8
9 import pandas as pd
10 import numpy as np
11
12
13 # In[2]:
14
15
16 data=pd.read_excel("Data\\First\\New\\201404091701.xlsx",sheet_name='201404091701')
17 data
18
19
20 # In[3]:
21
22
23 data.info()
24
25
26 # In[4]:
27
28
29 import missingno
30 import matplotlib.pyplot as plt
31 plt.rcParams['font.sans-serif']=['SimHei']
32 plt.rcParams['axes.unicode_minus']=False
33 missingno.bar(data, color=(190/255,190/255,190/255))
34 plt.tight_layout()
35
36
37 # In[5]:
38
39
40 data.replace({"起落架":{'DOWN':1},
41                 "空地电门0.2秒":{True:1,False:0},
42                 "空地电门0.4秒":{True:1,False:0},
43                 "空地电门0.6秒":{True:1,False:0},
```

```

44     "空地电门0.8秒":{True:1,False:0},
45     "空地电门1秒":{True:1,False:0},
46     "是否接通了A/T":{'DISENGD':0,'ENGAGED':1},
47     "是否接通了任意侧的A/P":{'OFF':0,'ON':1},
48 }, inplace=True)
49 data
50
51
52 # In[6]:
53
54
55 data=data.fillna(0)
56 data
57
58
59 # In[7]:
60
61
62 data.drop(labels=['月','日','起飞机场','落地机场','飞机重量'],axis=1,inplace=True)
63 data
64
65
66 # # QAR异常判断，剔除
67
68 # In[8]:
69
70
71 dup_row = data.duplicated(subset=['具体时间'], keep=False)
72 data.insert(0, 'is_dup', dup_row)
73 data[data['is_dup'] == True]
74
75
76 # In[9]:
77
78
79 data=data.drop_duplicates(subset=['具体时间'],keep='first')
80 data
81
82
83 # In[10]:
84
85
86 dup_row = data.duplicated(subset=['具体时间'], keep=False)
87 data.insert(0, 'is_dup_N', dup_row)

```

```

88 data[data['is_dup_N'] == True]
89
90
91 # In[11]:
92
93
94 def function(a, b):
95     if a == b:
96         return 1
97     else:
98         return 0
99
100
101 data['bool'] = data.apply(lambda x : function(x['俯仰角率'],x['俯仰角率.1']),axis = 1)
102 data
103
104
105 # In[12]:
106
107
108 data[data['bool']==0]
109
110
111 # In[13]:
112
113
114 data=data.drop(labels=['is_dup','is_dup_N','bool','具体时间','俯仰角率.1'],axis=1)
115 data


---


1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data=pd.read_csv('Data\\First\\NNew\\1-6D.csv')
15 data

```

```
16
17
18 # # 数据标准化
19
20 # In[3]:
21
22
23 import sklearn.preprocessing as sp
24 StandardTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
25 StandardTransform
26
27
28 # In[4]:
29
30
31 ListColumns=list(StandardTransform.columns)
32 ListColumns
33
34
35 # In[5]:
36
37
38 StandardTransformScaler = sp.StandardScaler()
39 StandardTransformScaler = StandardTransformScaler.fit(StandardTransform)
40 StandardTransform = StandardTransformScaler.transform(StandardTransform)
41 StandardTransform = pd.DataFrame(StandardTransform)
42 StandardTransform.columns = ListColumns
43 StandardTransform
44
45
46 # # 数据归一化
47
48 # In[6]:
49
50
51 import sklearn.preprocessing as sp
52 MinMaxTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
53 MinMaxTransform
54
55
56 # In[7]:
57
58
59 MinMaxTransformScaler=sp.MinMaxScaler(feature_range=(0.002, 1))
```

```

60 MinMaxTransformScaler=MinMaxTransformScaler.fit(MinMaxTransform)
61 MinMaxTransform=MinMaxTransformScaler.transform(MinMaxTransform)
62 MinMaxTransform=pd.DataFrame(MinMaxTransform)
63 MinMaxTransform.columns=ListColumns
64 MinMaxTransform
65
66
67 # # PCA&层次聚类
68
69 # In[8]:
70
71
72 import matplotlib.pyplot as plt
73
74
75 # In[9]:
76
77
78 from sklearn.decomposition import PCA
79
80 pca = PCA()
81 pca.fit(StandardTransform)
82 evr = pca.explained_variance_ratio_
83
84 plt.figure(figsize=(12, 5))
85 plt.plot(range(0, len(evr)), evr.cumsum(), marker="d", linestyle="--")
86 plt.xlabel("Number of components",font='Times New Roman',fontsize=13)
87 plt.ylabel("Cumulative explained variance",font='Times New Roman',fontsize=13)
88 plt.xticks(font='Times New Roman',fontsize=12)
89 plt.yticks(font='Times New Roman',fontsize=12)
90 plt.tight_layout()
91 plt.savefig("Figures\\1-6 PCA Cumulative explained variance.pdf")
92
93
94 # In[10]:
95
96
97 import scipy.cluster.hierarchy as sch
98 plt.figure(figsize=(12, 6))
99 dendrogram = sch.dendrogram(sch.linkage(StandardTransform.T, method = 'ward'))
100 plt.rcParams['font.sans-serif']=['SimHei']
101 plt.rcParams['axes.unicode_minus']=False
102 plt.xlabel('指标',fontsize=13)
103 plt.ylabel('离差平方和',fontsize=13)

```

```

104 plt.xticks(font='Times New Roman', fontsize=11)
105 plt.yticks(font='Times New Roman', fontsize=11)
106 plt.savefig("Figures\\1-6 层次聚类树状图.pdf")
107
108
109 # 环境: 风向、风速
110 # 人为操作: 下降率、地速、起落架、G值、杆量、盘量、RUDD位置
111 # 飞机状态: 姿态(俯仰角)、坡度、发动机N1值、磁航向、油门杆位置、下滑道偏差、航向道偏差、
112     俯仰角率
113
114
115
116 IFactor=MinMaxTransform.loc[:,['风向','风速','下降率','地速','起落架',
117     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒',
118     '着陆G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒',
119     '着陆G值0.9秒','着陆G值1秒',
120     '姿态(俯仰角)','姿态(俯仰角).1','姿态(俯仰角).2','姿态
121     (俯仰角).3','姿态(俯仰角).4',
122     '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
123     '坡度(左负右正)','坡度(左负右正).1',
124     '盘量','盘量.1','盘量.2','盘量.3','盘量.4',
125     '左侧发动机油门N1值','右侧发动机油门N1值','磁航向',
126     '左发油门杆位置(角度)','左发油门杆位置(角度).1','右发油门
127     杆位置(角度)','右发油门杆位置(角度).1',
128     'RUDD位置','下滑道偏差(C)','下滑道偏差(L)','下滑道偏差(R
129     )',
130     '航向道偏差(C)','航向道偏差(L)','航向道偏差(R),
131     俯仰角率']]]

132
133 IFactorEnv=IFactor.loc[:,['风向','风速']]
134 IFactorEnv
135
136
137 # In[13]:
138
139
140 IFactorPeo=IFactor.loc[:,['下降率','地速','起落架',
141     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆

```

```

G值0.5秒，'，着陆G值0.6秒，'，着陆G值0.7秒，'，着陆G
142 值0.8秒，'，着陆G
143 值0.9秒，'，着陆G值1秒，'
144 '杆量，'，杆量.1'，'杆量.2'，'杆量.3'，'杆量.4'，
145 '盘量，'，盘量.1'，'盘量.2'，'盘量.3'，'盘量.4'，'
146 'RUDD位置，]]]
147
148 # In[14]:
149
150
151 IFactorMac=IFactor.loc[:,~IFactor.columns.isin(['风向','风速','下降率','地速','起落架',
152 '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆
153 G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','着陆G
154 值0.9秒','着陆G值1秒'],
155 '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
156 '盘量','盘量.1','盘量.2','盘量.3','盘量.4','RUDD位置',)])]
157
158 # In[15]:
159
160
161 import copy
162 def ewm(data):
163     label_need = data.keys()[:]
164     data1 = data[label_need].values
165     data2 = data1
166     [m, n] = data2.shape
167     data3 = copy.deepcopy(data2)
168     p = copy.deepcopy(data3)
169     for j in range(0, n):
170         p[:, j] = data3[:, j] / sum(data3[:, j])
171     e = copy.deepcopy(data3[0, :])
172     for j in range(0, n):
173         e[j] = -1 / np.log(m) * sum(p[:, j] * np.log(p[:, j]))
174     w = (1 - e) / sum(1 - e)
175     total = 0
176     for sum_w in range(0, len(w)):
177         total = total + w[sum_w]
178     print(f'权重为: {w}, 权重之和为: {total}')
179
180
181 # In[16]:

```

```
182  
183  
184 ewm(IFactorEnv)  
185  
186  
187 # In[17]:  
188  
189  
190 ewm(IFactorPeo)  
191  
192  
193 # In[18]:  
194  
195  
196 ewm(IFactorMac)
```

D.7 航班 7 数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # # 附件一预处理（不包括异常值剔除及标准化）
5
6 # In[1]:
7
8
9 import pandas as pd
10 import numpy as np
11
12
13 # In[2]:
14
15
16 data=pd.read_excel("Data\\First\\New\\201404100843.xlsx",sheet_name='201404100843')
17 data
18
19
20 # In[3]:
21
22
23 data.info()
24
25
26 # In[4]:
27
28
29 import missingno
30 import matplotlib.pyplot as plt
31 plt.rcParams['font.sans-serif']=['SimHei']
32 plt.rcParams['axes.unicode_minus']=False
33 missingno.bar(data, color=(190/255,190/255,190/255))
34 plt.tight_layout()
35
36
37 # In[5]:
38
39
40 data.replace({"起落架":{'DOWN':1},
41                 "空地电门0.2秒":{True:1,False:0},
42                 "空地电门0.4秒":{True:1,False:0},
43                 "空地电门0.6秒":{True:1,False:0},
```

```

44     "空地电门0.8秒":{True:1,False:0},
45     "空地电门1秒":{True:1,False:0},
46     "是否接通了A/T":{'DISENGD':0,'ENGAGED':1},
47     "是否接通了任意侧的A/P":{'OFF':0,'ON':1},
48 }, inplace=True)
49 data
50
51
52 # In[6]:
53
54
55 data=data.fillna(0)
56 data
57
58
59 # In[7]:
60
61
62 data.drop(labels=['月','日','起飞机场','落地机场','飞机重量'],axis=1,inplace=True)
63 data
64
65
66 # # QAR异常判断，剔除
67
68 # In[8]:
69
70
71 dup_row = data.duplicated(subset=['具体时间'], keep=False)
72 data.insert(0, 'is_dup', dup_row)
73 data[data['is_dup'] == True]
74
75
76 # In[9]:
77
78
79 data=data.drop_duplicates(subset=['具体时间'],keep='first')
80 data
81
82
83 # In[10]:
84
85
86 dup_row = data.duplicated(subset=['具体时间'], keep=False)
87 data.insert(0, 'is_dup_N', dup_row)

```

```

88 data[data['is_dup_N'] == True]
89
90
91 # In[11]:
92
93
94 def function(a, b):
95     if a == b:
96         return 1
97     else:
98         return 0
99
100
101 data['bool'] = data.apply(lambda x : function(x['俯仰角率'],x['俯仰角率.1']),axis = 1)
102 data
103
104
105 # In[12]:
106
107
108 data[data['bool']==0]
109
110
111 # In[13]:
112
113
114 data=data.drop(labels=['is_dup','is_dup_N','bool','具体时间','俯仰角率.1'],axis=1)
115 data


---


1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data=pd.read_csv('Data\\First\\NNew\\1-7D.csv')
15 data

```

```
16
17
18 # # 数据标准化
19
20 # In[3]:
21
22
23 import sklearn.preprocessing as sp
24 StandardTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
25 StandardTransform
26
27
28 # In[4]:
29
30
31 ListColumns=list(StandardTransform.columns)
32 ListColumns
33
34
35 # In[5]:
36
37
38 StandardTransformScaler = sp.StandardScaler()
39 StandardTransformScaler = StandardTransformScaler.fit(StandardTransform)
40 StandardTransform = StandardTransformScaler.transform(StandardTransform)
41 StandardTransform = pd.DataFrame(StandardTransform)
42 StandardTransform.columns = ListColumns
43 StandardTransform
44
45
46 # # 数据归一化
47
48 # In[6]:
49
50
51 import sklearn.preprocessing as sp
52 MinMaxTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
53 MinMaxTransform
54
55
56 # In[7]:
57
58
59 MinMaxTransformScaler=sp.MinMaxScaler(feature_range=(0.002, 1))
```

```

60 MinMaxTransformScaler=MinMaxTransformScaler.fit(MinMaxTransform)
61 MinMaxTransform=MinMaxTransformScaler.transform(MinMaxTransform)
62 MinMaxTransform=pd.DataFrame(MinMaxTransform)
63 MinMaxTransform.columns=ListColumns
64 MinMaxTransform
65
66
67 # # PCA&层次聚类
68
69 # In[8]:
70
71
72 import matplotlib.pyplot as plt
73
74
75 # In[9]:
76
77
78 from sklearn.decomposition import PCA
79
80 pca = PCA()
81 pca.fit(StandardTransform)
82 evr = pca.explained_variance_ratio_
83
84 plt.figure(figsize=(12, 5))
85 plt.plot(range(0, len(evr)), evr.cumsum(), marker="d", linestyle="--")
86 plt.xlabel("Number of components",font='Times New Roman',fontsize=13)
87 plt.ylabel("Cumulative explained variance",font='Times New Roman',fontsize=13)
88 plt.xticks(font='Times New Roman',fontsize=12)
89 plt.yticks(font='Times New Roman',fontsize=12)
90 plt.tight_layout()
91 plt.savefig("Figures\\1-7 PCA Cumulative explained variance.pdf")
92
93
94 # In[10]:
95
96
97 import scipy.cluster.hierarchy as sch
98 plt.figure(figsize=(12, 6))
99 dendrogram = sch.dendrogram(sch.linkage(StandardTransform.T, method = 'ward'))
100 plt.rcParams['font.sans-serif']=['SimHei']
101 plt.rcParams['axes.unicode_minus']=False
102 plt.xlabel('指标',fontsize=13)
103 plt.ylabel('离差平方和',fontsize=13)

```

```

104 plt.xticks(font='Times New Roman', fontsize=11)
105 plt.yticks(font='Times New Roman', fontsize=11)
106 plt.savefig("Figures\\1-7 层次聚类树状图.pdf")
107
108
109 # 环境: 风向、风速
110 # 人为操作: 下降率、地速、起落架、G值、杆量、盘量、RUDD位置
111 # 飞机状态: 姿态(俯仰角)、坡度、发动机N1值、磁航向、油门杆位置、下滑道偏差、航向道偏差、
112     俯仰角率
113
114
115
116 IFactor=MinMaxTransform.loc[:,['风向','风速','下降率','地速','起落架',
117     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒',
118     '着陆G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒',
119     '着陆G值0.9秒','着陆G值1秒',
120     '姿态(俯仰角)','姿态(俯仰角).1','姿态(俯仰角).2','姿态
121     (俯仰角).3','姿态(俯仰角).4',
122     '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
123     '坡度(左负右正)','坡度(左负右正).1',
124     '盘量','盘量.1','盘量.2','盘量.3','盘量.4',
125     '左侧发动机油门N1值','右侧发动机油门N1值','磁航向',
126     '左发油门杆位置(角度)','左发油门杆位置(角度).1','右发油门
127     杆位置(角度)','右发油门杆位置(角度).1',
128     'RUDD位置','下滑道偏差(C)','下滑道偏差(L)','下滑道偏差(R
129     )',
130     '航向道偏差(C)','航向道偏差(L)','航向道偏差(R),
131     俯仰角率']]]

132
133 IFactorEnv=IFactor.loc[:,['风向','风速']]
134 IFactorEnv
135
136
137 # In[13]:
138
139
140 IFactorPeo=IFactor.loc[:,['下降率','地速','起落架',
141     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆

```

```

G值0.5秒，'，着陆G值0.6秒，'，着陆G值0.7秒，'，着陆G
142 值0.8秒，'，着陆G
143 值0.9秒，'，着陆G值1秒，'
144 '杆量，'，杆量.1'，'杆量.2'，'杆量.3'，'杆量.4'，
145 '盘量，'，盘量.1'，'盘量.2'，'盘量.3'，'盘量.4'，'
146 'RUDD位置，]]]
147
148 # In[14]:
149
150
151 IFactorMac=IFactor.loc[:,~IFactor.columns.isin(['风向','风速','下降率','地速','起落架',
152 '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆
153 G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','着陆G
154 值0.9秒','着陆G值1秒'],
155 '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
156 '盘量','盘量.1','盘量.2','盘量.3','盘量.4','RUDD位置',)])]
157
158 # In[15]:
159
160
161 import copy
162 def ewm(data):
163     label_need = data.keys()[:]
164     data1 = data[label_need].values
165     data2 = data1
166     [m, n] = data2.shape
167     data3 = copy.deepcopy(data2)
168     p = copy.deepcopy(data3)
169     for j in range(0, n):
170         p[:, j] = data3[:, j] / sum(data3[:, j])
171     e = copy.deepcopy(data3[0, :])
172     for j in range(0, n):
173         e[j] = -1 / np.log(m) * sum(p[:, j] * np.log(p[:, j]))
174     w = (1 - e) / sum(1 - e)
175     total = 0
176     for sum_w in range(0, len(w)):
177         total = total + w[sum_w]
178     print(f'权重为: {w}, 权重之和为: {total}')
179
180
181 # In[16]:

```

```
182  
183  
184 ewm(IFactorEnv)  
185  
186  
187 # In[17]:  
188  
189  
190 ewm(IFactorPeo)  
191  
192  
193 # In[18]:  
194  
195  
196 ewm(IFactorMac)
```

D.8 航班 8 数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # # 附件一预处理（不包括异常值剔除及标准化）
5
6 # In[1]:
7
8
9 import pandas as pd
10 import numpy as np
11
12
13 # In[2]:
14
15
16 data=pd.read_excel("Data\\First\\New\\201404101159.xlsx",sheet_name='201404101159')
17 data
18
19
20 # In[3]:
21
22
23 data.info()
24
25
26 # In[4]:
27
28
29 import missingno
30 import matplotlib.pyplot as plt
31 plt.rcParams['font.sans-serif']=['SimHei']
32 plt.rcParams['axes.unicode_minus']=False
33 missingno.bar(data, color=(190/255,190/255,190/255))
34 plt.tight_layout()
35 plt.savefig('Figures\\附件1航班8缺失值.pdf')
36
37
38 # In[5]:
39
40
41 data=data.fillna(0)
42 data
43
```

```

44
45 # In[6]:
46
47
48 data.drop(labels=[‘月’, ‘日’, ‘起飞机场’, ‘落地机场’, ‘飞机重量’], axis=1, inplace=True)
49 data
50
51
52 # # QAR异常判断，剔除
53
54 # In[7]:
55
56
57 dup_row = data.duplicated(subset=[‘具体时间’], keep=False)
58 data.insert(0, ‘is_dup’, dup_row)
59 data[data[‘is_dup’] == True]
60
61
62 # In[8]:
63
64
65 data=data.drop_duplicates(subset=[‘具体时间’],keep=’first’)
66 data
67
68
69 # In[9]:
70
71
72 dup_row = data.duplicated(subset=[‘具体时间’], keep=False)
73 data.insert(0, ‘is_dup_N’, dup_row)
74 data[data[‘is_dup_N’] == True]
75
76
77 # In[10]:
78
79
80 def function(a, b):
81     if a == b:
82         return 1
83     else:
84         return 0
85
86
87 data[‘bool’] = data.apply(lambda x : function(x[‘俯仰角率’],x[‘俯仰角率.1’]),axis = 1)

```

```

88 data
89
90
91 # In[11]:
92
93
94 data[data['bool']==0]
95
96
97 # In[12]:
98
99
100 data=data.drop(labels=['is_dup','is_dup_N','bool','具体时间','俯仰角率.1'],axis=1)
101 data


---


1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data=pd.read_csv('Data\\First\\NNew\\1-8D.csv')
15 data
16
17
18 # # 数据标准化
19
20 # In[3]:
21
22
23 import sklearn.preprocessing as sp
24 StandardTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
25 StandardTransform
26
27
28 # In[4]:
29

```

```
30
31 ListColumns=list(StandardTransform.columns)
32 ListColumns
33
34
35 # In[5]:
36
37
38 StandardTransformScaler = sp.StandardScaler()
39 StandardTransformScaler = StandardTransformScaler.fit(StandardTransform)
40 StandardTransform = StandardTransformScaler.transform(StandardTransform)
41 StandardTransform = pd.DataFrame(StandardTransform)
42 StandardTransform.columns = ListColumns
43 StandardTransform
44
45
46 # # 数据归一化
47
48 # In[6]:
49
50
51 import sklearn.preprocessing as sp
52 MinMaxTransform = data.loc[:,~data.columns.isin(['Unnamed: 0'])]
53 MinMaxTransform
54
55
56 # In[7]:
57
58
59 MinMaxTransformScaler=sp.MinMaxScaler(feature_range=(0.002, 1))
60 MinMaxTransformScaler=MinMaxTransformScaler.fit(MinMaxTransform)
61 MinMaxTransform=MinMaxTransformScaler.transform(MinMaxTransform)
62 MinMaxTransform=pd.DataFrame(MinMaxTransform)
63 MinMaxTransform.columns=ListColumns
64 MinMaxTransform
65
66
67 # # PCA&层次聚类
68
69 # In[8]:
70
71
72 import matplotlib.pyplot as plt
73
```

```

74
75 # In[9]:
76
77
78 from sklearn.decomposition import PCA
79
80 pca = PCA()
81 pca.fit(StandardTransform)
82 evr = pca.explained_variance_ratio_
83
84 plt.figure(figsize=(12, 5))
85 plt.plot(range(0, len(evr)), evr.cumsum(), marker="d", linestyle="--")
86 plt.xlabel("Number of components",font='Times New Roman',fontsize=13)
87 plt.ylabel("Cumulative explained variance",font='Times New Roman',fontsize=13)
88 plt.xticks(font='Times New Roman',fontsize=12)
89 plt.yticks(font='Times New Roman',fontsize=12)
90 plt.tight_layout()
91 plt.savefig("Figures\\1-8 PCA Cumulative explained variance.pdf")
92
93
94 # In[10]:
95
96
97 import scipy.cluster.hierarchy as sch
98 plt.figure(figsize=(12, 6))
99 dendrogram = sch.dendrogram(sch.linkage(StandardTransform.T, method = 'ward'))
100 plt.rcParams['font.sans-serif']=['SimHei']
101 plt.rcParams['axes.unicode_minus']=False
102 plt.xlabel('指标',fontsize=13)
103 plt.ylabel('离差平方和',fontsize=13)
104 plt.xticks(font='Times New Roman',fontsize=11)
105 plt.yticks(font='Times New Roman',fontsize=11)
106 plt.savefig("Figures\\1-8 层次聚类树状图.pdf")
107
108
109 # 环境: 风向、风速
110 # 人为操作: 下降率、地速、起落架、G值、杆量、盘量、RUDD位置
111 # 飞机状态: 姿态（俯仰角）、坡度、发动机N1值、磁航向、油门杆位置、下滑道偏差、航向道偏差、
112 # 俯仰角率
113 # In[11]:
114
115
116 IFactor=MinMaxTransform.loc[:,['风向','风速','下降率','地速','起落架','着陆G值0.1秒','着

```

陆G值0.2秒，，着陆G值0.3秒，，着陆G值0.4秒，，着陆G值0.5秒，，着陆G值0.6秒，，着陆G值0.7秒
'，，着陆G值0.8秒，，着陆G值0.9秒，，着陆G值1秒，，姿态（俯仰角），，姿态（俯仰角）.1'，，姿态
(俯仰角).2'，，姿态（俯仰角）.3'，，姿态（俯仰角）.4'，，杆量，，杆量.1'，，杆量.2'，，杆量.3'，
，杆量.4'，，坡度（左负右正），，坡度（左负右正）.1'，，盘量，，盘量.1'，，盘量.2'，，盘量.3'，，盘
量.4'，，左侧发动机油门N1值，，右侧发动机油门N1值，，磁航向，，左发油门杆位置（角度），，左
发油门杆位置（角度）.1'，，右发油门杆位置（角度），，右发油门杆位置（角度）.1'，，RUDD位置，
，下滑道偏差（C），，下滑道偏差（L），，下滑道偏差（R），，航向道偏差（C），，航向道偏差（L
），，航向道偏差（R），，俯仰角率，]]

```
117 IFactor
118
119
120 # In[12]:
121
122
123 IFactorEnv=IFactor.loc[:,['风向','风速']]
124 IFactorEnv
125
126
127 # In[13]:
128
129
130 IFactorPeo=IFactor.loc[:,['下降率','地速','起落架',
131     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆
132     G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','着陆G
133     值0.9秒','着陆G值1秒',
134     '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
135     '盘量','盘量.1','盘量.2','盘量.3','盘量.4',
136     'RUDD位置',]]
137
138 # In[14]:
139
140
141 IFactorMac=IFactor.loc[:,~IFactor.columns.isin(['风向','风速','下降率','地速','起落架',
142     '着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆
143     G值0.5秒','着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','着陆G
144     值0.9秒','着陆G值1秒',
145     '杆量','杆量.1','杆量.2','杆量.3','杆量.4',
146     '盘量','盘量.1','盘量.2','盘量.3','盘量.4','RUDD位置',])]
147
148 # In[15]:
```

```

149
150
151 import copy
152 def ewm(data):
153     label_need = data.keys()[:]
154     data1 = data[label_need].values
155     data2 = data1
156     [m, n] = data2.shape
157     data3 = copy.deepcopy(data2)
158     p = copy.deepcopy(data3)
159     for j in range(0, n):
160         p[:, j] = data3[:, j] / sum(data3[:, j])
161     e = copy.deepcopy(data3[0, :])
162     for j in range(0, n):
163         e[j] = -1 / np.log(m) * sum(p[:, j] * np.log(p[:, j]))
164     w = (1 - e) / sum(1 - e)
165     total = 0
166     for sum_w in range(0, len(w)):
167         total = total + w[sum_w]
168     print(f'权重为: {w}, 权重之和为: {total}')
169
170
171 # In[16]:
172
173
174 ewm(IFactorEnv)
175
176
177 # In[17]:
178
179
180 ewm(IFactorPeo)
181
182
183 # In[18]:
184
185
186 ewm(IFactorMac)

```

D.9 问题二数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 def Sample(data):
15     data = data[['空地电门0.2秒', '空地电门0.4秒', '空地电门0.6秒', '空地电门0.8秒', '空地电门
16     1秒',
17             '着陆G值0.1秒', '着陆G值0.2秒', '着陆G值0.3秒', '着陆G值0.4秒', '着陆G值0.5秒',
18             '着陆G值0.6秒', '着陆G值0.7秒', '着陆G值0.8秒', '着陆G值0.9秒', '着陆G值1秒',
19             '姿态(俯仰角)', '姿态(俯仰角).1', '姿态(俯仰角).2', '姿态(俯仰角).3',
20             '姿态(俯仰角).4',
21             '杆量', '杆量.1', '杆量.2', '杆量.3', '杆量.4',
22             '坡度(左负右正)', '坡度(左负右正).1',
23             '盘量', '盘量.1', '盘量.2', '盘量.3', '盘量.4']]]

24     return data
25
26
27
28 data1 = pd.read_excel('Data\\Two\\LandingData\\1.xlsx', sheet_name='Sheet1')
29 data2 = pd.read_excel('Data\\Two\\LandingData\\2.xlsx', sheet_name='Sheet1')
30 data3 = pd.read_excel('Data\\Two\\LandingData\\3.xlsx', sheet_name='Sheet1')
31 data4 = pd.read_excel('Data\\Two\\LandingData\\4.xlsx', sheet_name='Sheet1')
32 data5 = pd.read_excel('Data\\Two\\LandingData\\5.xlsx', sheet_name='Sheet1')
33 data6 = pd.read_excel('Data\\Two\\LandingData\\6.xlsx', sheet_name='Sheet1')
34 data7 = pd.read_excel('Data\\Two\\LandingData\\7.xlsx', sheet_name='Sheet1')
35 data8 = pd.read_excel('Data\\Two\\LandingData\\8.xlsx', sheet_name='Sheet1')
36
37
38 # In[4]:
```

```

41 data1N = Sample(data1)
42 data2N = Sample(data2)
43 data3N = Sample(data3)
44 data4N = Sample(data4)
45 data5N = Sample(data5)
46 data6N = Sample(data6)
47 data7N = Sample(data7)
48 data8N = Sample(data8)

49
50
51 # In[5]:
52
53
54 data1N.to_csv('Data\\Two\\LandingData\\1N.csv', index=False)
55 data2N.to_csv('Data\\Two\\LandingData\\2N.csv', index=False)
56 data3N.to_csv('Data\\Two\\LandingData\\3N.csv', index=False)
57 data4N.to_csv('Data\\Two\\LandingData\\4N.csv', index=False)
58 data5N.to_csv('Data\\Two\\LandingData\\5N.csv', index=False)
59 data6N.to_csv('Data\\Two\\LandingData\\6N.csv', index=False)
60 data7N.to_csv('Data\\Two\\LandingData\\7N.csv', index=False)
61 data8N.to_csv('Data\\Two\\LandingData\\8N.csv', index=False)



---


1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9 import matplotlib.pyplot as plt
10
11
12 # In[2]:
13
14
15 data1=pd.read_excel('Data\\Two\\Landing\\1Landing.xlsx',sheet_name='Sheet1')
16 data2=pd.read_excel('Data\\Two\\Landing\\2Landing.xlsx',sheet_name='Sheet1')
17 data3=pd.read_excel('Data\\Two\\Landing\\3Landing.xlsx',sheet_name='Sheet1')
18 data4=pd.read_excel('Data\\Two\\Landing\\4Landing.xlsx',sheet_name='Sheet1')
19 data5=pd.read_excel('Data\\Two\\Landing\\5Landing.xlsx',sheet_name='Sheet1')
20 data6=pd.read_excel('Data\\Two\\Landing\\6Landing.xlsx',sheet_name='Sheet1')
21 data7=pd.read_excel('Data\\Two\\Landing\\7Landing.xlsx',sheet_name='Sheet1')
22 data8=pd.read_excel('Data\\Two\\Landing\\8Landing.xlsx',sheet_name='Sheet1')

```

```

23
24
25 # In[3]:
26
27
28 def draw(data,i):
29     data.fillna(method='ffill',axis = 0,inplace=True)
30     x=data['时刻']
31     y1=data['杆量']
32     y2=data['盘量']
33     y3=data['着陆G值']
34     y4=data['姿态（俯仰角）']
35     y5=data['坡度（左负右正）']
36     plt.rcParams['font.sans-serif']=['SimHei']
37     plt.rcParams['axes.unicode_minus']=False
38     plt.figure(figsize=(12, 6))
39     plt.plot(x, y1, ms=5, label='杆量', color="blue")
40     plt.plot(x, y2, ms=5, label='盘量', color="red")
41     plt.plot(x, y3, ms=5, label='着陆阶段G值', color="green")
42     plt.plot(x, y4, ms=5, label='姿态（俯仰角）', color="black")
43     plt.plot(x, y5, ms=5, label='坡度（左负右正）', color="orange")
44     plt.xlabel('时刻',size=13)
45     plt.ylabel('量值',size=13)
46     plt.xticks(font='Times New Roman',size=12)
47     plt.yticks(font='Times New Roman',size=12)
48     plt.legend()
49     plt.savefig('Pictures\\1-{} Landing.pdf'.format(i))
50
51
52 # In[4]:
53
54
55 draw(data1,1)
56
57
58 # In[5]:
59
60
61 draw(data2,2)
62
63
64 # In[6]:
65
66

```

```
67 draw(data3,3)
68
69
70 # In[7]:
71
72
73 draw(data4,4)
74
75
76 # In[8]:
77
78
79 draw(data5,5)
80
81
82 # In[9]:
83
84
85 draw(data6,6)
86
87
88 # In[10]:
89
90
91 draw(data7,7)
92
93
94 # In[11]:
95
96
97 draw(data8,8)
```

D.10 问题四数据分析

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data = pd.read_excel('Data\\Third\\附件3: 飞行参数测量数据（剔除C）.xlsx', sheet_name='
    数据')
15 data
16
17
18 # In[3]:
19
20
21 data.info()
22
23
24 # In[4]:
25
26
27 data.isnull().sum()
28
29
30 # In[5]:
31
32
33 import missingno
34 import matplotlib.pyplot as plt
35 plt.rcParams['font.sans-serif']=['SimHei']
36 plt.rcParams['axes.unicode_minus']=False
37 missingno.bar(data, color=(190/255,190/255,190/255))
38 plt.tight_layout()
39 plt.savefig('Figures\\附件3缺失值.pdf')
40
41
42 # In[6]:
```

```

43
44
45 data.drop(labels=[‘机型’, ‘落地主操控’, ‘V1_Method’, ‘Vr_Method’,
46             ‘EvtsEStoG2’, ‘EvtsG2toG3’, ‘EvtsTDG3toSD’, ‘TO_V1’,
47             ‘TO_V2’, ‘Max(CAS-Vfe)F1’, ‘Max(CAS-Vfe)F4’, ‘Max(CAS-Vfe)F5’,
48             ‘Max(CAS-Vfe)F6’, ‘Max(CAS-Vfe)F7’, ‘Max(CAS-Vfe)F8’, ‘Max(CAS-Vfe)F9’,
49             ,
50             ‘Max(CAS-Vfe)F10’, ‘Max(CAS-Vfe)Slat’, ‘CAS-V2atG1’, ‘Min(CAS-V2)
51             TOtoG1’,
52             ‘Max(CAS-V2)TOtoG1’, ‘(CAS-V2)atG2’, ‘Min(CAS-V2)G1toG2’, ‘Max(CAS-V2)
53             G1toG2’,
54             ‘TD_Vref’, ‘Max(CAS-Vfe)F2-1’, ‘Max(CAS-Vfe)F6-1’, ‘Max(CAS-Vfe)F7-1’,
55             ‘Max(CAS-Vfe)F8-1’, ‘Max(CAS-Vfe)F9-1’, ‘Max(CAS-Vfe)F10-1’, ‘Max(CAS-
56             Vfe)Slat-1’,
57             ‘SPDBKarmdTime’, ‘SPDBKarmdAAL’, ‘CAS-Vref_G3’, ‘CAS-Vref_G2’,
58             ‘Min(CAS-Vref)G3toG2’, ‘Max(CAS-Vref)G3toG2’, ‘Max(G_S)TDG3toG2’, ‘Min
59             (G_S)TDG3toG2’,
60             ‘CAS-Vref_G1’, ‘RoD_G1’, ‘Pitch_G1’, ‘Ave(PWR)_G1’,
61             ‘Min(CAS-Vref)G2toG1’, ‘Max(CAS-Vref)G2toG1’, ‘Max(G_S)TDG2toG1’, ‘Min
62             (G_S)TDG2toG1’,
63             ‘CAS-VrefAt50’, ‘CAS-VrefAtTD’, ‘Min(CAS-Vref)G1to50’, ‘Max(CAS-Vref)
64             G1toTD’,
65             ‘MaxGlideSG1to100’, ‘MinGlideSG1to100’,
66             ‘TO Gate 1’, ‘TO Gate 2’, ‘TD Gate 3’, ‘TD Gate 2’, ‘TD Gate 1-1’, ‘
67             TO_Vr’], axis=1, inplace=True)
68
69 # In[7]:
70
71
72 data.fillna(data.mode().iloc[0], inplace=True)
73 data
74
75
76 # In[8]:
77
78

```

```

79 data.isnull().sum()
80
81
82 # In[10]:
83
84
85 data
86
87
88 # In[11]:
89
90
91 #标签编码
92 import sklearn.preprocessing as sp
93 le=sp.LabelEncoder()
94
95 Competency=le.fit_transform(data["落地主操控人员资质"])
96 V2Method=le.fit_transform(data[" V2_Method"])
97 Vref_Method=le.fit_transform(data[" Vref_Method"])
98 RoDMethod=le.fit_transform(data[" RoD_Method"])
99 MACHMethod=le.fit_transform(data[" MACH_Method"])
100
101 data["落地主操控人员资质"]=pd.DataFrame(Competency)
102 data[" V2_Method"]=pd.DataFrame(V2Method)
103 data[" Vref_Method"]=pd.DataFrame(Vref_Method)
104 data[" RoD_Method"]=pd.DataFrame(RoDMethod)
105 data[" MACH_Method"]=pd.DataFrame(MACHMethod)
106 data
107
108
109 # In[12]:
110
111
112 #数据标准化
113 StandardTransform = data.loc[:,~data.columns.isin(['落地主操控人员资质',' V2_Method',' Vref_Method',' RoD_Method',' MACH_Method'])]
114 StandardTransform
115
116
117 # In[13]:
118
119
120 ListDataColumns=list(StandardTransform.columns)
121 ListDataColumns

```

```
122  
123  
124 # In[14]:  
125  
126  
127 StandardTransformScaler = sp.StandardScaler()  
128 StandardTransformScaler = StandardTransformScaler.fit(StandardTransform)  
129 StandardTransform = StandardTransformScaler.transform(StandardTransform)  
130 StandardTransform = pd.DataFrame(StandardTransform)  
131 StandardTransform.columns = ListDataColumns  
132 StandardTransform  
133  
134  
135 # In[15]:  
136  
137  
138 dataLeave=data[['落地主操控人员资质','V2_Method','Vref_Method','RoD_Method','  
    MACH_Method']]  
139 dataLeave  
140  
141  
142 # In[16]:  
143  
144  
145 dataNew=pd.concat([dataLeave, StandardTransform],axis=1)  
146 dataNew  
147  
148  
149 # In[17]:  
150  
151  
152 y=dataNew["落地主操控人员资质"]  
153 X=dataNew.loc[:,~dataNew.columns.isin(['落地主操控人员资质'])]  
154  
155  
156 # In[18]:  
157  
158  
159 #数据集划分  
160 from sklearn.model_selection import train_test_split  
161 X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.1,random_state=2023)  
162  
163  
164 # In[19]:
```

```

165
166
167 from sklearn.ensemble import RandomForestClassifier
168 from sklearn.tree import DecisionTreeClassifier
169
170 DecisionTree = DecisionTreeClassifier(random_state=2023)
171 DecisionTree = DecisionTree.fit(X_train, y_train)
172
173 RandomForest = RandomForestClassifier(random_state=2023)
174 RandomForest = RandomForest.fit(X_train, y_train)
175 RandomForest_score = RandomForest.score(X_test, y_test)
176 RandomForest_score
177
178
179 # In[20]:
180
181
182 std = np.std([i.feature_importances_ for i in RandomForest.estimators_], axis=0)
183 importances = DecisionTree.feature_importances_
184 feat_with_importance = pd.Series(importances, X.columns)
185 fig, ax = plt.subplots(figsize=(24,10))
186 feat_with_importance.plot.bar(yerr=std, ax=ax, color=(5/255,126/255,215/255))
187 ax.set_ylabel("Mean decrease in impurity",font='Times New Roman',size=14)
188 plt.yticks(font='Times New Roman',size=13)
189 plt.xticks(font='Times New Roman',size=13)
190 plt.tight_layout()
191 plt.savefig('Figures\\feature_importance_RF.pdf')
192
193
194 # In[21]:
195
196
197 from xgboost import XGBClassifier
198
199 XGB = XGBClassifier()
200 XGB.fit(X_train, y_train)
201 XGB_score = XGB.score(X_test, y_test)
202 XGB_score
203
204
205 # In[22]:
206
207
208 from xgboost import plot_importance

```

```

209 fig, ax = plt.subplots(figsize=(10,24))
210 plot_importance(XGB, height=0.4, ax=ax)
211 plt.xticks(fontsize=12, font='Times New Roman')
212 plt.yticks(fontsize=12, font='Times New Roman')
213 plt.tight_layout()
214 plt.savefig('Figures\\feature_importance_XGB.pdf')
215
216
217 # In[23]:
218
219
220 from sklearn.decomposition import PCA
221
222 pca = PCA()
223 pca.fit(X)
224 evr = pca.explained_variance_ratio_
225
226 plt.figure(figsize=(12, 5))
227 plt.plot(range(0, len(evr)), evr.cumsum(), marker="d", linestyle="--")
228 plt.xlabel("Number of components",font='Times New Roman',size=14)
229 plt.ylabel("Cumulative explained variance",font='Times New Roman',size=14)
230 plt.xticks(font='Times New Roman',size=12)
231 plt.yticks(font='Times New Roman',size=12)
232 plt.tight_layout()
233 plt.savefig('Figures\\资质PCA累计方差解释.pdf')
234
235
236 # In[24]:
237
238
239 #降维
240 y=dataNew["落地主操控人员资质"]
241 X=dataNew.loc[:,~dataNew.columns.isin(['落地主操控人员资质','CASgearselected','MinPRWTDG2toG1','MaxLeft(LOC)TDG2toG1','V2_Method','MaxRight(LOC)TDG3toG2'])]
242 X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.1,random_state=2023)
243
244
245 # In[25]:
246
247
248 from sklearn.ensemble import RandomForestClassifier
249 from sklearn.tree import DecisionTreeClassifier
250
251 DecisionTree = DecisionTreeClassifier(random_state=2023)

```

```

252 DecisionTree = DecisionTree.fit(X_train, y_train)
253
254 RandomForest = RandomForestClassifier(random_state=2023)
255 RandomForest = RandomForest.fit(X_train, y_train)
256 RandomForest_score = RandomForest.score(X_test, y_test)
257 RandomForest_score
258
259
260 # In[26]:
261
262
263 from xgboost import XGBClassifier
264
265 XGB = XGBClassifier(random_state=2023,)
266 XGB.fit(X_train, y_train)
267 XGB_score = XGB.score(X_test, y_test)
268 XGB_score
269
270
271 # In[27]:
272
273
274 from sklearn.metrics import mean_absolute_error
275 from sklearn.metrics import mean_squared_error
276
277
278 # In[28]:
279
280
281 print(f'XGBoost 平均绝对误差: ,
282     f'{mean_absolute_error(y_test, XGB.predict(X_test), sample_weight=None,
283         multioutput="uniform_average")}\n'
284     f'XGBoost 均方误差: ,
285     f'{mean_squared_error(y_test, XGB.predict(X_test), sample_weight=None, multioutput
286         ="uniform_average")}')
287
288
289
290 from yellowbrick.classifier import ROCAUC
291 classes=['A','F','J','M','T']
292 visualizer = ROCAUC(XGB, classes=classes)
293 visualizer.fit(X_train, y_train)

```

```

294 visualizer.score(X_test, y_test)
295 plt.xticks(font='Times New Roman')
296 plt.yticks(font='Times New Roman')
297 visualizer.show(outpath='Figures\\ROCAUC.pdf')
298
299
300 # In[30]:
301
302
303 from yellowbrick.classifier import ConfusionMatrix
304 classes=['A','F','J','M','T']
305 confusion_matrix = ConfusionMatrix(XGB, classes=classes)
306 confusion_matrix.fit(X_train, y_train)
307 confusion_matrix.score(X_test, y_test)
308 plt.xticks(font='Times New Roman',rotation=0)
309 plt.yticks(font='Times New Roman')
310 confusion_matrix.show(outpath='Figures\\ConfusionMatrix.pdf')
311
312
313 # In[31]:
314
315
316 from yellowbrick.classifier import ClassificationReport
317 visualizer = ClassificationReport(XGB, classes=classes, support=True, cmap='Greens')
318 visualizer.fit(X_train, y_train)
319 visualizer.score(X_test, y_test)
320 plt.xticks(font='Times New Roman')
321 plt.yticks(font='Times New Roman')
322 visualizer.show(outpath='Figures\\ClassificationReport.pdf')
323
324
325 # In[32]:
326
327
328 from sklearn.model_selection import cross_val_score
329 XGB_5K = cross_val_score(estimator=XGB,X=X_train,y=y_train,cv=5)
330 print(XGB_5K, '\n', XGB_5K.mean())
331 print(XGB_5K.std())

```

D.11 问题五仿真

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8 import numpy as np
9
10
11 # In[2]:
12
13
14 data1=pd.read_excel("Data\\First\\New\\201404070532.xlsx",sheet_name="201404070532")
15 data2=pd.read_excel("Data\\First\\New\\201404071917.xlsx",sheet_name="201404071917")
16 data3=pd.read_excel("Data\\First\\New\\201404080617.xlsx",sheet_name="201404080617")
17 data4=pd.read_excel("Data\\First\\New\\201404081034.xlsx",sheet_name="201404081034")
18 data5=pd.read_excel("Data\\First\\New\\201404090110.xlsx",sheet_name="201404090110")
19 data6=pd.read_excel("Data\\First\\New\\201404091701.xlsx",sheet_name="201404091701")
20 data7=pd.read_excel("Data\\First\\New\\201404100843.xlsx",sheet_name="201404100843")
21 data8=pd.read_excel("Data\\First\\New\\201404101159.xlsx",sheet_name="201404101159")
22
23
24 # In[3]:
25
26
27 def MaxFactor(data):
28     data['Max_G']=data[['着陆G值0.1秒', '着陆G值0.2秒', '着陆G值0.3秒', '着陆G值0.4秒', '着陆
29         G值0.5秒', '着陆G值0.6秒', '着陆G值0.7秒', '着陆G值0.8秒', '着陆G值0.9秒', '着陆G值1秒',
30         ]].max(axis=1)
31     data['Max_姿态']=data[['姿态（俯仰角）.1', '姿态（俯仰角）.2', '姿态
32         （俯仰角）.3', '姿态（俯仰角）.4']].max(axis=1)
33     data['Max_杆量']=data[['杆量.1', '杆量.2', '杆量.3', '杆量.4']].max(axis=1)
34     data['Max_坡度']=data[['坡度（左负右正）.1', '坡度（左负右正）.2']].max(axis=1)
35     data['Max_盘量']=data[['盘量.1', '盘量.2', '盘量.3', '盘量.4']].max(axis=1)
36     return data
37
38
39 # In[4]:
40
41
42 data1A=MaxFactor(data1)
43 data2A=MaxFactor(data2)
```

```

41 data3A=MaxFactor(data3)
42 data4A=MaxFactor(data4)
43 data5A=MaxFactor(data5)
44 data6A=MaxFactor(data6)
45 data7A=MaxFactor(data7)
46 data8A=MaxFactor(data8)

47
48
49 # In[5]:
50
51
52 def FactorSelect(data):
53     New=data[['海拔高度','无线电高度',
54             '计算空速','地速',
55             'Max_G','Max_姿态','Max_杆量','Max_坡度','Max_盘量',
56             '风向','风速',
57             '下滑道偏差 (C)',,'下滑道偏差 (L)',,'下滑道偏差 (R)',,
58             '航向道偏差 (C)',,'航向道偏差 (L)',,'航向道偏差 (R)',,
59             '俯仰角率',,'下降率',,'飞机重量']]
60     return New
61
62
63 # In[6]:
64
65
66 data1AN=FactorSelect(data1A)
67 data2AN=FactorSelect(data2A)
68 data3AN=FactorSelect(data3A)
69 data4AN=FactorSelect(data4A)
70 data5AN=FactorSelect(data5A)
71 data6AN=FactorSelect(data6A)
72 data7AN=FactorSelect(data7A)
73 data8AN=FactorSelect(data8A)

74
75
76 # In[7]:
77
78
79 import matplotlib.pyplot as plt
80
81
82 # In[8]:
83
84

```

```

85 def draw_HBWXD(data,i):
86     y1=data['海拔高度']
87     y2=data['无线电高度']
88     plt.rcParams['font.sans-serif']=['SimHei']
89     plt.rcParams['axes.unicode_minus']=False
90     plt.figure(figsize=(14, 4))
91     plt.plot(y1, ms=5, label='海拔高度', color="blue")
92     plt.plot(y2, ms=5, label='无线电高度', color="red")
93     plt.xlabel('时刻',size=13)
94     plt.ylabel('量值',size=13)
95     plt.xticks(font='Times New Roman',size=12)
96     plt.yticks(font='Times New Roman',size=12)
97     plt.legend()
98     plt.savefig("Figures\\全航段\\第"+str(i)+"次航班海拔高度与无线电高度.pdf")
99
100
101 # In[9]:
102
103
104 def draw_KSDS(data,i):
105     y1=data['计算空速']
106     y2=data['地速']
107     y3=data['风速']
108     plt.rcParams['font.sans-serif']=['SimHei']
109     plt.rcParams['axes.unicode_minus']=False
110     plt.figure(figsize=(14, 4))
111     plt.plot(y1, ms=5, label='计算空速', color="blue")
112     plt.plot(y2, ms=5, label='地速', color="red")
113     plt.plot(y3, ms=5, label='风速', color="green")
114     plt.xlabel('时刻',size=13)
115     plt.ylabel('量值',size=13)
116     plt.xticks(font='Times New Roman',size=12)
117     plt.yticks(font='Times New Roman',size=12)
118     plt.legend()
119     plt.savefig("Figures\\全航段\\第"+str(i)+"次航班空速、地速、风速.pdf")
120
121
122 # In[10]:
123
124
125 def draw_FV(data,i):
126     y1=data['Max_G']
127     y2=data['Max_姿态']
128     y3=data['Max_杆量']

```

```

129     y4=data['Max_坡度']
130     y5=data['Max_盘量']
131     plt.rcParams['font.sans-serif']=['SimHei']
132     plt.rcParams['axes.unicode_minus']=False
133     plt.figure(figsize=(14, 5))
134     plt.plot(y1, ms=5, label='G', color="green")
135     plt.plot(y2, ms=5, label='姿态', color="black")
136     plt.plot(y3, ms=5, label='杆量', color="blue")
137     plt.plot(y4, ms=5, label='坡度', color="orange")
138     plt.plot(y5, ms=5, label='盘量', color="red")
139     plt.xlabel('时刻',size=13)
140     plt.ylabel('量值',size=13)
141     plt.xticks(font='Times New Roman',size=12)
142     plt.yticks(font='Times New Roman',size=12)
143     plt.legend()
144     plt.savefig("Figures\\全航段\\第"+str(i)+"次航班飞行参量.pdf")
145
146
147 # In[11]:
148
149
150 def draw_XHDPC(data,i):
151     y1=data['下滑道偏差 (C) ']
152     y2=data['下滑道偏差 (L) ']
153     y3=data['下滑道偏差 (R) ']
154     plt.rcParams['font.sans-serif']=['SimHei']
155     plt.rcParams['axes.unicode_minus']=False
156     plt.figure(figsize=(14, 4))
157     plt.plot(y1, ms=5, label='下滑道偏差 (C)', color="blue")
158     plt.plot(y2, ms=5, label='下滑道偏差 (L)', color="red")
159     plt.plot(y3, ms=5, label='下滑道偏差 (R)', color="orange")
160     plt.xlabel('时刻',size=13)
161     plt.ylabel('量值',size=13)
162     plt.xticks(font='Times New Roman',size=12)
163     plt.yticks(font='Times New Roman',size=12)
164     plt.legend()
165     plt.savefig("Figures\\全航段\\第"+str(i)+"次航班下滑道偏差.pdf")
166
167
168 # In[12]:
169
170
171 def draw_HXDPD(data,i):
172     y1=data['航向道偏差 (C) ']

```

```

173     y2=data[‘航向道偏差（L）’]
174     y3=data[‘航向道偏差（R）’]
175     plt.rcParams[‘font.sans-serif’]=[‘SimHei’]
176     plt.rcParams[‘axes.unicode_minus’]=False
177     plt.figure(figsize=(14, 4))
178     plt.plot(y1, ms=5, label=‘航向道偏差（C）’, color=“green”)
179     plt.plot(y2, ms=5, label=‘航向道偏差（L）’, color=“red”)
180     plt.plot(y3, ms=5, label=‘航向道偏差（R）’, color=“orange”)
181     plt.xlabel(‘时刻’,size=13)
182     plt.ylabel(‘量值’,size=13)
183     plt.xticks(font=‘Times New Roman’,size=12)
184     plt.yticks(font=‘Times New Roman’,size=12)
185     plt.legend()
186     plt.savefig(“Figures\\全航段\\第”+str(i)+“次航班航向道偏差.pdf”)
187
188
189 # In[13]:
190
191
192 def draw_FYJXJ(data,i):
193     y1=data[‘俯仰角率’]
194     y2=data[‘下降率’]
195     plt.rcParams[‘font.sans-serif’]=[‘SimHei’]
196     plt.rcParams[‘axes.unicode_minus’]=False
197     plt.figure(figsize=(14, 4))
198     plt.plot(y1, ms=5, label=‘俯仰角率’, color=“green”)
199     plt.plot(y2, ms=5, label=‘下降率’, color=“red”)
200     plt.xlabel(‘时刻’,size=13)
201     plt.ylabel(‘量值’,size=13)
202     plt.xticks(font=‘Times New Roman’,size=12)
203     plt.yticks(font=‘Times New Roman’,size=12)
204     plt.legend()
205     plt.savefig(“Figures\\全航段\\第”+str(i)+“次航班俯仰角率、下降率.pdf”)
206
207
208 # In[14]:
209
210
211 def draw_M(data,i):
212     y=data[‘飞机重量’]
213     plt.rcParams[‘font.sans-serif’]=[‘SimHei’]
214     plt.rcParams[‘axes.unicode_minus’]=False
215     plt.figure(figsize=(14, 4))
216     plt.plot(y, ms=5, label=‘飞机重量’, color=“green”)

```

```

217     plt.xlabel('时刻',size=13)
218     plt.ylabel('量值',size=13)
219     plt.xticks(font='Times New Roman',size=12)
220     plt.yticks(font='Times New Roman',size=12)
221     plt.legend()
222     plt.savefig("Figures\\全航段\\第"+str(i)+"次航班飞机重量.pdf")
223
224
225 # # 飞机重量
226
227 # In[15]:
228
229
230 draw_M(data1AN,1)
231 draw_M(data2AN,2)
232 draw_M(data3AN,3)
233 draw_M(data4AN,4)
234 draw_M(data5AN,5)
235 draw_M(data6AN,6)
236 draw_M(data7AN,7)
237 draw_M(data8AN,8)
238
239
240 # # 下降率、俯仰角率
241
242 # In[16]:
243
244
245 draw_FYJXJ(data1AN,1)
246 draw_FYJXJ(data2AN,2)
247 draw_FYJXJ(data3AN,3)
248 draw_FYJXJ(data4AN,4)
249 draw_FYJXJ(data5AN,5)
250 draw_FYJXJ(data6AN,6)
251 draw_FYJXJ(data7AN,7)
252 draw_FYJXJ(data8AN,8)
253
254
255 # # 航向道偏差
256
257 # In[17]:
258
259
260 draw_HXDPC(data1AN,1)

```

```
261 draw_HXDPC(data2AN,2)
262 draw_HXDPC(data3AN,3)
263 draw_HXDPC(data4AN,4)
264 draw_HXDPC(data5AN,5)
265 draw_HXDPC(data6AN,6)
266 draw_HXDPC(data7AN,7)
267 draw_HXDPC(data8AN,8)
268
269
270 # # 下滑道偏差
271
272 # In[18]:
273
274
275 draw_XHDPC(data1AN,1)
276 draw_XHDPC(data2AN,2)
277 draw_XHDPC(data3AN,3)
278 draw_XHDPC(data4AN,4)
279 draw_XHDPC(data5AN,5)
280 draw_XHDPC(data6AN,6)
281 draw_XHDPC(data7AN,7)
282 draw_XHDPC(data8AN,8)
283
284
285 # # 五因素
286
287 # In[19]:
288
289
290 draw_FV(data1AN,1)
291 draw_FV(data2AN,2)
292 draw_FV(data3AN,3)
293 draw_FV(data4AN,4)
294 draw_FV(data5AN,5)
295 draw_FV(data6AN,6)
296 draw_FV(data7AN,7)
297 draw_FV(data8AN,8)
298
299
300 # # 速度
301
302 # In[20]:
303
304
```

```
305 draw_KSDS(data1AN,1)
306 draw_KSDS(data2AN,2)
307 draw_KSDS(data3AN,3)
308 draw_KSDS(data4AN,4)
309 draw_KSDS(data5AN,5)
310 draw_KSDS(data6AN,6)
311 draw_KSDS(data7AN,7)
312 draw_KSDS(data8AN,8)

313
314
315 # # 海拔、无线电高度
316
317 # In[21]:
318
319
320 draw_HBWXD(data1AN,1)
321 draw_HBWXD(data2AN,2)
322 draw_HBWXD(data3AN,3)
323 draw_HBWXD(data4AN,4)
324 draw_HBWXD(data5AN,5)
325 draw_HBWXD(data6AN,6)
326 draw_HBWXD(data7AN,7)
327 draw_HBWXD(data8AN,8)
```

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # In[1]:
5
6
7 import pandas as pd
8
9
10 # In[2]:
11
12
13 data1=pd.read_excel("Data\\First\\New\\201404070532.xlsx",sheet_name="201404070532")
14 data2=pd.read_excel("Data\\First\\New\\201404071917.xlsx",sheet_name="201404071917")
15 data3=pd.read_excel("Data\\First\\New\\201404080617.xlsx",sheet_name="201404080617")
16 data4=pd.read_excel("Data\\First\\New\\201404081034.xlsx",sheet_name="201404081034")
17 data5=pd.read_excel("Data\\First\\New\\201404090110.xlsx",sheet_name="201404090110")
18 data6=pd.read_excel("Data\\First\\New\\201404091701.xlsx",sheet_name="201404091701")
19 data7=pd.read_excel("Data\\First\\New\\201404100843.xlsx",sheet_name="201404100843")
20 data8=pd.read_excel("Data\\First\\New\\201404101159.xlsx",sheet_name="201404101159")
```

```

21 dataAll=pd.read_excel("Data\\First\\New\\汇总.xlsx",sheet_name="Sheet1")
22
23
24 # In[3]:
25
26
27 def MaxFactor(data):
28     data['G值']=data[['着陆G值0.1秒','着陆G值0.2秒','着陆G值0.3秒','着陆G值0.4秒','着陆G
29         值0.5秒',
30             '着陆G值0.6秒','着陆G值0.7秒','着陆G值0.8秒','着陆G值0.9秒','着陆G值1
31             秒']].max(axis=1)
32     data['姿态']=data[['姿态（俯仰角）.1','姿态（俯仰角）.2','姿态（俯仰
33         角）.3','姿态（俯仰角）.4']].max(axis=1)
34     data['杆量']=data[['杆量.1','杆量.2','杆量.3','杆量.4']].max(axis=1)
35     data['坡度']=data[['坡度（左负右正）.1']].max(axis=1)
36     data['盘量']=data[['盘量.1','盘量.2','盘量.3','盘量.4']].max(axis=1)
37     data['左油门杆位置']=data[['左发油门杆位置（角度）.1']].max(
38         axis=1)
39     data['右油门杆位置']=data[['右发油门杆位置（角度）.1']].max(
40         axis=1)
41
42     return data
43
44
45
46
47
48
49
50
51 # In[4]:
52
53
54 New=data[['具体时间',
55             '计算空速','G值','姿态','杆量','坡度','盘量',
56             '下滑道偏差（C）','下滑道偏差（L）','下滑道偏差（R）',
57             '航向道偏差（C）','航向道偏差（L）','航向道偏差（R）',
58             '俯仰角率','下降率','RUDD位置','左油门杆位置','右油门杆位置']]
59

```

```

60 dataAF=FactorSelect(dataAI)
61
62
63 # In[7]:
64
65
66 dataAF
67
68
69 # In[8]:
70
71
72 import matplotlib.pyplot as plt
73 plt.rcParams['font.sans-serif']=['SimHei']
74 plt.rcParams['axes.unicode_minus']=False
75
76
77 # In[9]:
78
79
80 def drawHist(x):
81     plt.figure(figsize=(10,8))
82     dataAF.hist(column=x,bins=150,color='#9ACD32',grid=False)
83     plt.xticks(font='Times New Roman',fontsize=12)
84     plt.yticks(font='Times New Roman',fontsize=12)
85     plt.savefig('Figures\\{}.pdf'.format(x))
86
87
88 # In[10]:
89
90
91 drawHist('计算空速')
92 drawHist('G值')
93 drawHist('姿态')
94 drawHist('杆量')
95 drawHist('坡度')
96 drawHist('盘量')
97 drawHist('下滑道偏差 (C)')
98 drawHist('下滑道偏差 (L)')
99 drawHist('下滑道偏差 (R)')
100 drawHist('航向道偏差 (C)')
101 drawHist('航向道偏差 (L)')
102 drawHist('航向道偏差 (R)')
103 drawHist('俯仰角率')

```

```

104 drawHist('下降率')
105 drawHist('RUDD位置')
106 drawHist('左油门杆位置')
107 drawHist('右油门杆位置')

108
109
110 # In[11]:
111
112
113 def calculateRateTh(x):
114     t=dataAF[x].diff()
115     mean=t.mean()
116     std=t.std()
117     max=mean+3*std
118     min=mean-3*std
119     return min,max

120
121
122 # In[12]:
123
124
125 def calculateTh(x):
126     mean=dataAF[x].mean()
127     std=dataAF[x].std()
128     max=mean+3*std
129     min=mean-3*std
130     return min,max

131
132
133 # In[13]:
134
135
136 RateTh=['姿态','坡度','RUDD位置','左油门杆位置','右油门杆位置']
137 Th=['G值','杆量','盘量','下滑道偏差 (C)','下滑道偏差 (L)','下滑道偏差 (R)',
138 '航向道偏差 (C)','航向道偏差 (L)','航向道偏差 (R)','俯仰角率','下降率']

139
140
141 # In[14]:
142
143
144 for i in RateTh:
145     min,max=calculateRateTh(i)
146     print(i,min,max)

147

```

```

148
149 # In[15]:
150
151
152 for i in Th:
153     min,max=calculateTh(i)
154     print(i,min,max)
155
156
157 # 姿态 -0.1823728785192135 0.18236945078921915
158 # 坡度 -0.6673918000300177 0.6673918000300177
159 # RUDD位置 -0.06688924609154025 0.06688922658809789
160 # 左油门杆位置 -0.7397604532999682 0.7397604532999682
161 # 右油门杆位置 -0.7364677711308804 0.7364677711308804
162 #
163 # G值 0.9537159362101948 1.0635978346157227
164 # 杆量 -0.045490727710496254 0.044095951682060375
165 # 盘量 -0.061697333362055275 0.054986543622766275
166 # 下滑道偏差 (C) -3.272516829802762 3.346957205632264
167 # 下滑道偏差 (L) -3.088455218634214 3.147513401994933
168 # 下滑道偏差 (R) -3.0607728912464105 3.1096427210309474
169 # 航向道偏差 (C) -4.9660114773697135 4.740072025511289
170 # 航向道偏差 (L) -4.952541392349814 4.725793916814326
171 # 航向道偏差 (R) -4.961188798370152 4.756167466584085
172 # 俯仰角率 -0.269455167818758 0.2101911997652408
173 # 下降率 -1660.3728475810756 1660.7590236085364
174
175 # In[16]:
176
177
178 def zitai(x):
179     if x<=-0.1823728785192135 or x>=0.18236945078921915:
180         return '异常'
181     else:
182         return '正常'
183
184
185 def podu(x):
186     if x<=-0.6673918000300177 or x>=0.6673918000300177:
187         return '异常'
188     else:
189         return '正常'
190
191

```

```

192 def rudd(x):
193     if x<=-0.06688924609154025 or x>=0.06688922658809789:
194         return '异常',
195     else:
196         return '正常',
197
198
199 def zuoyoumengan(x):
200     if x<=-0.7397604532999682 or x>=0.7397604532999682:
201         return '异常',
202     else:
203         return '正常',
204
205
206 def youyoumengan(x):
207     if x<=-0.7364677711308804 or x>=0.7364677711308804:
208         return '异常',
209     else:
210         return '正常',
211
212
213 def g(x):
214     if x<=0.9537159362101948:
215         return '失重',
216     elif x>=1.0635978346157227:
217         return '超重',
218     else:
219         return '正常',
220
221
222 def gangliang(x):
223     if x<=-0.045490727710496254:
224         return '拉杆过度',
225     elif x>=0.044095951682060375:
226         return '松杆(推杆)过度',
227     else:
228         return '正常',
229
230
231 def panliang(x):
232     if x<=-0.061697333362055275:
233         return '左旋过度',
234     elif x>=0.054986543622766275:
235         return '右旋过度',

```

```

236     else:
237         return '正常',
238
239
240 def xiahuadaoC(x):
241     if x<=-3.272516829802762 or x>=3.346957205632264:
242         return '异常C'
243     else:
244         return '正常',
245
246
247 def xiahuadaoL(x):
248     if x<=-3.088455218634214 or x>=3.147513401994933:
249         return '异常L'
250     else:
251         return '正常',
252
253
254 def xiahuadaoR(x):
255     if x<=-3.0607728912464105 or x>=3.1096427210309474:
256         return '异常R'
257     else:
258         return '正常',
259
260
261 def hangxiangdaoC(x):
262     if x<=-4.9660114773697135 or x>=4.740072025511289:
263         return '异常C'
264     else:
265         return '正常',
266
267
268 def hangxiangdaoL(x):
269     if x<=-4.952541392349814 or x>=4.725793916814326:
270         return '异常L'
271     else:
272         return '正常',
273
274
275 def hangxiangdaoR(x):
276     if x<=-4.961188798370152 or x>=4.756167466584085:
277         return '异常R'
278     else:
279         return '正常',

```

```

280
281
282 def fuyangjiaolv(x):
283     if x<=-0.269455167818758 or x>=0.2101911997652408:
284         return '异常'
285     else:
286         return '正常'
287
288
289 def xiajianglv(x):
290     if x<=-1660.3728475810756 or x>=1660.7590236085364:
291         return '异常'
292     else:
293         return '正常'
294
295
296 # In[17]:
297
298
299 def System(data):
300     data=FactorSelect(MaxFactor(data))
301     data['飞行姿态变化']=data['姿态'].diff().apply(lambda x:zitai(x))
302     data['坡度变化']=data['坡度'].diff().apply(lambda x:podu(x))
303     data['RUDD位置变化']=data['RUDD位置'].diff().apply(lambda x:rudd(x))
304     data['左油门杆位置变化']=data['左油门杆位置'].diff().apply(lambda x:zuoyoumengan(x))
305     data['右油门杆位置变化']=data['右油门杆位置'].diff().apply(lambda x:youyoumengan(x))
306     data['失重超重情况']=data['G值'].apply(lambda x:g(x))
307     data['杆量情况']=data['杆量'].apply(lambda x:gangliang(x))
308     data['盘量情况']=data['盘量'].apply(lambda x:panliang(x))
309     data['下滑道偏差 (C) 情况']=data['下滑道偏差 (C) '].apply(lambda x:xiahuadaoC(x))
310     data['下滑道偏差 (L) 情况']=data['下滑道偏差 (L) '].apply(lambda x:xiahuadaoL(x))
311     data['下滑道偏差 (R) 情况']=data['下滑道偏差 (R) '].apply(lambda x:xiahuadaoR(x))
312     data['航向道偏差 (C) 情况']=data['航向道偏差 (C) '].apply(lambda x:hangxiangdaoC(x))
313     data['航向道偏差 (L) 情况']=data['航向道偏差 (L) '].apply(lambda x:hangxiangdaoL(x))
314     data['航向道偏差 (R) 情况']=data['航向道偏差 (R) '].apply(lambda x:hangxiangdaoR(x))
315     data['俯仰角率情况']=data['俯仰角率'].apply(lambda x:fuyangjiaolv(x))
316     data['下降率情况']=data['下降率'].apply(lambda x:xiajianglv(x))
317     return data
318
319
320 # In[18]:
321
322
323 data1V=System(data1)

```

```
324 data2V=System(data2)
325 data3V=System(data3)
326 data4V=System(data4)
327 data5V=System(data5)
328 data6V=System(data6)
329 data7V=System(data7)
330 data8V=System(data8)

331
332
333 # In[19]:
334
335
336 data1V
337
338
339 # In[20]:
340
341
342 data2V
343
344
345 # In[21]:
346
347
348 data3V
349
350
351 # In[22]:
352
353
354 data4V
355
356
357 # In[23]:
358
359
360 data5V
361
362
363 # In[24]:
364
365
366 data6V
367
```

```
368  
369 # In[25]:  
370  
371  
372 data7V  
373  
374  
375 # In[26]:  
376  
377  
378 data8V
```

D.12 南丁格尔玫瑰图

```
1 import pandas as pd
2 from pyecharts.charts import Pie
3 from pyecharts import options as opts
4 from snapshot_phantomjs import snapshot
5 from pyecharts.render import make_snapshot
6
7
8 def draw_pie(grade, num):
9     color_series = ['#14ADCF', '#209AC9', '#1E91CA', '#2C6BA0', '#2B55A1', '#2D3D8E', ,
10                   '#44388E', '#6A368B', '#7D3990', '#A63F98', '#C31C88', '#D52178', '#D5225B']
11
12     df = pd.DataFrame({'grade': grade, 'num': num})
13     df.sort_values(by='num', ascending=False, inplace=True)
14     v = df['grade'].values.tolist()
15     d = df['num'].values.tolist()
16
17     pie1 = Pie(init_opts=opts.InitOpts(width='880px', height='800px'))
18     pie1.set_colors(color_series)
19     pie1.add("", [list(z) for z in zip(v, d)],
20               radius=["20%", "80%"],
21               center=["50%", "60%"],
22               rosetype="area")
23
24     pie1.set_global_opts(legend_opts=opts.LegendOpts(is_show=False), toolbox_opts=opts.
25                          ToolboxOpts())
26
27     pie1.set_series_opts(label_opts=opts.LabelOpts(is_show=True, position="inside",
28                           font_size=12, formatter="{b} 阶段超限 {c} 次", font_style="italic", font_weight="bold",
29                           font_family="Microsoft YaHei"), )
30
31     make_snapshot(snapshot, pie1.render(), 'Figures\\NightingaleRoseDiagram.png',
32                   is_remove_html=True)
33
34
35     draw_pie(['着陆', '空中', '进近', '(着陆)地面', '未知', '(起飞)地面', '起飞',
36               ], [26406, 11119, 5591, 398, 283, 158, 148])
```
