

第十二届“中国软件杯”大学生软件设计大赛

A10-基于机器学习的分布式系统故障诊断系统

设计文档



参赛队员：姜昕坤、吴启通、齐新凯、刘佰航

指导教师：刘连山

所在单位：山东科技大学

申报时间：2023 年 7 月

目录

第 1 章 队伍信息介绍	3
第 2 章 模型介绍	3
2.1 模型介绍	3
2.2 数据处理	4
第 3 章 实现思路	4
3.1 技术概要	5
3.1.1 主要功能	5
3.1.2 技术路线	5
3.1.2 web 系统使用展示	6
3.2 技术方案	13
3.2.1 模型部分	13
3.2.1 web 部分	14
3.3 部署 Web	16
3.3.1 Python 环境部署	16
3.3.2 相关环境配置	17
3.3.3 Windows 版本 Redis 的安装	17
3.3.4 负载均衡	18
3.3.5 注意事项	19
3.4 平台架构	19
3.5 总结	20
第 4 章 原理分析	20
4.1 GBDT 算法	20
4.2 CatBoost 算法	22
4.3 特征工程-CatBoost 算法	25
4.3.1 过滤式特征选择	25
4.3.2 互信息	25
4.3.3 高相关过滤、低方差过滤	27
4.3.4 创新点	28
第 5 章 结果分析	29
5.1 验证集结果	29
5.2 测试集结果	30
第 6 章 重难点突破及未来展望	30
6.1 重难点突破	30
6.2 未来展望	30

第 1 章 队伍信息介绍

团队介绍					
团队 主要 成员	姓名	性别	年龄	年级、专业	分工
	姜昕坤	男	20	2021 级计算机科学与技术专业	数据处理、特征工程、模型选择与优化
	吴启通	男	19	2021 级计算机科学与技术专业	数据处理、特征工程、模型选择与优化
	齐新凯	男	19	2021 级计算机科学与技术专业	Web 系统开发
	刘佰航	男	21	2020 级信息安全专业	Web 系统开发

第 2 章 模型介绍

2.1 模型介绍

CatBoost 是俄罗斯的搜索巨头 Yandex 在 2017 年开源的机器学习库, 它是一种基于对称决策树 (oblivious trees) 的为基学习器实现的参数较少、支持类别型变量和高准确性的 GBDT 框架, 主要优点是高效合理地处理类别型特征, 另外提出了新的方法来处理梯度偏差 (Gradient bias) 以及预测偏移 (Prediction shift) 问题, 提高算法的准确性和泛化能力。以下是几种常用的机器学习模型在学习各大网站数据之后的预测结果:










	CatBoost		LightGBM		XGBoost		H2O	
	Tuned	Default	Tuned	Default	Tuned	Default	Tuned	Default
 Adult	0.26974	0.27298 +1.21%	0.27602 +2.33%	0.28716 +6.46%	0.27542 +2.11%	0.28009 +3.84%	0.27510 +1.99%	0.27607 +2.35%
 Amazon	0.13772	0.13811 +0.29%	0.16360 +18.80%	0.16716 +21.38%	0.16327 +18.56%	0.16536 +20.07%	0.16264 +18.10%	0.16950 +23.08%
 Click prediction	0.39090	0.39112 +0.06%	0.39633 +1.39%	0.39749 +1.69%	0.39624 +1.37%	0.39764 +1.73%	0.39759 +1.72%	0.39785 +1.78%
 KDD appetency	0.07151	0.07138 -0.19%	0.07179 +0.40%	0.07482 +4.63%	0.07176 +0.35%	0.07466 +4.41%	0.07246 +1.33%	0.07355 +2.86%
 KDD churn	0.23129	0.23193 +0.28%	0.23205 +0.33%	0.23565 +1.89%	0.23312 +0.80%	0.23369 +1.04%	0.23275 +0.64%	0.23287 +0.69%
 KDD internet	0.20875	0.22021 +5.49%	0.22315 +6.90%	0.23627 +13.19%	0.22532 +7.94%	0.23468 +12.43%	0.22209 +6.40%	0.24023 +15.09%
 KDD upselling	0.16613	0.16674 +0.37%	0.16682 +0.42%	0.17107 +2.98%	0.16632 +0.12%	0.16873 +1.57%	0.16824 +1.28%	0.16981 +2.22%
 KDD 98	0.19467	0.19479 +0.07%	0.19576 +0.56%	0.19837 +1.91%	0.19568 +0.52%	0.19795 +1.69%	0.19539 +0.37%	0.19607 +0.72%
 Kick prediction	0.28479	0.28491 +0.05%	0.29566 +3.82%	0.29877 +4.91%	0.29465 +3.47%	0.29816 +4.70%	0.29481 +3.52%	0.29635 +4.06%

图 2.1 tuned 代表调优后的结果，Default 代表默认值。

本表中的数字表示分类模式的 Logloss 值(越低越好)。百分比是根据的 CatBoost 模型测量结果的度量差异。

2.2 数据处理

数据处理是构建机器学习模型的关键步骤之一，它涉及对原始数据进行一系列的预处理操作。对原始数据集分析可知，各个样本特征之间的数据相差过大，首先对原始数据集进行标准化处理，将不同尺度、分布的数据转化为具有统一尺度的数据，从而提升模型的性能和准确性。

针对原始数据中的缺失值，分别采用均值、中位数、众数以及 0 进行填充，效果如表 2.1 所示。

表 2.1 不同缺失值填充方法的效果对比

训练集	验证集	缺失值填充方法	模型	验证集F1
preprocess_train	validate_1000	中位数	catboost	60.45
preprocess_train	validate_1000	众数	catboost	69.14
preprocess_train	validate_1000	均值	catboost	70.83
preprocess_train	validate_1000	0	catboost	79.91

因此本作品采取标准化处理后的零填充方法处理缺失值。

第 3 章 实现思路

3.1 技术概要

3.1.1 主要功能

本系统为基于机器学习的分布式系统故障检测系统，在我们的系统中，可以实现对故障样本的故障分类功能，能够高效的分析并识别故障类别，有利于实现分布式系统故障运维的智能化，大大降低了分布式系统运维工作难度。

我们的系统做到：

1. 支持注册和登录功能，方便于不同的用户使用
2. 具备模型训练模块，使用了 celery 异步任务，在训练模块中，我们支持上传训练集以及在线训练的功能，同时还提供了将训练进度同步到 web 的功能，在训练结束后还有返回训练模块初始状态的功能。
3. 用表格展示了该用户已经训练好的模型的相关信息，并支持模型下载与删除，模型的详细情况等。
4. 具备模型预测的功能，同时预测的结果会可视化展示在 web 系统中，并支持预测结果的下载。
5. 为了方便管理以及查看用户的相关情况，我们设置了管理员，当使用管理员账号登录时，就会显示登录界面，在登录界面中我们可以查看用户所有的信息以及活跃度，还可以进行封号处理，以及公告的发布。另外我们还设置了超级管理员用来赋予用户管理员权限的。

3.1.2 技术路线

机器学习方面：

我们用了 catboost 模型。CatBoost 是一种梯度提升框架，用于解决分类和回归问题。相对于其他梯度提升框架，CatBoost 能够自动处理分类特征，而无需将其转换为数值形式，这使得模型的训练过程更加简化。

Catboost 模型的特点：

1. 处理分类特征：CatBoost 能够直接处理分类特征，而无需进行特征转换或独热编码。它使用一种基于排序的方法来处理分类变量，有效地捕捉了分类特征的信息。

2. 自动特征缩放: CatBoost 在训练之前会自动对特征进行缩放, 这有助于加快模型的训练速度, 并且使得模型对于特征的变化更加鲁棒。
3. 适应性学习率: CatBoost 引入了一种自适应学习率策略, 它根据每个树的贡献来调整学习率。这种策略能够在训练过程中自动选择最佳的学习率, 并且能够更好地处理梯度爆炸和过拟合问题。
4. 对缺失值的鲁棒性: CatBoost 能够自动处理缺失值, 无需对其进行填充或删除。模型可以利用缺失值所携带的信息, 并在训练过程中进行合理的处理。
5. 防止过拟合: CatBoost 通过引入随机化技术, 如随机排列梯度和随机选择样本, 来降低过拟合的风险。此外, 它还支持常见的正则化技术, 如 L1 和 L2 正则化。
6. 优秀的性能: CatBoost 在性能方面表现出色, 它能够处理高维度的数据集, 并具有高效的训练和预测速度。它还支持多线程训练和 GPU 加速, 进一步提高了训练速度。

Web 方面:

在 web 方面我们前端使用了 html, css, JavaScript, Ajax, axios, 以及 vue 框架, 还用 element-ui 组件库, echarts 图标库来美化界面。后端部分我们使用了 Flask 框架, sqlite 数据库, Redis 消息队列以及 celery 异步任务队列, nginx 负载均衡, waitress 这一个 wsgi 服务器, 大大提高了平台的并发性。

3.1.3 web 系统使用展示

首先被渲染的页面:



这个界面背景采用了一个科技画风的视频，左上角的文字用 JavaScript 实现了逐字显示，下方文字使用了 echarts 图标库以一种特效显示出来，下方按钮链接着登录界面。

登陆界面：

登录

账号

密码

登录

没有账户? [注册](#)

支持登录，并在数据库中查询是否存在该用户，如果存在且信息匹配就可以成功登录。

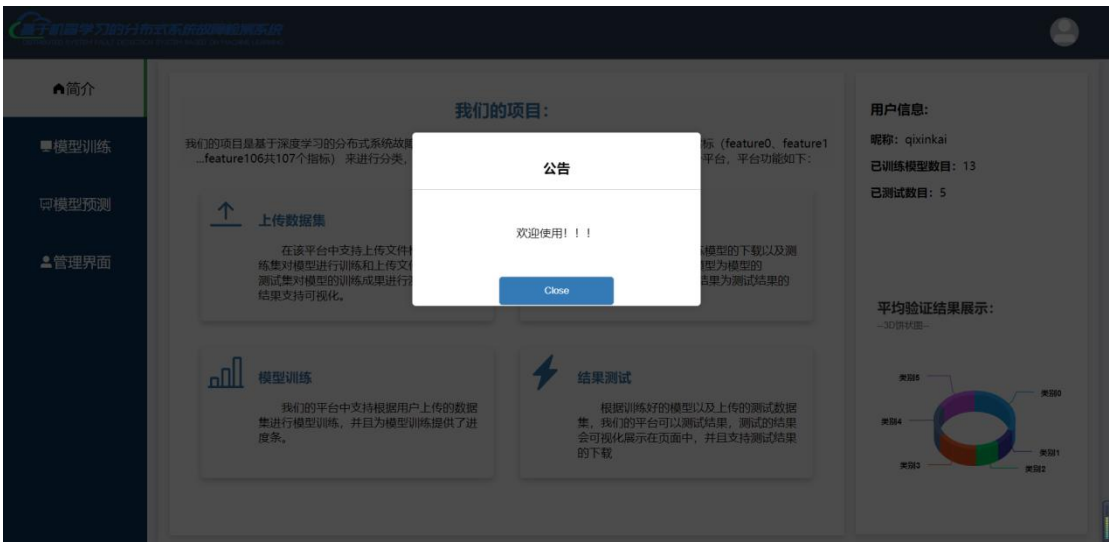
注册界面：



127.0.0.1:5000/login

进行用户信息注册，将注册好的信息存放在数据库 user 表中，其中密码通过哈希加密进行存放，可以保证用户信息的安全性。

公告：



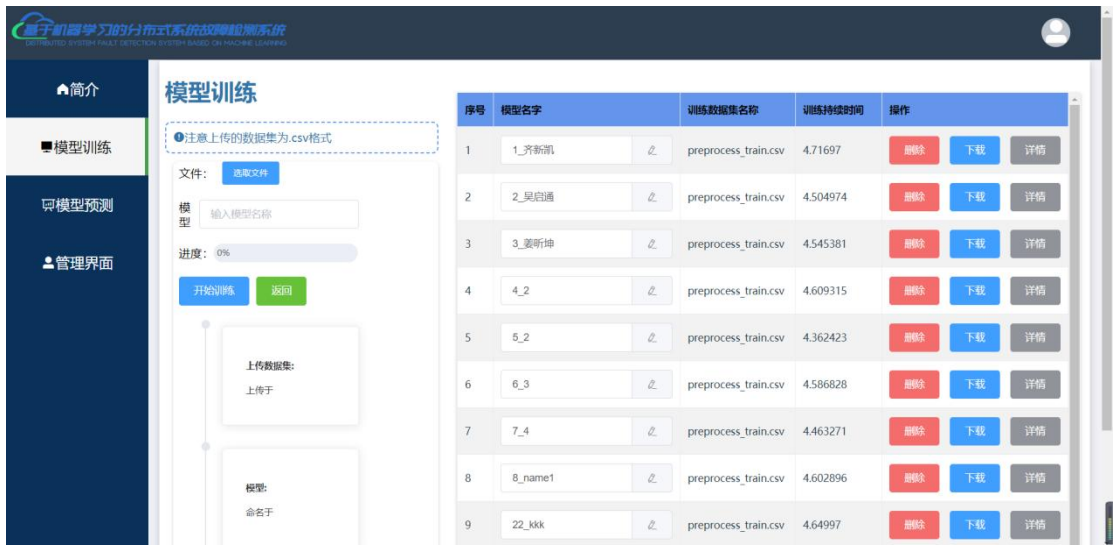
这是我们的公告，只有当管理员发布时才会产生公告。

简介页面：



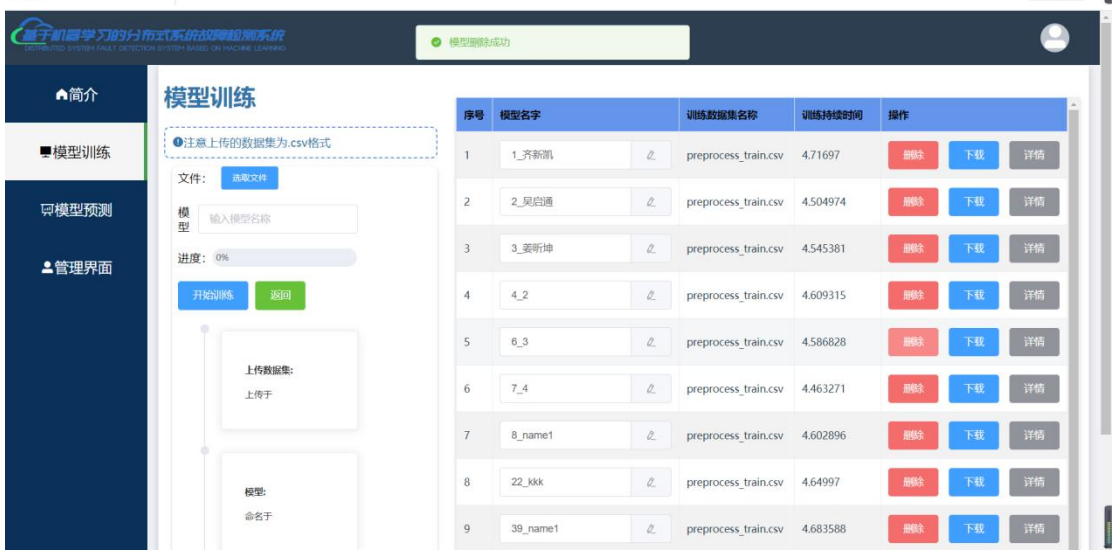
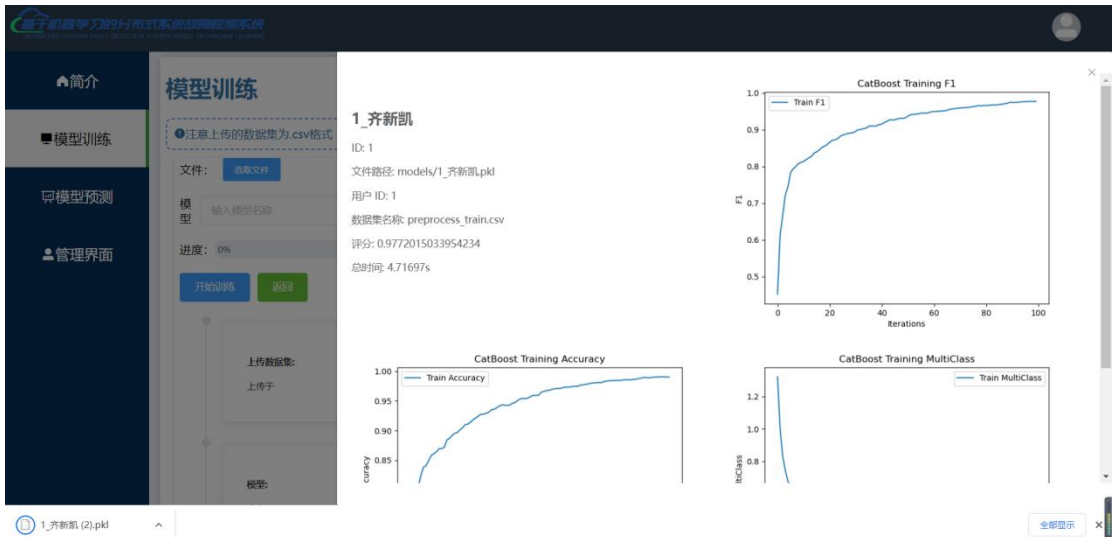
用来展示平台的使用方式, 并且可以显示用户的基本信息以及提供了账户的登出功能, 还提供了用户模型预测结果的平均值展示。

模型训练:



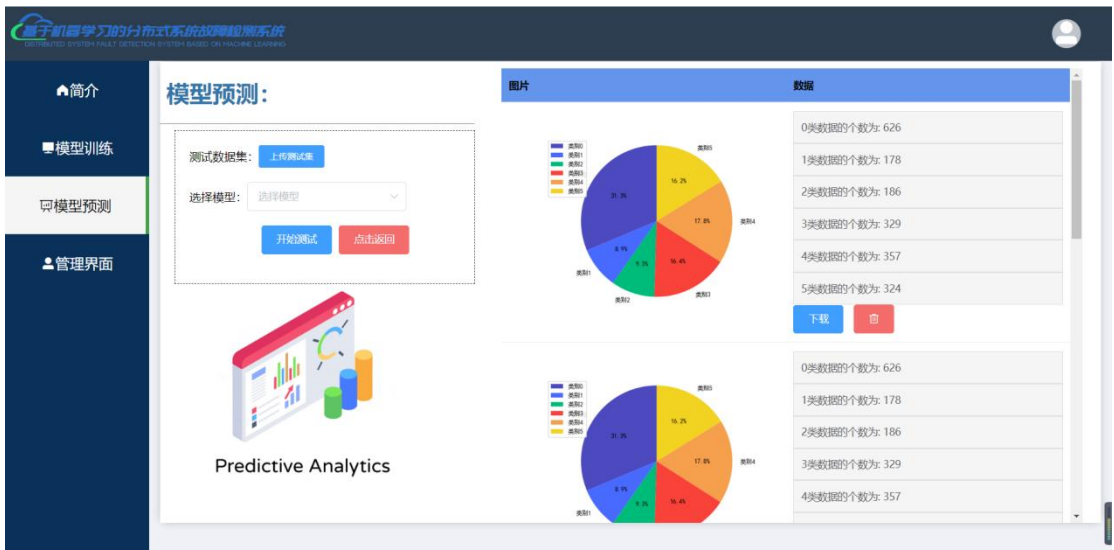
支持在线训练, 以及训练好的模型的展示。当点击返回后, 模型训练模块会回归初始状态。

下方是点开详情, 点开下载之后以及删除模型部分:



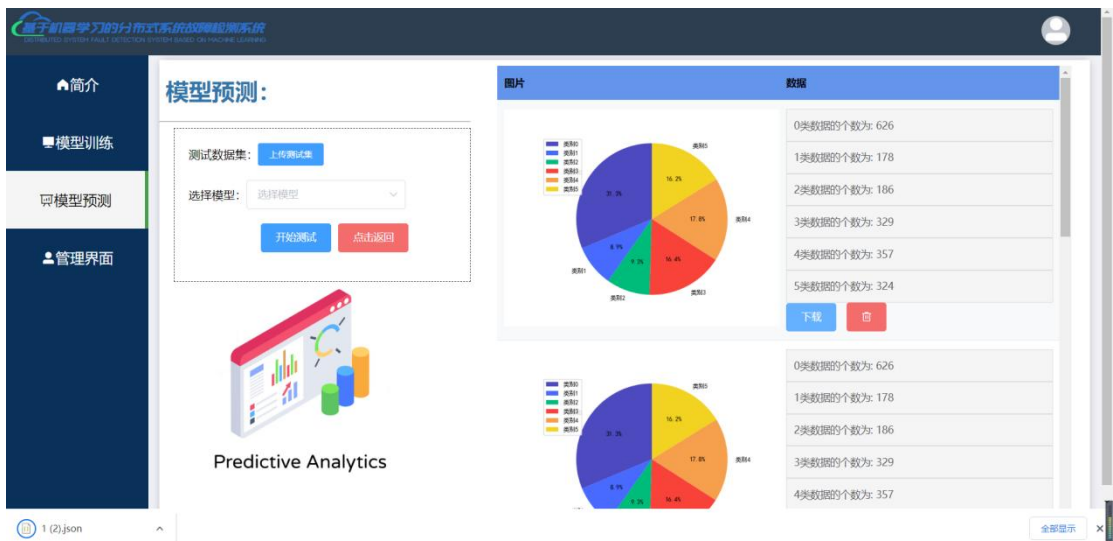
在这里模型被成功删除，并且在数据库 model 表中保存的各种信息都会被删除。

模型预测：



在这一部分里支持用户上传测试集，选择该用户已经训练好的模型进行预测，将预测结果可视化展示在右侧。

下面是测试结果的下载：



在这个图里展示了下载的功能。

用户管理-数据统计：

这里统计了今日的新增用户数，新增训练好的模型数目以及新增的模型预测书，还提供了这些数据的总数据，下方的图标使用了 echarts 图标库来进行绘图，展示了近五日（包括今日的）各项数据。



用户管理-人物管理：

在这里支持管理员进行公告发布以及用户操作，在用户操作中，我们这里展示了封号，详情，关键字查询以及分页的。



详情页面：



这里展示了该用户的登录以及各项最后一次的时间，为了统计用户的活跃度，我们用图像展示了仅七天的登陆数据，当点击某一天的登陆数据时会在下方展示出当天的登录详情。

用户管理-超级管理员：

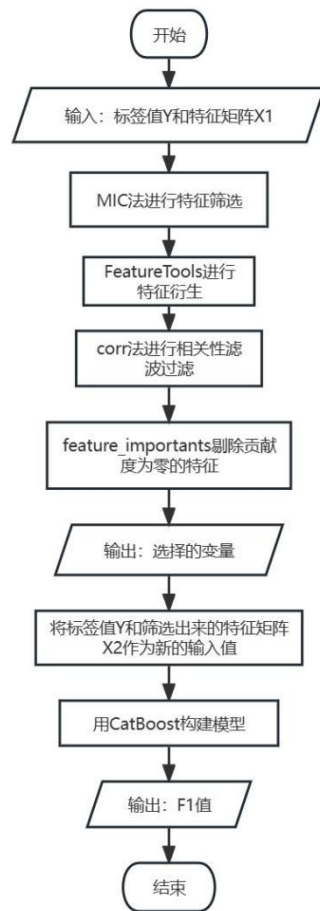
这个地方的主要作用是为用户赋予管理员权限。



3.2 技术方案

3.2.1 模型部分

我们首先借助 MIC 方法选择前 100 的特征，将对模型贡献度为零或为模型带来负面影响的特征去除。再利用 FeatureTools 进行特征衍生，再使用 pandas 库中的 corr 方法进行相关性滤波，过滤线性相关程度过大的值，再借助 sklearn 库中的 VarianceThreshold 方法，对方差阈值小于 0.5 的特征列除去，最后再次使用 CatBoost 模型中的重要性分析，删除对模型贡献度为零的特征，获得最终结果。



3.2.1 web 部分

1、前端方案:

我们采用了 Vue 框架，这是一个轻便且强大的前端框架，用于简化 JavaScript 部分的开发。Vue 框架的双向数据绑定和组件化思想大大提高了我们的开发效率，使我们能够更专注于业务逻辑的实现。

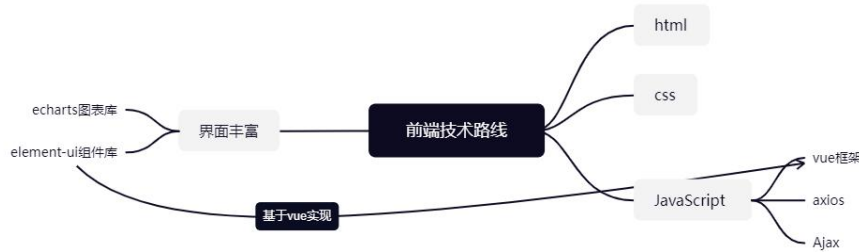
此外，我们还采用了 Ajax 与 axios 来与后端进行数据交互，保证了数据通讯的稳定和高效。axios 作为一个基于 Promise 的 HTTP 库，能够使得我们的异步请求更简洁，也更好地控制了代码的结构，提升了程序的可维护性。

在界面设计上，我们选用了 Element UI 组件库，它提供了一套丰富且高质量的界面元素和组件，我们引用了其中的多个组件，大大简化了开发流程。不仅如此，Element UI 的使用还使得我们的界面更加整洁、规整，对用户的体验产生了积极的影响。

同时，我们还引入了 Echarts 进行图像展示，利用其强大的图表和可视化选

项，使得我们的页面变得更加直观和美观，用户可以轻易理解到更为复杂的数据信息。

在浏览器性能方面，尽管引入了多个库和框架，但是经过我们的优化，这些都不会对性能产生太大的影响。Vue 框架的设计就充分考虑了性能问题，它的轻量级和组件懒加载机制可以显著减少首屏加载时间，提高页面响应速度。而 axios 库的使用，由于其基于 Promise 的特性，可以有效减少阻塞，使得网页在处理数据请求时仍保持流畅。Element UI 和 Echarts 虽然增加了额外的资源请求，但它们的按需引入特性使得我们可以只加载需要的组件和功能，最大程度地减小了对浏览器负担。



2、后端部分：

在后端开发部分，我们选择了 Python Web 的 Flask 框架。Flask 是一种轻量级的 Web 服务器网关接口（WSGI）Web 应用框架，设计简洁且易于扩展，可灵活处理各类 Web 请求，大大提高了开发效率。

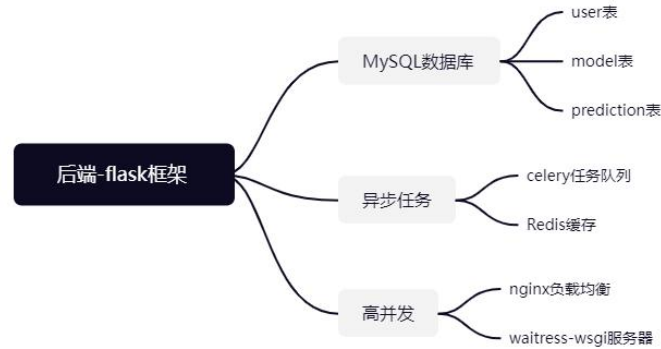
我们使用 sqlite 数据库来存储用户信息、模型的相关信息（包括模型的存储路径等），以及预测结果的相关信息，用户登录信息，公告等。Sqlite 作为一个成熟的关系型数据库管理系统，提供了稳定的数据管理和高效的数据查询能力，对于保证系统的稳定性和响应速度有着重要作用。

我们使用了 Celery 作为任务队列，Redis 作为消息队列，实现了异步任务处理。这种设计可以大大提升系统的并发处理能力，使得在用户数量增多时，我们的服务仍能保持良好的响应速度。Celery 作为一个分布式任务队列，能够有效地对后台任务进行管理和调度，而 Redis 则提供了高速的内存数据存储，让任务的处理和分发变得更为高效。

为了进一步提高系统的性能和稳定性，我们引入了 waitress 这个 WSGI 服务器。相较于传统的 WSGI 服务器，waitress 在处理并发请求时能够更好地利用系

统资源，提高了系统的运行效率。

我们使用了 Nginx 作为负载均衡器，这是一个高性能的 HTTP 和反向代理服务器，在处理 Web 系统高并发请求时，Nginx 可以有效地分发网络流量，确保每个请求都能得到快速响应，大大提高了我们系统的可扩展性和稳定性。



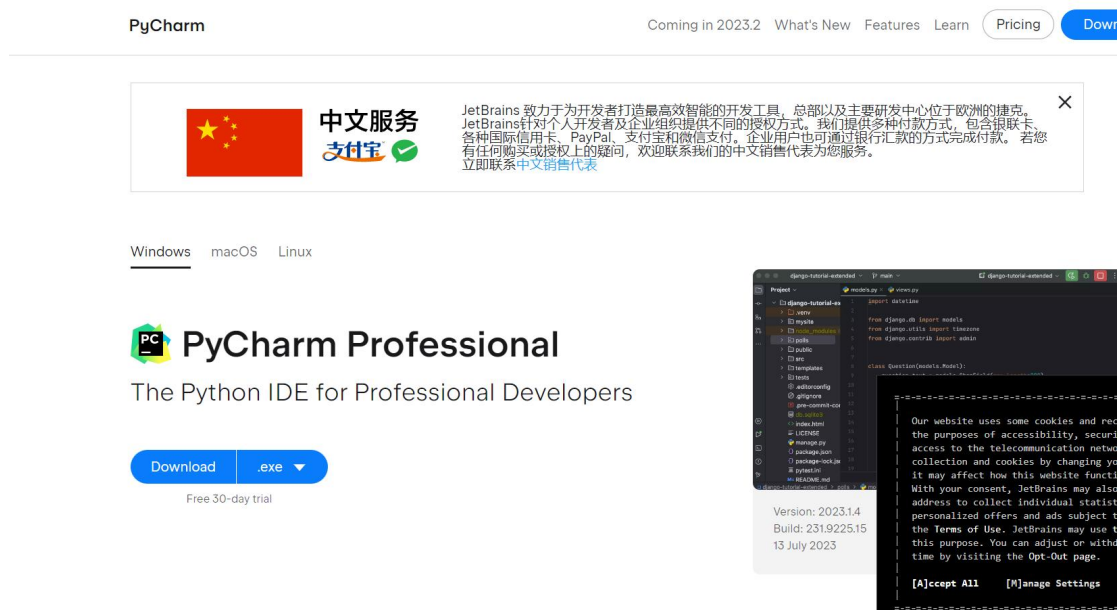
3.3 部署 Web

3.3.1 Python 环境部署

我们的 web 是用 python flask 框架开发的，所以需要安装 python 编译器和配置环境。

下面是 pycharm 的 Windows 的网站：

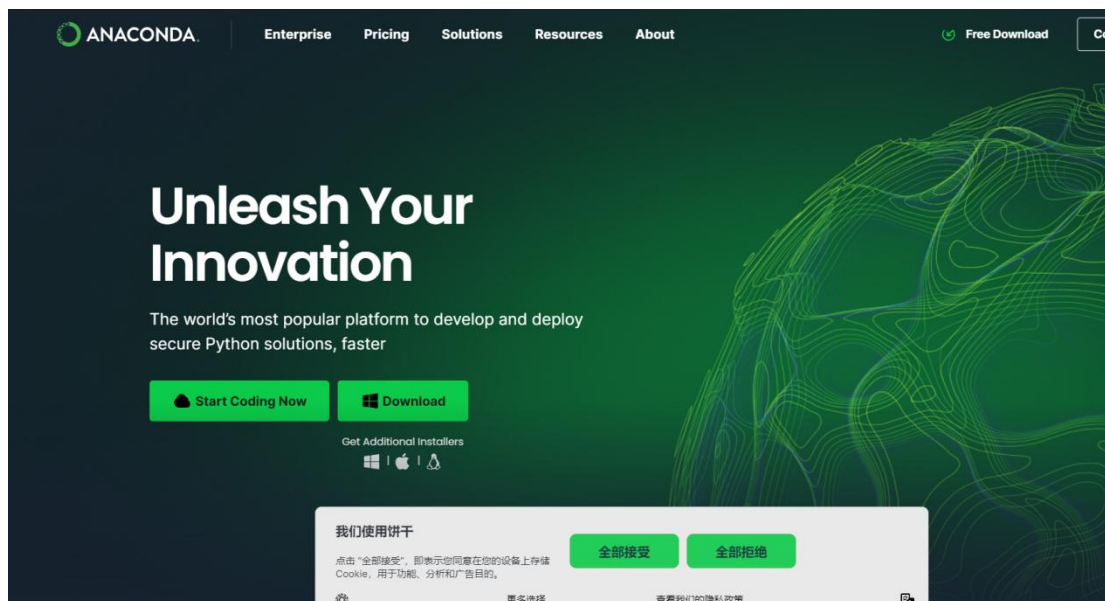
<https://www.jetbrains.com/pycharm/download/?section=windows>



点击直接下载即可。

另外还需要安装一个 anaconda，官网如下：

<https://www.anaconda.com/>版本安装最新版本即可。



安装完成之后，对于 python 的版本。Python 版本使用的是 python3.10，将其配置好即可。

3.3.2 相关环境配置

对于 python 的各种库，我们将其打包放置在 requirement.txt 文件里，当然，如果你想自己下载的话，可以使用 pip 进行下载。对于前端所需要调用的东西。你可以下载 nodejs 然后使用 npm 进行下载，我们这里建议 pip 和 npm 升到最新版本，下载的东西默认最新版本就行。当然，对于前端的各种库，我们的项目里已经打包好了，你不需要下载。在项目里前端 js 的都是完整的。

3.3.3 Windows 版本 Redis 的安装

对于这一个我们建议去 GitHub 上进行下载。网站如下：

<https://github.com/MicrosoftArchive/redis/releases>

下载 3.0.504 即可，另外，建议下载 .msi 格式的，因为这个是可以选择自动配环境，剩下了许多功夫。安装好后端口选择 6379。对于 Redis 的启动，我都放在了各项命令.txt 中。

2016 年 7 月 1 日

思里科焦尔

高-3.0.504

10a978f

比较

3.0.504

最新的

这是针对 Windows 3.0 上的 Redis 的关键错误修复版本。
如果您在集群配置中运行以前版本的 3.0，则应紧急升级到 3.0.504。
该修复解决了集群故障转移过程的问题。

此版本基于 antirez/redis 3.0.5 以及特定于 Windows 的修复。

有关详细信息，请参阅[发行说明](#)。

资产

4

Redis-x64-3.0.504.msi	6.42MB	2016年7月1日
Redis-x64-3.0.504.zip	5.6MB	2016年7月1日
源代码 (压缩)		2016年7月1日
源代码 (tar.gz)		2016年7月1日

223

44

42

99

47

61

已有 342 人回复

3.3.4 负载均衡

安装一下 nginx 的 Windows 版本

下载链接如下：<https://nginx.org/en/download.html>

建议安装稳定版的 1.24.0 版本。

nginx: 下载

主线版本

变化

nginx-1.25.1

pgp

nginx/Windows-1.25.1

pgp

稳定版

变更-1.24

nginx-1.24.0

pgp

nginx/Windows-1.24.0

pgp

旧版本

变更-1.22

nginx-1.22.1

pgp

nginx/Windows-1.22.1

pgp

变更-1.20

nginx-1.20.2

pgp

nginx/Windows-1.20.2

pgp

变更-1.18

nginx-1.18.0

pgp

nginx/Windows-1.18.0

pgp

变更-1.16

nginx-1.16.1

pgp

nginx/Windows-1.16.1

pgp

变更-1.14

nginx-1.14.2

pgp

nginx/Windows-1.14.2

pgp

变更-1.12

nginx-1.12.2

pgp

nginx/Windows-1.12.2

pgp

变更-1.10

nginx-1.10.3

pgp

nginx/Windows-1.10.3

pgp

变更-1.8

nginx-1.8.1

pgp

nginx/Windows-1.8.1

pgp

变更-1.6

nginx-1.6.3

pgp

nginx/Windows-1.6.3

pgp

变更-1.4

nginx-1.4.7

pgp

nginx/Windows-1.4.7

pgp

变更-1.2

nginx-1.2.9

pgp

nginx/Windows-1.2.9

pgp

变更-1.0

nginx-1.0.15

pgp

nginx/Windows-1.0.15

pgp

变化-0.8

nginx-0.8.55

pgp

nginx/Windows-0.8.55

pgp

变化-0.7

nginx-0.7.69

pgp

nginx/Windows-0.7.69

pgp

变化-0.6

nginx-0.6.30

pgp

nginx/Windows-0.6.30

pgp

NGINX

英语

русский

有关下载的

新闻安全文档常见问题解答书籍支持trac twitter 博客单位njs

安装完 nginx 之后，需要做一下配置，具体的配置我们都放在了项目文件里的各项命令.txt 里，你可以查看然后复制到自己的配置中。

下面我们来说一下配置文件在哪：

打开你安装好的 nginx 文件夹：

然后打开 conf 文件夹：

conf	2023/6/4 22:32	文件夹	
contrib	2023/4/11 23:31	文件夹	
docs	2023/4/11 23:31	文件夹	
html	2023/4/11 23:31	文件夹	
logs	2023/6/4 22:32	文件夹	
temp	2023/5/30 15:56	文件夹	
nginx.exe	2023/4/11 23:29	应用程序	3,722 KB

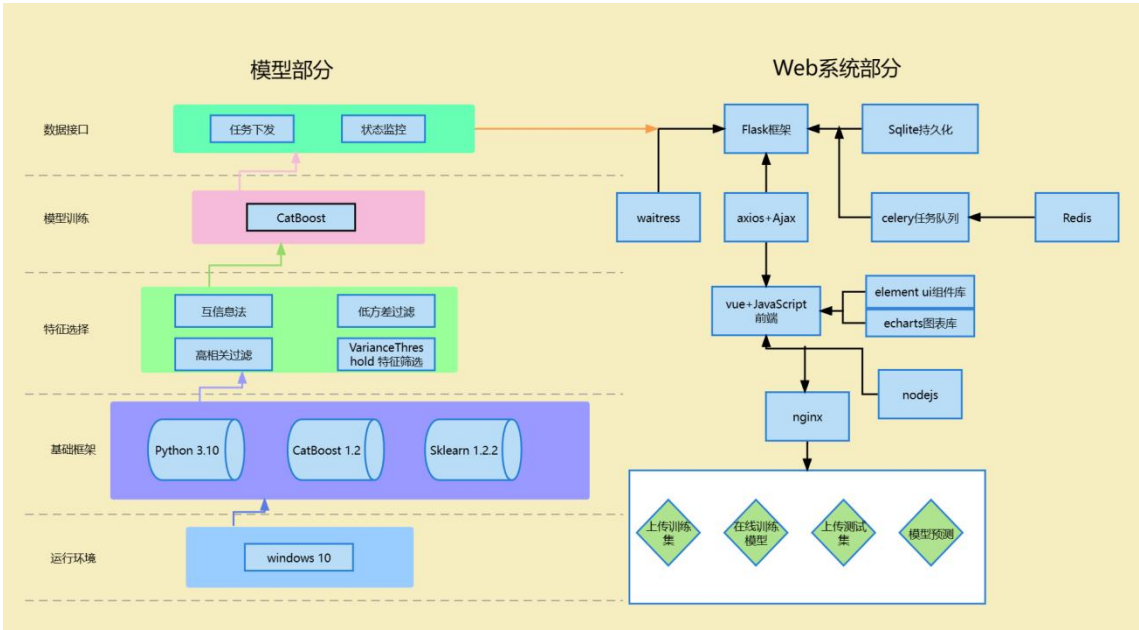
选中其中的 nginx.conf 之后，将配置导入进去即可。

对于 nginx 的启动，我们都放在了各项命令.txt 中。

3.3.5 注意事项

- 1、各项启动命令均在各项命令.txt 里有所展现
- 2、Celery 在 4.0 之后在 Windows 版本运行需要用：`celery -A celery_worker.celery worker -l info -P eventlet` 而不是通用的命令。
- 3、python 版本为 3.10，如果用了其他版本可能会导致兼容性问题。

3.4 平台架构



3.5 总结

在这次软件杯比赛中，我们运用了 catboost 分类模型进行故障的分类，在 web 部分，在前端开发方面，我们使用了 Vue.js 框架进行组件化开发，有效提高了代码复用率和开发效率。使用 Element UI 作为主要的 UI 组件库，使得用户界面的构建更加方便快捷。我们还使用了 Echarts 图表库来实现数据的可视化，这使得我们能够用直观的方式展示和解释模型的预测结果。

在后端部分，我们选择了 Python 的 Flask 框架来构建 API 接口，Flask 轻量且易于扩展，可以灵活地满足我们的需求。数据存储方面，我们使用了 sqlite 数据库，我们在其中存储和管理用户信息、模型，预测结果等数据。

为了异步处理任务和实现消息传递，我们使用了 Celery 作为任务队列和 Redis 作为消息队列。这样可以让应用程序在后台异步处理一些计算密集或时间较长的任务，提高了系统的响应速度和用户体验。

在服务器部分，我们使用了 Waitress 作为 WSGI 服务器，用来托管我们的 Flask 应用。相比于传统的 WSGI 服务器，Waitress 更适合用于生产环境，并且对于各种工作负载都有出色的性能。

我们还使用了 Nginx 进行反向代理和负载均衡，这大大提高了我们应用的可扩展性和可靠性。Nginx 可以有效地管理并发连接，提高服务的并发处理能力，同时也可以多个服务器之间分配流量，提高系统的可用性。

通过这一系列技术的综合运用，我们成功地完成了这次软件杯比赛的任务。

第 4 章 原理分析

4.1 GBDT 算法

GBDT (Gradient Boosting Decision Trees) 是一种集成学习算法，它通过迭代训练一系列的决策树模型，并以梯度下降的方式不断优化模型的预测能力。GBDT 模型所输出的结果是由其包含的若干棵决策树累加而成，每一棵决策树都是对之前决策树组合预测残差的拟合，是对之前模型结果的一种“修正”。梯度提升树既可以用于回归问题（此时被称为 CART 回归树），也可以被用于解决分类问题（此时被称为分类树）。

GBDT 的原理可以概括为以下几个步骤：

初始化：将训练数据的目标值作为初始预测值，通常使用平均值作为初始预测。

迭代训练决策树：GBDT 通过迭代训练多棵决策树，每一棵树都试图纠正前面所有树的预测误差。每棵树的训练过程包括以下几个步骤：

- a. 计算残差：计算当前模型的预测值与真实值之间的残差，即目标值与当前预测值的差异。
- b. 构建决策树：使用训练数据的特征和残差作为输入，构建一棵决策树模型。决策树的构建过程通常采用贪婪算法，通过选择最佳的特征和切分点来最小化残差。
- c. 更新预测值：使用当前决策树的预测结果与之前的预测结果相加，得到更新后的预测值。

迭代更新：重复步骤 2，训练更多的决策树并更新预测值，直到达到设定的迭代次数或预测误差满足要求。如图 1.1 所示

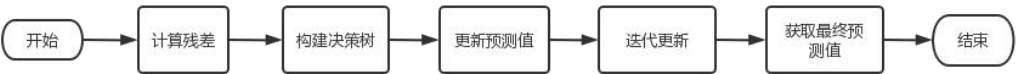


图 4.1 GBDT 训练过程

得到最终预测值：将所有决策树的预测结果相加，得到最终的预测值。

GBDT 的优点在于能够处理各种类型的特征（包括连续型和离散型），并且具有较强的预测能力。然而，GBDT 也有一些需要注意的地方，比如对于高维稀疏数据的处理较为困难，模型的训练过程是串行的，无法并行化处理。此外，GBDT 容易过拟合训练数据，需要通过合适的正则化方法进行控制。

该算法被广泛应用在，具有大量噪音数据和复杂依赖关系等特点的大数据任务场景，比如网络搜索、推荐系统、金融风控和医疗诊断等各个领域，并且相比其他机器学习算法取得了非常优异的效果。GBDT 算法是基于集成技术的一系列变体，它通过选择能够使损失函数 $L(\hat{y}, y)$ 达到最小化的值，从而迭代地生成一组决策树，以分类任务为例，各决策树叶节点中所产生的类别概率之和，就是集合的输出概率，最后共同作用完成决策任务

4.2 CatBoost 算法

为了解决无法直接处理分类变量、模型容易过拟合等现有机器学习模型存在的突出问题，基于平衡树构造的 GBDT 算法 CatBoost 于 2018 年被提出。该算法因其独特优势，包括能管理分类变量、自带处理过拟合的机制，并解决梯度偏差以及预测偏移等问题，各行业研究人员成功地将 CatBoost 用于涉及不同领域数据的机器学习研究。

CatBoost 被应用于不同领域数据研究主要有四点优势，如图 1.2 所示。

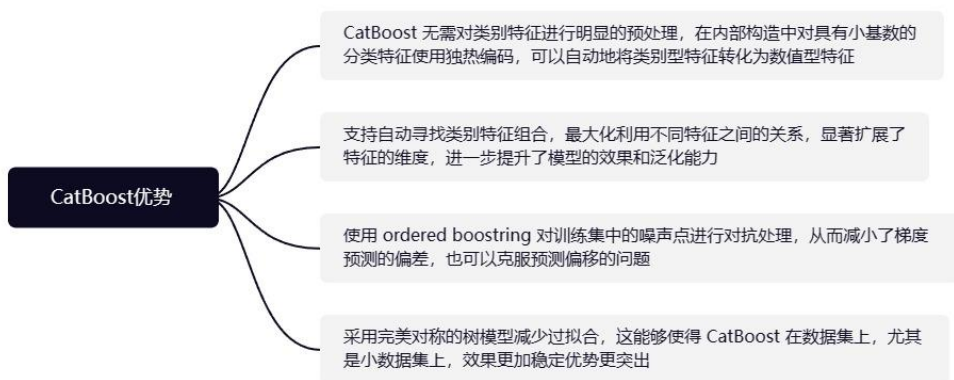


图 4.2 CatBoost 优势

这些优势在算法原理上主要体现为，运用目标变量的统计策略，特征组合的贪心策略以及解决预测漂移问题的策略这三点改进。

(1) 目标变量的统计策略

类别特征在数据分析中往往属于非常难以处理的一类特征，处理的好坏会对模型的效果影响巨大，较为常见的处理类别特征方法是，直接使用类别对应目标变量的均值来替换原先的类别值。但是经常会出现以下几个算法缺陷：

- 当类别特征与标签相关性并不是非常强的时候，模型会非常容易过拟合；
- b. 当类别特征的基数较小的时候，统计的均值噪音信息较大；
- c. 每个类别含有更多的信息，直接取均值的话会丢失非常多的信息；

CatBoost 算法在传统 GBDT 模型基础上改进，采用独热编码方式，将类别特征进行预处理，转化为数值型的特征，但在这个过程中很容易出现转换后特征维度迅速膨胀等问题。为了解决这一问题，进而提出了目标变量的统计策略(Target Statistics)，加入了平滑先验 p ，使用目标变量估算每个类别的期望值。

$$\bar{y}_i = \frac{\sum_{j=1}^n 1_{\{x_{j,k}=x_{i,k}\}} * y_i + ap}{\sum_{j=1}^n 1_{\{x_{j,k}=x_{i,k}\}} + a} \quad (4-1)$$

其中 a ($a>0$) 是一个参数, 平滑先验 p 是数据集合中的目标均值。经过改进之后, 可以缓解类别频次较小样本均值带来的不可靠信息, 但是模型过拟合, 容易产生目标变量泄露。训练数据集和测试集中的均值条件分布差异大, 若直接取均值会出现丢失每个类别中大量信息等问题。

有序的目标变量统计是 CatBoost 中为了解决这些问题, 提出的一种高效策略。它的准则依赖于排序, 每个样本的目标变量统计值仅依赖于历史的观察。为了使这个想法适应标准的离线设置, 引入了一个人工“时间”, 也就是对训练样本进行随机扰动, 对于每个样本, 我们使用所有的“历史”观察来计算目标变量统计值, $D_k = D$ 用于测试, 即:

$$D_k = \{x_j: \sigma(j) < \sigma(k)\}$$

$$\bar{y}_i = \frac{\sum_{x_j \in D_k} 1_{\{x_{j,k}=x_{i,k}\}} * y_i + ap}{\sum_{x_j \in D_k} 1_{\{x_{j,k}=x_{i,k}\}} + a} \quad (4-2)$$

CatBoost 对梯度提升的不同步骤使用的排列方式不同。

(2) 特征组合的贪心策略

两个单独的特征很难体现出共同的交叉关系, 但是若两者进行组合, 往往可以因为交互作用, 产生出新的影响因素, CatBoost 模型捕捉到了这一点, 实现了类别特征的自动化组合, 并希望借此更好地捕获数据中的高阶数据信息。但随机的特征组合会随着类别特征个数的变多而指数级的增加, 我们也很难对所有的数据进行组合, 因此采用贪婪算法的策略。

CatBoost 在每次分割时会进行贪心的特征组合, 我们在选择第一个节点时, 会优先考虑单个节点, 例如 f_1 。在生成第二个节点时, 还会考虑 f_1 和其它类别变量的组合, 然后选择其中最好的, 即贪心地进行类别特征的交叉组合, 考虑了变量之间交互作用的影响, 可以额外获得更多的特征信息, 在一定程度上提高模型精度。

(3) 预测漂移问题的策略

在之前的梯度提升算法模型中未被发现和解决的一个问题便是预测漂移。在以往的 GBDT 模型推导时，我们在第 t 轮的目的是通过学习一个新的树模型，尽可能最小化下面的式子：

$$h^t = \arg \min_{h \in H} EL(y, F^{t-1}(x) + h(x)) \quad (4-3)$$

这个时候我们一般会选择用下面的式子通过负梯度对其进行近似拟合：

$$h^t = \arg \min_{h \in H} E(-g^t(x, y) - h(x))^2 \quad (4-4)$$

最后将学习得到新的学习器拼接上来得到 $F^t(x) = F^{t-1}(x) + h^t$ 。然而在实践中，我们经常会选择使用数据集 $D(x_k)$ 来近似优化：

$$h^t = \arg \min_{h \in H} \frac{1}{n} \sum_{k=1}^n (-g^t(x_k, y_k) - h(x_k))^2 \quad (4-5)$$

考虑到数据集 $D(x_k)$ 的随机性，关于训练样本 x_k 的条件分布 $F^{t-1}(x_k) | x_k$ ，和测试样本 x 的条件分布 $F^{t-1}(x) | x$ 相比发生了偏移，那么上面式子 (1-4) 定义的基预测器 h^t 和下面的式子 (1-5) 是存在偏差的，这种偏移产生的误差最终会影响到训练模型的泛化能力。

为解决上述所涉及的预测偏移问题，CatBoost 设计了有序增强算法，对应算法流程如下：

- a. 首先随机生成一个 $1, 2, \dots, n$ 的排列；
- b. 使用前面 i 个样本得到学习模型，维护 n 个不同的模型 M_1, M_2, \dots, M_n
- c. 进行多次迭代得到样本 j 的残差近似值，使用模型 M_{j-1} 进行估计。

尽管从算法来看 CatBoost 能够把传统的 GBDT 模型转化为有序的提升算法，解决模型在迭代过程中出现的的梯度偏移问题，但我们发现构建这一过程需要 n 个模型，导致时间及空间复杂度显著升高，因此实际应用中，CatBoost 建模时通常选用对称树(oblivious decision trees)作为基预测器。选用相同的树模型作为拆分标准，构建平衡树，减少过度拟合，加快预测时的执行速度。

4.3 特征工程-CatBoost 算法

4.3.1 过滤式特征选择

过滤式特征选择是特征选择的一个较为传统的实现路线，顾名思义，过滤式特征选择是指在将数据集应用于机器学习过程中之前，通过某种准则将原始特征集进行过滤，过滤掉无关和冗余特征。过滤式特征选择是一个常规和直观的路线，但是由于其所采用的过滤式准则往往基于数学和统计学原理，因此其在理论上更具有可解释性和确定性。更重要的是，过滤式特征选择的步骤相对其他特征选择路线复杂性低，因此其效率也更高。

通常情况下，过滤式特征选择分为三大阶段。第一阶段即选定过滤式准则，由于过滤式特征选择是按照过滤式准则对特征进行筛选的，因此这一阶段至关重要。当前的过滤式特征选择算法研究的重点也是这一阶段；第二阶段即根据过滤式准则，以某种搜索策略（通常是贪婪式搜索策略）对原始特征子集进行过滤；第三阶段即验证特征选择的效果，该阶段一般会涉及到大量实验。

4.3.2 互信息

互信息(Mutual Information)作为信息理论中一种经典的信息度量，它可以表征一个随机变量由于已知另一个随机变量而减少的不确定性，也即两个变量共有的信息。本作品首先利用互信息挖掘变量之间线性和非线性的关系，进而评价两个变量的统计相关性。下面分析互信息的相关理论。

1、信息熵

1948 年 C. E. Shannon（香农）借用热力学中熵的概念提出了信息熵的概念，热力学中的熵用于度量一个系统的混乱程度。与此类似，信息熵描述的是信源的不确定性，是信源中所有目标的平均信息量，信息熵是信息理论中用于度量信息量的一个概念。

对于离散型随机变量 X ，其信息熵可用如下公式计算：

$$H(X) = -\sum_{x \in \Omega_X} p(x) \log_2(p(x)) \quad (4-6)$$

其中 Ω_x 代表变量 X 的取值空间， $p(x)$ 代表 $X=x$ 的概率。

变量 X 的信息熵描述了变量 X 的不确定性，变量不确定性越大，其信息熵就越大，这也意味着变量 X 隐含的信息越多。

2、联合熵

在信息熵的基础上，联合熵用于度量一个联合分布的随机变量系统的不确定性。两个离散型随机变量 X 和 Y 的联合信息熵定义为：

$$H(X,Y) = -\sum_{x \in \Omega_X} \sum_{y \in \Omega_Y} P(x,y) \log_2[P(x,y)] \quad (4-7)$$

其中 Ω_X 、 Ω_Y 分别代表变量 X 、 Y 的取值空间， $p(x,y)$ 代表 $X=x$ 和 $Y=y$ 的联合概率。

3、条件熵

条件熵 $H(Y|X)$ 用于度量在已知随机变量 X 的条件下随机变量 Y 的不确定性。具体来说，其定义如下：

$$H(Y|X) = -\sum_{x,y} p(x,y) \log_2(p(x|y)) \quad (4-8)$$

其中， $p(x|y)$ 是条件概率，代表 $Y=y$ 的条件下， $X=x$ 的概率； $p(x,y)$ 代表 $X=x$ 且 $Y=y$ 的联合概率。

关于熵、联合熵、条件熵、互信息相互间的关系，图 1.3 以韦恩图的形式予以了说明。图 1.3 中，整个蓝色集合代表变量 X 的熵；整个黄色集合代表变量 Y 的熵。两个集合的交集即 0 处表示互信息 $I(X;Y)$ ；蓝色集合中的 1 代表条件熵 $H(X|Y)$ ；黄色集合中的 1 代表条件熵 $H(Y|X)$ 。两个集合的并集代表变量 X 和 Y 的联合信息熵 $H(X,Y)$ 。

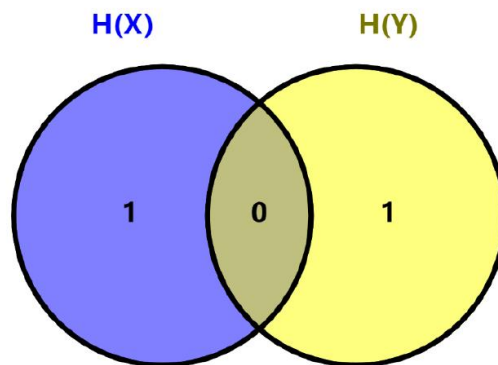


图 4.3 熵、联合熵、条件熵、互信息的关系图

4.3.3 高相关过滤、低方差过滤

低方差过滤 (Low Variance Filtering)：低方差过滤在降维操作中常用于选择具有高方差（变化大）的特征，同时过滤掉具有低方差（变化小）的特征。这是因为低方差的特征对于数据的区分性和信息量贡献较小，保留它们在降维后可能会导致较差的结果。

具体步骤如下：

- a. 计算每个特征的方差。
- b. 根据预设的阈值或排序，选择具有高方差的特征。
- c. 去除低方差的特征，只保留选择的高方差特征。
- d. 低方差过滤有助于减少数据中的噪声或冗余信息，提高降维后数据的质量。

高相关过滤 (High Correlation Filtering)：高相关过滤在降维操作中常用于选择具有较高相关性的特征，将高度相关的特征合并或删除。这是因为高度相关的特征提供的信息重复或冗余，保留它们在降维后可能会浪费存储空间和计算资源。

具体步骤如下：

- a. 计算特征之间的相关性（如相关系数）。
- b. 根据预设的阈值或排序，选择具有较低相关性的特征。
- c. 去除高度相关的特征，只保留选择的低相关性特征。
- d. 高相关过滤有助于减少冗余信息，提高降维后数据的效率和解释性。

本作品主要使用以上特征选择算法对数据进行处理。

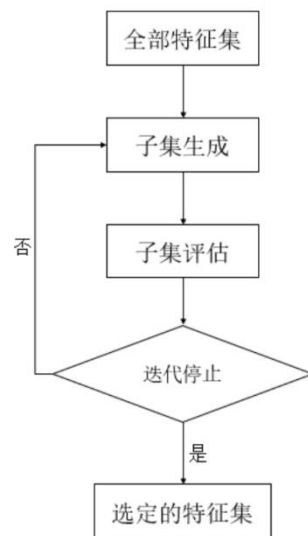


图 4.4 特征选择总体流程

4.3.4 创新点

本文创新点首先借助 MIC 方法选择前 100 的特征，将对模型贡献度为零或为模型带来负面影响的特征去除。再使用 pandas 库中的 corr 方法进行相关性滤波，过滤线性相关程度大于 0.5 的值，再借助 sklearn 库中的 VarianceThreshold 方法，对方差阈值小于 0.5 的特征列除去，最后再次使用 CatBoost 模型中的重要性分析，删除对模型贡献度为零的特征，获得最终结果。

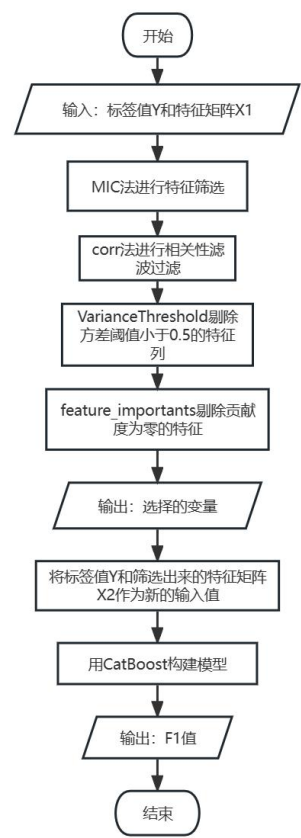


图 4.5 特征工程-CatBoost 算法流程图

第 5 章 结果分析

5.1 验证集结果

	precision	recall	f1-score	support
0	0.81	0.97	0.88	176
1	0.87	0.84	0.85	166
2	0.86	0.73	0.79	171
3	0.99	0.95	0.97	169
4	0.93	0.99	0.96	156
5	1.00	0.95	0.97	162
accuracy			0.91	1000
macro avg	0.91	0.91	0.91	1000
weighted avg	0.91	0.91	0.90	1000

The MacroF1 of the valid is:90.80%

验证集结果如图所示，MacroF1 达到 90.80%。

5.2 测试集结果

```
类别0的数量: 438
类别1的数量: 111
类别2的数量: 490
类别3的数量: 294
类别4的数量: 349
类别5的数量: 318
数据总数: 2000
```

测试集结果：各类别预测数量如图所示。

第 6 章 重难点突破及未来展望

6.1 重难点突破

本项目前端可以正常的获取后端的进度，并且在 JavaScript 里进度输出正常，但是在显示时，进度条却是只跳转两次，第一次是跳转到 1%-99% 中的某一个，第二次跳转是跳转到 100%，起初对于这个问题我们以为是网络影响，经过查阅资料、相互讨论，发现是 vue 的 dom 更新不及时导致的，因此我们使用了 vue 框架的 `this.$nextTick()` 函数进行了强制更新解决了这个问题。

6.2 未来展望

随着技术的不断发展，本项目的分布式故障诊断系统将会融合更多先进的机器学习、深度学习和人工智能技术。例如，结合强化学习、迁移学习等方法，系统将变得更加智能和自适应，能够应对更多不同类型设备和工业场景。

另外，随着大数据的发展，工业生产中的数据将会更加多样化，包括图像、声音、文本等。本项目可以进一步探索如何有效整合这些多模态数据，提供更全面的故障诊断和预测能力。同时，随着产业物联网和智能制造的兴起，未来的分布式故障诊断系统不仅仅限于故障诊断，还要向更高级别的智能决策支持发展。例如，根据故障预测结果，系统可以提供优化维护计划、资源分配等建议，为工业企业带来更高的效益和竞争力。

除此工业生产的应用，本项目还可以在医疗保健中处理疾病诊断、药物响应预测、医疗图像分类等。在金融行业中，本项目可用于信用风险评估、欺诈检测、客户细分、信用评分等任务。