**摘 要**

在互联网大数据技术越发成熟的今天，越来越多的行业对数据挖掘与数据分析的需求日益增多。在实际应用过程中，数据挖掘技术的流程以及常用手段都能通过固定的编程思路进行编码实现。基于此，本文在自动代码生成技术模版的基础上，通过对数据挖掘过程中主要步骤——数据清洗，机器学习数据建模等方法的编程思路的研究，从而抽象出主要代码逻辑，以及关键参数，使得用户能通过自定义配置主要参数最终实现自动生成数据挖掘代码用以分析实际数据的功能。本代码生成器的实际意义主要在于在生成代码的基础上，构建运行引擎，实时展示生成代码的运行结果，达到系统闭环，以完成低代码，高效率，高可用的数据挖掘代码构建。

**关键词**：数据挖掘，代码生成，低代码, 数据分析，大数据，机器学习

# ABSTRACT

With the increasing maturity of data analytics and big data technologies, more and more industries have an increasing demand for data mining. In the actual application process, the process of data mining technology as well as the common means can be coded and implemented by fixed programming ideas. Based on this, this paper examines the programming ideas of the methods used in the main steps of the data mining process - data cleaning and data modeling - based on the automatic template-based code generation techniques. Thus, the main code logic and key parameters are abstracted to enable users to customize the main parameters to automatically generate data mining code for analyzing actual data. Based on the generated code, a runtime engine is built to display the runtime results of the generated code in real time to achieve a closed-loop system, so as to complete the construction of data mining code with low code, high efficiency and high availability.

**Keywords**: data mining, code generation, low code, data analysis, big data

# 目录

[ABSTRACT 2](#_Toc72615876)

[目录 3](#_Toc72615877)

[1 绪论 1](#_Toc72615878)

[1.1 数据挖掘代码生成器的背景和意义 1](#_Toc72615879)

[1.2 自动代码生成技术的发展状况 1](#_Toc72615880)

[1.3 数据挖掘代码生成器的研究内容 2](#_Toc72615881)

[1.4 系统功能简述 3](#_Toc72615882)

[2 数据挖掘代码生成系统分析 4](#_Toc72615883)

[2.1 整体架构分析 4](#_Toc72615884)

[2.1.1 系统架构图 4](#_Toc72615885)

[2.2 系统需求分析 4](#_Toc72615886)

[2.2.1 需求分析概述 5](#_Toc72615887)

[2.2.2 代码生成系统流程图： 5](#_Toc72615888)

[2.3 系统安全分析 6](#_Toc72615889)

[2.3.1 sql注入防御 6](#_Toc72615890)

[2.3.2 路径穿越防御 6](#_Toc72615891)

[2.3.3 恶意注册防御 7](#_Toc72615892)

[2.4 本章小结 7](#_Toc72615893)

[3 数据库分析与设计-MongoDB 7](#_Toc72615894)

[3.1 后台数据库以及MongoDB概述 7](#_Toc72615895)

[3.1.1 MongoDB数据库介绍 7](#_Toc72615896)

[3.1.2 选择MongoDB数据库分析 8](#_Toc72615897)

[3.2 数据集合的具体设计 8](#_Toc72615898)

[3.2.1 数据库集合设计展示 8](#_Toc72615899)

[3.3 本章小结 10](#_Toc72615900)

[4 数据挖掘代码生成器模板代码设计 10](#_Toc72615901)

[4.1 数据清洗代码生成器 10](#_Toc72615902)

[4.1.1 常见数据清洗方法简介 11](#_Toc72615903)

[4.1.2 数据清洗代码模板设计 12](#_Toc72615904)

[4.1.3 数据清洗代码生成交互 12](#_Toc72615905)

[4.2 数据建模（机器学习）代码生成器 13](#_Toc72615906)

[4.2.1 有监督与无监督学习算法 14](#_Toc72615907)

[4.2.2 模型评估方法简介 15](#_Toc72615908)

[4.2.3 数据建模机器学习方法优劣 19](#_Toc72615909)

[4.2.4 数据建模（机器学习）代码生成模板设计 26](#_Toc72615910)

[4.3 本章小结 31](#_Toc72615911)

[5 数据分析系统引擎设计 31](#_Toc72615912)

[5.1 一键生成探索性分析报告 31](#_Toc72615913)

[5.2 数据清洗引擎设计 33](#_Toc72615914)

[5.3 数据建模（机器学习）引擎设计 35](#_Toc72615915)

[5.4 常用模型参数模板化/推荐 36](#_Toc72615916)

[5.5 本章小结 37](#_Toc72615917)

[6 参考文献 38](#_Toc72615918)

[致 谢 40](#_Toc72615919)

# 绪论

## 数据挖掘代码生成器的背景和意义

回顾近年来的发展，数据挖掘这项技术的应用不仅是在信息行业激起了广泛的关注，而且在其他行业，特别是在企业级数据库行业也得到了极大的青睐。

随着互联网各种的技术的成熟以及应用的落地，数据量的与日俱增，不同行业对数据分析的需求也因此产生了巨大的增长。此外，在目前的各种互联网产品中，后台存储的大量数据拥有着巨大的挖掘价值，随着机器学习技术在各领域的广泛应用，使得对大批量数据的建模分析变得容易。

然而，在企业实际运作中，由于不同数据源产生的数据往往有着不同字段，以及不同的预处理逻辑，使得面对不同的数据集时需要针对特定场景进行代码的重新编写。鉴于数据挖掘流程有着明确的处理流程，高可复用的逻辑，即数据探索、数据清洗、数据建模，模型评估等步骤。本文着眼于将数据挖掘流程模板化，通过用户在界面选择对应方法，个性化定制，生成以用户上传数据集为分析目标的python数据挖掘代码，提高数据分析流程的效率。

## 自动代码生成技术的发展状况

代码自动生成技术，代码生成器的相关研究从20世纪末开始发展至今,已经趋于平稳和成熟。通过对谷歌学术Code Generator(代码生成器)的检索，绘制出最近十年关于代码生成器的相关文献，专利发表情况。如图1所示(下图统计图中横轴表示时间，纵轴表示相关内容的发表数量,文中的数据统计于谷歌学术)

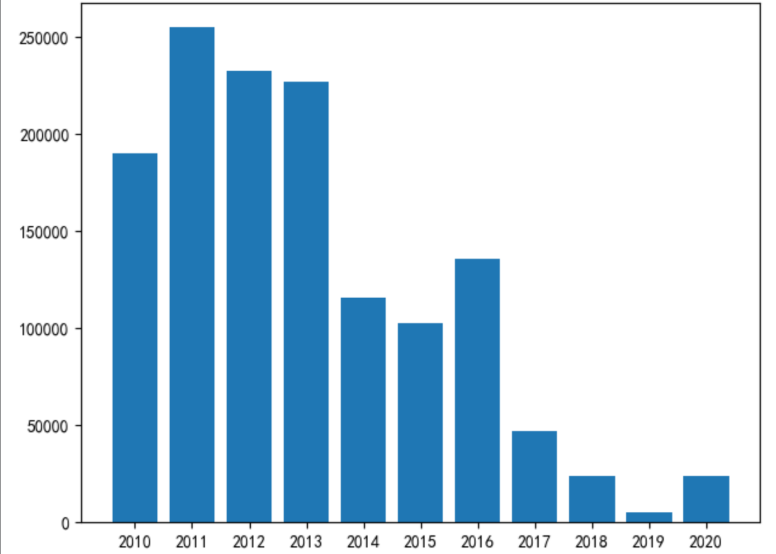


图1代码自动生成技术近十年趋势图

随着研究的深入，与“代码生成”相关的研究点越来越多，形成了一个庞大的研究网络，该网络包括：软件开发，编译器，模型驱动，嵌入式系统，设计模式，编译器等知识领域。国内软件公司在自动代码生成和模型驱动体系结构的研究和开发中也取得了许多成就。例如，金蝶软件公司是第一个支持模型驱动的体系结构软件开发的商业工程工具，金蝶eas4.0，初凡科技是​​基于“模型驱动的体系结构”并由“商业模型驱动的全自动软件工厂”开发的企业应用平台。实现代码与模型之间的迭代转换[1]

国内许多大学和研究机构也对代码生成进行了长期的研究，特别是基于模型驱动骨架的代码生成软件工程，并取得了很多成果。 以下给出了中国自动代码生成研究中的典型问题，包括基于模型转换的自动代码生成,代码的生成和对象关系映射,基于模型的用户界面自动代码生成的技术,逆向工程和应用软件流程等方面的相关文献。图表用来自动建立和生成网络源代码,提出了基于网络源代码生成的算法,在平台无关模型到与平台相关联的网络源代码模型中,网络用户接口建模和自动源代码的生成,用户接口源代码的自动生成等。

## 数据挖掘代码生成器的研究内容

本文主要研究通过Python实现基于Web端的数据挖掘代码生成器。代码的生成主要采用基于基于模板的代码生成方法[2]。关于基于模板的代码生成，该项技术的原理主要是为了将功能的需求进行拆解，从而拆分成两个部分。第一部分是相对来说固定不变的部分,即静态部分，来自后台系统中的模板文件；另一部分是根据外部输入变化的部分，即来自前端用户的输入（如选择哪些列做数据预处理，选择什么模型构建数据等等），如图2所示。



图2基于模板的自动代码生成

本系统基于数据挖掘的流程，通过预先设置好数据预处理、数据可视化、数据建模对应的模板文件。经过代码生成引擎调用，结合前端数据的输入，动态生成可运行的Python脚本，同时运行脚本生成可视化报告，用于前端展示。

为了能够取得更直观的界面以及更佳的用户体验，本文采用Django + MongoDB + Vue的架构实现代码生成器的系统功能。

用户通过上传数据集，系统读取数据存入MongoDB，前端可展示该用户所有数据集，通过预览按钮解析上传文件并做对应前端展示。数据展示端提供一键生成数据分析报告功能，以交互式网页展示所选数据集每一列的类型、缺失值、异常值、强相关变量，并以高亮的形式展现，用以提示用户，作为数据预处理的参考。

## 系统功能简述

本系统以数据驱动。通过保存用户上传的数据集到系统中用户的数据集管理界面，以供用户选择需要分析的数据集，并进行数据挖掘相关操作。

①、数据预览。数据预览将根据一个数据集以表格的形式展示,并且支持根据一个数据集具体内容自适应,从而制作出探索性数据分析报告。数据分析研究报告是针对大量数据整体进行的探究和分析，将数据集中存在缺失值、异常值、高相关度的关键信息予以高亮展示。同时针对数据集各列的维度进行详细分析以及分布特征的可视化图表。数据分析报告给出的高亮信息用以为数据清洗提供参考方向

②、数据清洗。用户在已上传数据集列表中，选择需要分析的数据集，自定义选择不同的列，在提供的数据清洗方法中自由组合不同的数据清洗规则以达到清洗“脏”数据的目的。系统支持的数据清洗规则主要包括缺失值填充、排序、模糊匹配、数据标准化等，允许用户自定义调换规则顺序，后台引擎根据用户输入的规则进行校验，并展示清洗后的数据表格。

③、数据建模。用户选择指定数据集，选择数据集中的若干列作为数据特征，选择一列作为目标列。提供分类、回归、聚类问题的若干种训练模型，以及对应的模型评估方法，模型评价结果以可视化图表的形式展示。参数选择完毕通过前端界面弹窗显示生成的代码。

# 数据挖掘代码生成系统分析

## 整体架构分析

### 系统架构图

本系统基于Python的Web框架django运行，前端使用vue框架以达到前后端分离的效果。

图形用户界面

描述已自动生成

图2-1 数据挖掘代码生成系统架构图

## 系统需求分析

### 需求分析概述

1.普通用户：普通用户可上传数据集（最大支持上传5份数据集），用户可在上传的数据集中选择需要分析的数据集进行预览，根据预览界面的数据分析报告生成详细数据探索分析报告，该报告为用户进行数据清洗提供指导方向。若数据集无异常数据，则可通过数据建模界面选择指定列需要的模型。根据用户配置，前端弹出窗口展示生成的数据文件。由代码生成器生成的代码支持导出（不支持运行）。

2.高级用户：高级用户可上传数据集（最大支持上传10份数据集），用户可在上传的数据集中选择需要分析的数据集进行预览，根据预览界面的数据分析报告生成详细数据探索分析报告，该报告为用户进行数据清洗提供指导方向。若数据集无异常数据，则可通过数据建模界面选择指定列需要的模型。根据用户配置，前端弹出窗口展示生成的数据文件。由代码生成器生成的代码支持导出，同时支持生成代码的运行，并以可视化html的形式展示模型评估报告。

### 代码生成系统流程图：

图示

描述已自动生成

图2-2 代码生成系统流程图

## 系统安全分析

本系统基于Web应用实现，因此对于Web安全的防御也在考虑范围之内。基于系统的功能，对于可能发生在此系统的常见的web网络攻击的方式进行了识别以及预防，包括sql注入、路径穿越、恶意注册等。

### sql注入防御

sql注入是指web系统未校验用户的输入数据，使得攻击者可以在web应用程序中的输入框（可能对应sql查询语句的位置）输入内容的末尾添加额外的sql查询语句,从而实现欺骗web后台数据库[服务器](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E6%9C%8D%E5%8A%A1%E5%99%A8/613818)，执行一条未经授权的任意内容查询，得到用户所属权限之外的数据信息。

考虑到本系统的功能特性 ，后台数据服务使用了非关系型数据库，对于sql注入问题进行了有效的规避和预防。具体介绍参见第3章。

### 路径穿越防御

目录穿越的目的旨在访问存在在网站根目录外的文件或目录。通过浏览web应用的页面，攻击者可以寻找存储在 Web 服务器上的其他文件的相对路径。

本系统对路径穿越的识别主要存在于登录界面。为防止用户初次登录后，知悉了系统首页url后，能够直接跳过登录页面到达系统首页。由于系统首页的各个功能调用各类功能需要通过session中的用户名进行识别，获取不同用户的数据集，因此需要在系统实现过程中，需要禁止通过浏览器手动修改当前session，穿越到其他用户的主要界面。为防止此类情况的非授权跳转，这里使用了vue的导航守卫功能，通过控制session中是否存在用户名来控制页面的跳转。代码如下所示：



### 恶意注册防御

通常情况下，每条用户信息会存放在数据库表中。在用户注册时，注册接口可以通过浏览器的开发模式获取，为了防止用户拿到注册接口后，遵循注册规则无限制地注册账号，导致后台数据库爆仓。因此在用户注册时，本系统通过配置邮件服务器，使用邮箱号进行注册账户，保证每个邮箱后仅能注册一个账号，避免攻击者恶意注册账号。

## 本章小结

本章就代码生成系统的整体架构和功能需求进行了详细的介绍和图解。使用Internet应用程序的B/S结构来实现网站，采用前后端分离的架构方式，详细的介绍了代码生成系统的存储，引擎，展示结构。并通过流程图的形式来介绍了下整个网站系统的流程，以及各个模块所具有的功能。此外，本章就Web安全的角度出发，考虑了本系统在实际场景下可能出现的情况做了预防。

# 数据库分析与设计-MongoDB

## 后台数据库以及MongoDB概述

### MongoDB数据库介绍

NoSQL 数据库是非关系性数据存储的一种流行界限和定义,它改变了非关系性数据库和理论中长期占有主导地位的情况。非关系式的数据存储方式不仅仅是一种必须有固定的表架构,而且一般不必再根据所有的表之间做出任何连接。 NoSQL 数据库并没有使用任何一种传统的关系式数据库二维表格式模型,通常是采用键值格式来进行数据的存储。MongoDB 是 NoSQL 中最为普遍采用的一种数据库[3]。

MongoDB是基于分布式管理的用于大容量文件存储的数据库。由 C++ 语言编写。MongoDB的目的是提供一种高性能的数据存储解决方案，该解决方案具有更好的性能，并且易于为Web应用程序扩展[3]。MongoDB的文档存储格式通常与JSON相似，并且存储的内容通常是文档类型。 这样，我们将有机会在实现关系数据库的某些功能的基础上对某些字段建立索引，同时具备更好的可拓展性。

### 选择MongoDB数据库分析

本系统以数据集为核心构造数据挖掘代码，由于不同的数据集拥有不同的列索引以及不同的数据格式，导致存储的数据类型必定会很复杂，考虑到平台在代码生成/数据预处理过程中会对频繁访问数据集，同时对数据集的不同属性进行不同的操作，如果将数据集文件以数据库二维表的形式存储在关系型数据库，则无法遵循传统关系型数据库所定义的ACID规则。

## 数据集合的具体设计

### 数据库集合设计展示

以下就是本网站系统中所用到的所有数据集合

1. user\_model：用户集合：存储使用用户信息以及用户上传数据集的列表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 备注 |
| \_id | ObjectId | 默认自增id |
| username | String | 用户名 |
| password | String | 密码(base64加密) |
| email | String | 注册邮箱 |
| isVip | Boolean | 高级用户标识 |
| vipEndtime | Double | Vip到期时间 |
| dataset | Array | 用户上传的数据集名称 |

表3-1

1. dataset\_model；数据集的集合，用来保存整个平台上传的全部数据集

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 备注 |
| \_id | ObjectId | 默认自增id |
| username | String | 用户名 |
| dataset\_name | String | 数据集名称 |
| columns | Array | 数据集的各列名称 |
| data | Object | 数据集内容 |

表3-2

1. temp：临时集合，用于存储验证码内容

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 备注 |
| \_id | ObjectId | 默认自增id |
| address | String | 发送目标地址邮箱 |
| check\_code | Int32 | 验证码 |
| send\_time | Double | 验证码发送时间 |

表3-3

1. 构建历史集合

|  |  |  |
| --- | --- | --- |
| 字段名称 | 字段类型 | 备注 |
| \_id | ObjectId | 默认自增id |
| address | String | 发送目标地址邮箱 |
| check\_code | Int32 | 验证码 |
| send\_time | Double | 验证码发送时间 |

表3-4

## 本章小结

本章就系统的实际应用场景出发进行分析，选择MongoDB进行数据存储，使得数据存储更具备拓展性。针对业务需求，将用户数据集合，数据集集合，临时数据（用于保存用户注册账户的邮件验证码）集合，分别进行储存，以实现降低整个系统业务各个模块之间的耦合度，提升运行效率的目的。

# 数据挖掘代码生成器模板代码设计

代码的生成实现方式主要采用基于模板的代码生成方法[2]。原理是把功能的需求拆分为二,第一部分是相对固定不变的部分,即静态部分，大多是来自后台系统中的模板文件；另一部分是根据外部输入变化的部分，即来自前端用户的输入（如选择哪些列做数据预处理，选择什么模型构建数据等等)



图4-1基于模板的自动代码生成

一般而言，生成代码需要经由三个维度进行考虑。第一个维度是寻找复用性高的代码，也就是可以直接使用的代码；第二个维度是有规律的代码（即结构相似的代码），此类型可通过自定义配置进行生成，例如不同的数据集的不同名称，不同字段，字段有着不同的类型，但是读取数据，切割数据，训练模型的代码却是相同的。第三个维度是没有规律的代码，即我们在代码生成中需要考虑的特殊场景。

## 数据清洗代码生成器

在数据挖掘领域，我们经常会遇到提取特征数据异常的情况，如数据丢失和噪声。 如果不处理数据，将直接干扰最终挖掘模型建立后的模型结果，甚至最终模型也会失效，从而导致任务失败。所以，在建模之前进行数据清洗操作是非常有必要进行的，这也是在数据挖掘过程中不可忽视的，需要投入精力的重点。

### 常见数据清洗方法简介

数据清洗大致上分为数据读写、探索与描述、对数据进行简单处理、对重复值处理、缺失值处理、异常值处理、文本字符串的处理……。依据上述步骤，本系统提供的代码生成器支持缺失值填充、重复项删除、异常值删除、文本筛选（查询）、数据标准化等操作。

其中对于缺失值的处理，我们常常通过对这些值进行数据填充，以达到处理、规范数据的目的。缺失值填充的方法主要包括根据众数填充、按0填充、根据均值填充、使用上/下一个数据填充或者数据分析师根据对业务需求的理解，自定义填充数据等等多种填充方法。

同时，本系统提供数据标准化的相关方法如标准差标准化、最大-最小规范化等。关于数据标准化。数据标准化是指将数据按比例缩放，从而使它落入小型的特殊区域。在某些可以比较和评估的指标处理中经常都会被我们用到,去除这些数据的单元和量级限制,将其他指标转化成为无量纲的纯数值,无量纲的纯数值可以更加便捷地分析和处理不同的单位或多个量级的指标,使得这些指标可以对其他数据进行比较和相互加权[4]。

最小-最大规范化——又被称作离散标准化，这种规范化方法是通过对原始数据进行线性变换，将原始数据值映射到[0, 1]的区间内。其转换公式如下：



最大最小规范化的方法保留了原来数据特征中所有数据存在的相互关系，是消除量纲和数据值范围影响最简单的方法。这种规范化处理方法的一个缺点是如果结果数值太过于集中且某些数值很大,那么规范化后的各种结果数值都可能接近于0,并且差别不大。

标准差标准化也称为零-均值规范化，经过分析后所处理的数据平方均值定义为0,标准差定义为1。转化公式为。



其中，为原始数据的标准差，为原始数据的均值，上述公式是当前实际应用中使用频率最多的数据标准化方式。

### 数据清洗代码模板设计

数据清洗的模板相对简单且固定。通过固定的读取方法，对不同数据列调用不同的数据清洗方法，清洗方法通过预定义的参数文件获取并填充。数据清洗的代码均通过pandas库完成。

数据清洗方法我们通过将上一小节提及的清洗方法以json文件形式存储，构造数据清洗方法—方法代码的映射关系，达到快速构建的效果。每当用户添加一条规则时，通过映射关系文件找到对应代码，进行逐行添加。映射关系文件如下图所示。（以下json文件中的df表示通过pandas读取的数据集文件,cols为需要操作的列，其他部分为对应数据清洗方法的调用）

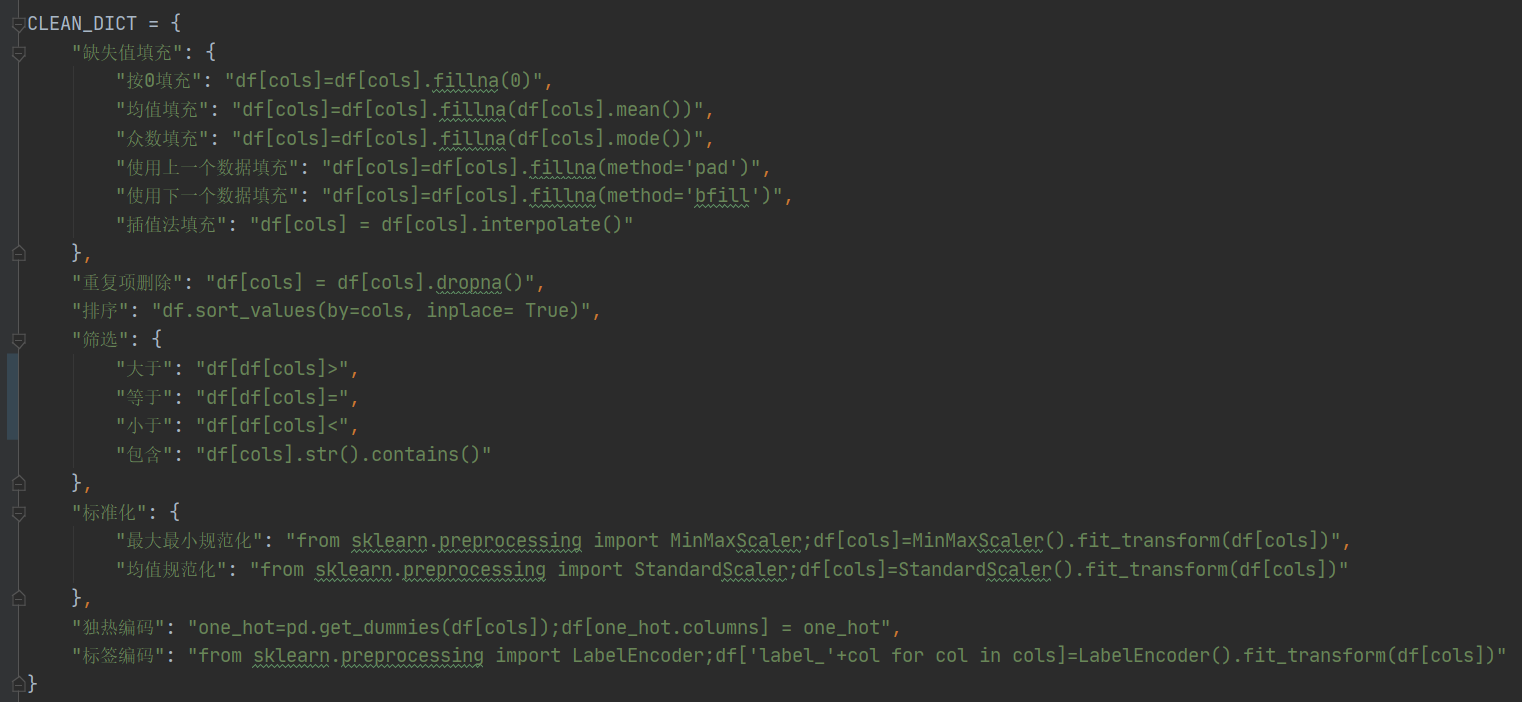


图4-1-2 数据清洗方法选择关系映射

### 数据清洗代码生成交互

前端页面通过提供下拉框以供用户选择，通过自定义多条清洗规则（规则形式:对xxx列进行xxx操作，具体包含的操作方法在上节中已做介绍）进行代码生成，具体流程图以及界面原型图如下

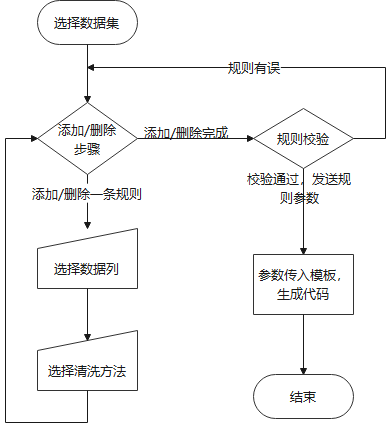


图4-2 数据清洗代码生成交互流程图



图4-3数据清洗代码生成原型图

## 数据建模（机器学习）代码生成器

数据建模是在数据分析过程中必不可少的一部分，在人工智能的发展日益成熟的今天，机器学习方法被应用在了方方面面。因此，这里当我们谈到数据建模时，也绕不开对机器学习的讨论。其核心思想是创建一种算法，这种算法可以从我们提供的一系列数据中挖掘出特定的规则，这种规则可以达到不必为特定问题编写特殊代码的目的。这样以来，我们需要做的工作就是将应用场景中产生的数据“馈送”到机器学习创建的算法中，然后它将在数据上建立自己的逻辑，建立起相关的数据模型。

### 有监督与无监督学习算法

机器学习的划分主要包括两个大类，分别是有监督学习和无监督学习。先讨论有监督学习，这种学习类型是利用现有的训练样本（包括标签）来训练数据，从而获得最优的模型，然后使用该模型将所有输入映射到相应的输出，然后判断输出的预测值，从而实现 回归和分类任务，以便能够预测或分类同一任务的数据。 例如，当我们获得带有标准答案的练习然后参加考试时，与没有标准答案的其他练习相比，我们的准确率更高。 例如，当我们年轻的时候，我们不知道牛和鸟是否属于同一类动物。 当我们长大后，随着各种知识的不断输入，我们的大脑模型变得越来越精确，因此判断动物是否属于同一种类的结果变得越来越精确。

关于有监督学习方法中分类与回归的区别通过下表进行比对。下表就分类与回归问题的输出类型、目的、评价方法以及应用场景进行对比，如下表 4-2-1所示。下表提及的评价方法的会在下一小节进行说明。

表4-2-1 分类与回归模型简介

|  |  |  |
| --- | --- | --- |
|  | **分类** | **回归** |
| 输出类型 | 离散数据 | 连续数据 |
| 目的 | 寻找决策边界 | 找到最优拟合 |
| 评价方法 | 准确率，混淆矩阵等 | 平均绝对误差、均方误差、决定系数等 |
| 应用场景 | 分类问题是用于带有标签的属性值，通常标签是离散的。例如，当判断一幅图片上所给出的植物是一棵树还是一朵花时，分类结果并没有一个模糊的概念，最终正确的答案只有一个。 | 回归问题通常是用历史的数据来预测未来特定场景下的数据值，例如未来几周的天气情况，某只股票的价格走势，或者若干年后某个地区的城区房屋价格等等，例如一支股票的今日收盘价为为15.91元，通过对他的历史数据学习回归，我们得到预测的今日收盘价为为15.89元，基于此，我们就可以判定这是一个拟合较好的回归模型，回归问题是对未来即将发生的真实值的尽可能近似的预测。 |

关于无监督学习，是指对没有给出一个概念性标签（如数据集分类）的训练样本数据集进行学习和训练，从而发现训练样本集中的结构性特征，并随后进行聚类。这里，数据集中所有的训练数据中的标签（分类）是未知的,因此训练样本的分歧比较高。即我们事先没有任何训练数据集的样本标签，而需要直接对已有的数据进行训练，以及构建模型。我们熟知的典型的无监督学习算法是聚类算法。

因此本代码生成器主要是通过构建回归、分类、聚类问题的代码生成模板，建立快速生成对应模型需求的代码。

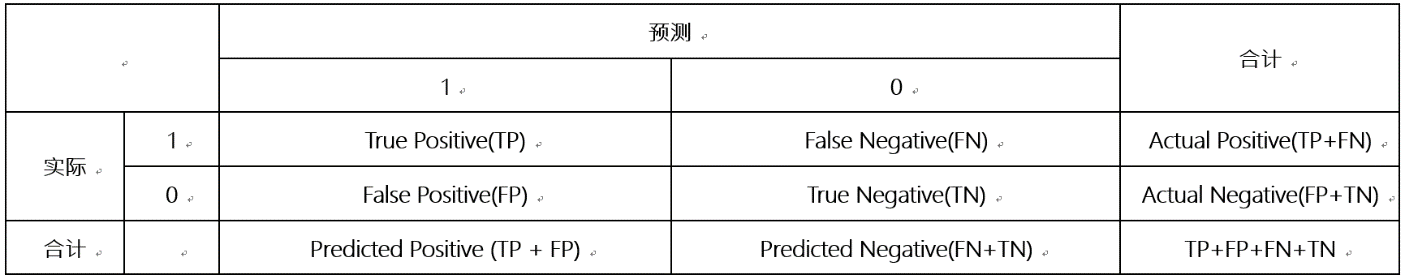
### 模型评估方法简介

在模型建立完成后，我们还需要对模型制定不同的规则进行评价来判断建模效果的优劣。因此，我们通过分类、回归、聚类三种模型分别定义了不同的评价指标。

#### 分类模型评价指标

在分类模型的评价过程中，我们往往以混淆矩阵为核心进行模型好坏的评估，混淆矩阵如下表所示。

表4-2-2 混淆矩阵



其中，表的横轴为真实值结果、纵轴为模型的预测值结果。在对角线上为预测正确数目，反对角线上为预测错误。上表中TP(True Positive)表示将正类预测为一个正类;FN(False Negative)则表示将正类预测为负的一类;FP(False Negative)表示将负的类别预测为正的类别;TN(True Negative)表示将负的类别预测为正的类别。通过上述四个评价指标我们可以做出进一步的计算。在本系统中，我们通过混淆矩阵，计算模型的准确率、漏报率、误报率，并依据这些值绘制ROC曲线可视化结果，并计算AUC值对模型的好坏进行评估。

①准确率(Accuracy)：对于给定的测试集，分类模型正确分类的样本数与总样本数之比即Accuracy = (TP+TN)/(TP+TN+FP+FN)。

②漏报率(FPR)：即负正类率，指本来正常（Condition negative），但是误认为是异常（Predicted condition positive）的值，即FPR= FP/(TP+TN+FP+FN)。

③误报率(FNR):本来异常（Condition positive），但是误认为是正常（Predicted condition negative）。即FNR = FN/(TP+TN+FP+FN)。

④ROC曲线

ROC曲线中主要的两个指标就是TPR和FPR。其中横轴代表假正率(FPR)，纵轴代表真正率(TPR)。一个标准的ROC曲线图如下图4-2-1所示。

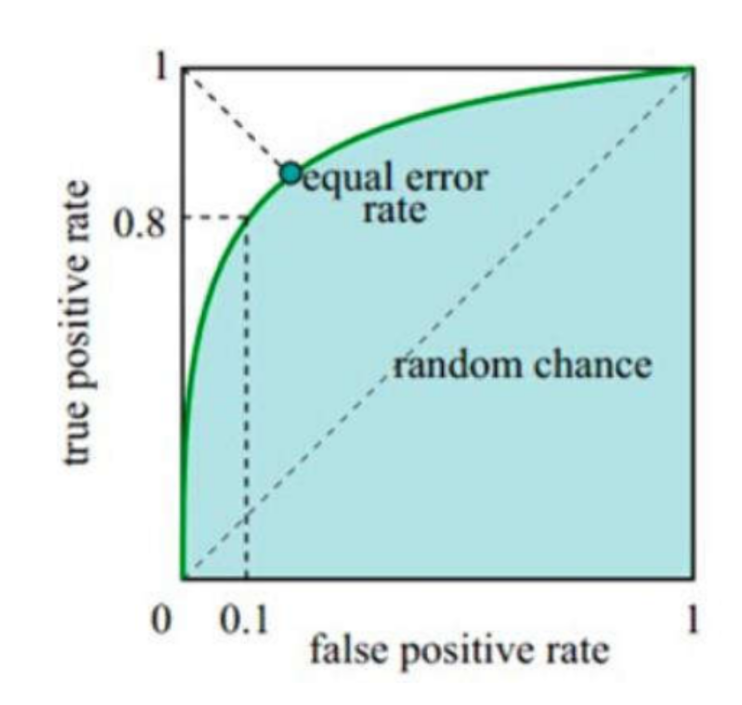


图4-2-1 roc曲线图

根据roc曲线我们就可以如此地判定一个模型的性能优劣：FPR表示一个模型的虚报（或者说预测错误）的严重性和准确性，而TPR则表示一个模型的预测和正确性。因为我们一般期望虚报（或者预测错误）的数目需要越少越合理，覆盖（预测正确）的数目越多越好。即TPR越大，同时FPR越小（其中表ROC曲线越陡），那么该模型的性能和效果就会边得更加良好。

1. AUC值

AUC值的全称为曲线下面积（Area Under Curve）。这里的曲线也即是上述的ROC曲线。AUC值指得是一个概率值，当你随机地挑选一个正的样本和一个负的样本，当前分类算法根据计算得到的Score值将这个正样本排在负样本前面的概率就是AUC值。通常情况下，AUC数值介于0.5-1之间，可以评价分类器的好坏，数值越大说明越好[5]。

#### 回归模型评价指标

（一）平均绝对误差（Mean Absolute Error，MAE）



平均绝对误差就是指预测值和真实数据之间的绝对值之间的差值的平均。平均绝对误差可以很好地体现模型中给出的预测值和真实值之间误差的实际情况.

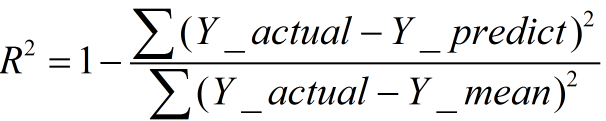
（二）均方误差（Mean Squared Error，MSE）



预测值与真实值偏差的平方和除以观测次数的比值。这也是通常在线性回归模型中最常用的目标函数，线性回归过程中训练模型参数尽可能的让该损失函数减小。那么模型之间的对比也可以用它来比较。

MSE可以很好地评价一个模型在实验中预测效果的变化，我们所计算得到的MSE的值越小，说明这个预测模型在对实验结果数据进行预测时具有很好的准确性。

（三）R-square(决定系数)



分母代表了数据集中属性的整体离散程度，分子代表了预测的数据与来自原始数据之间的误差平方和，二者进行相除就可以有效地消除来自原始数据的离散性程度的干扰。

其实“决定系数”是通过数据的变化来评价模型拟合效果的好坏。

理论上取值的范围（-∞，1], 正常的取值范围位于区间[0, 1]。实际操作中通常会选择一条拟合效果较好的曲线计算R²，因而很少有曲线出现-∞。

R²越是接近1，表明我们在这个模型训练中所取得到的参数方程中各种变量对y的解释能力越强，这个模型在数据的拟合效果的也比较优秀。反过来如果R²越接近0，表明模型拟合的越差。

缺点：数据集的模型样本数量越多，R²也就越大，因此，不同类型数据集经过模型输出结果的比较会存在着一定的误差。

#### 聚类模型评价指标

无监督聚类中不存在样本标签，因此并没有可以直接计算数据的的聚类评估方法。 但是，我们可以从每个聚类的离散程度和不同聚类之间的分散性评估聚类模型的最终效果。 常见的是轮廓系数和calinski harabasz指数[6]。

 1、Calinski-Harabaz Index

Calinski-Harabasz分数值的数学计算公式：

s(k)=

其中，M为训练集样本的个数，K是类别的数量。 BK是不同的类别之间的协方差矩阵，而wk是每一个聚类类别中所有数据的协方差矩阵。 TR是矩阵的迹线。 也就是说，类别中每个数据的协方差越小，模型中的聚类效果就会变得更佳，类别之间的协方差就会变得更大，此类模型的calinski harabasz得分便会越高[6]

2. 轮廓系数Silhouette coefficient

轮廓系数（Silhouette coefficient）主要定义适用于一些实际类别信息，尚无完全充分了解的应用场景。对于一个单独的类别样本，设a值的是与它同类别中其他样本的平均距离，b是与它距离最近不同类别中样本的平均距离，轮廓系数的公式由此定义为：

s=

对于任何一个样本的集合，其轮廓系数是所有集合中各个样本轮廓系数的一个平均值。轮廓系数的取得值区间范围为[−1,1]，同一种类别样本之间越距离相近且其他不同类别样本距离越远，分数就会越高。

### 数据建模机器学习方法优劣

本文主要围绕数据挖掘的代码生成方法进行展开，对于使用到的各类机器学习的模型原理不做探究，仅针对不同模型的优劣做出对比以供参考。

#### 决策树

优点：

* 1. 决策树是一个可以轻松地被我们所了解与诠释的。在对模型进行解释之后,人们就已经具备了理解决策树分类的意思;；
  2. 对于决策树来说，数据的准备往往是简单或没有必要的。其他分类模型常常会要求首先对这些数据进行预处理,把这些数据做得更加一般化,比如在树中去掉许多重复或者是空白的属性。
  3. 能够处理连续和离散属性。而其它的技术一般都需要简化大量的数据属性。决策树采用的是白色盒子模型。如果我们能够给出一个观察和比较的决策树模型,那我们就能够很容易地从决策树的模型中来推断得出一个相应的判断分类和判别逻辑的表达式。
  4. 可以轻松通过一系列静态测试来对模型进行模型评估。表示有可能测量该模型的可信度。
  5. 在尽量短的时间内能够对大型数据集训练出可行且效果良好的模型。
  6. 同时也可以对具有多种不同属性的数据集训练决策树模型。
  7. 决策树模型也可以扩展应用到一些比较大型的数据库中，同时它的大小不受数据库的大小的影响。

缺点

1. 对于每个类别中的样品数量存在不一致的数据，在决策树模型当中,模型训练所获得的数据信息增益的结果往往会偏向于那些具有更多数值的样品数据特征。
2. 决策树处理数据集含有缺失数据情况时会存在一定困难。
3. 过度拟合问题的出现。
4. 往往会忽视数据集中属性之间的相关性特征。

#### ANN人工神经网络

优点：数据集分类模型的最终精确率高，并且可以实现数据集的并行，分布式处理能力强，可以实现分布式存储和模型学习能力强，数据集中的噪声具有很强的鲁棒性和一定的容错能力，可以完全逼近实际问题中的复杂非线性关系，具有联想与记忆等功能。

缺点：在对神经网络的设计和构造中，需要大量的参数信息来帮助其参与计算。例如不同的网络拓扑，网络各层权重的初始值和阈值； 模型中各层之间的学习过程无法详细地观察，输出结果往往很难理解，会直接影响模型最终结果的可信度和接受性； 模型的培养训练与学习往往需要一段时间，即使对于不良的网络模型，也有可能是无法实现其学习的目的。

#### KNN最近邻算法

**优点**

1. 计算简单且有效。每一次重新训练模型计算花费的代价较低。
2. 该算法的计算时间及其空间与训练集之间的规模大小是呈线性关系的（有些情况下不算太大）。
3. 由于KNN算法主要依赖每个待分类中心点以及它们周围有限个相互邻近的样品,而不是依赖于判别各个类中心点边界的方法和途径来确定它们所属类别的,因此对于各个类中心点在范围内相互交叉或者是重叠得比较多的一个等待分类中心点的样本来说, knn 算法较其他的分类方法更加适合。
4. KNN算法相对于其他一般类别的分析方法来说是比较合理的,它更有利于实现对样本中容量较大的类域进行自动化分类,而那些对样本中容量较小的类域选择采用 knn 算法往往很容易造成分类时出错。

**缺点**

1. KNN算法是懒惰学习算法（基本上不用训练），因此比其他一些积极学习的算法训练起来要快很多。
2. 对于最终分类类别的评分不是在0-1之间规格化的（不像其他算法最终得到分类的概率评分）。
3. 该算法输出结果的可解释性相对而言很差，不像决策树输出结果的可解释性较强。
4. 该算法在分类过程中存在一定的不足:例如，当训练样本分布不平衡时，如果数据集中某一类的训练样品容量很大，而其他类训练样品容量很小时，这就会极有可能造成当训练样品输入一个新的样品时，该训练样品最近的的K个邻居中大容量类的样品占大多数。这主要是因为,由于该算法只需要在每个点的中心附近"最近的"相邻样本,当某一类的相邻样本数量很大,又或者此类样本本身并不是一个接近目标样本的类别,或者此类样本很难找到一个靠近目标的相邻样本。当然,样品的数目和量的多少最终也不能直接影响正常运行的结果。我们可以通过对各分类给予权值(及对该样品距离较小的相邻邻居给予更高权值)的选择性方法加以改进。
5. 有限元的计算数据比较多。目前我们常用的一种解决办法就是提前将已知训练集的样本点提前删除并预剪枝处理,事先删除对训练集分类影响不大的部分。

#### SVM支持向量机

优点

* 1. 支持向量机可以有效解决在小规模样本的特殊情况下使用机器进行学习时的问题。
  2. 可以提高泛化性能。
  3. 可以解决高维、非线性问题。
  4. 可以避免神经网络网络结构中的设计和损失函数在极小值梯度下降过程中，容易出现局部的极小值。

缺点

* 1. 对缺失数据敏感。
  2. 对于非线性可分的问题没有一种完全可以被普遍接受的的解决办法，必须谨慎地选择合适的核函数才能够进行非线性转换的处理。

#### 朴素贝叶斯NBC

优点

1. 朴素贝叶斯模型起源于一种古典型的概率数学理论，有着扎实的古典概型的数学基础，以及比较稳定的分类计算能力。
2. 朴素贝叶斯模型所需参与计算和估计的各种参数非常少，对于那些存在缺失数据的信息集不太敏感，算法的实现和估计过程也比其他算法简单。

缺点

1. 在数据集特征维度较多或者数据集中各个数据特征之间相关度较大时，朴素贝叶斯模型的特征分类分析结果就可能会远远地低于决策树分类模型的结果。而且是当一个数据集中每个特征相关性比较小时，朴素贝叶斯模型的主要特点之一是其特征性能可以表现得很好。
2. 需要知道先验概率，同时分类决策存在一定错误率。

从理论上来说，朴素贝叶斯模型与其他的分类模型比较起来具有最小的误差率。但是，在我们处理现场实际问题时并非总会如此。 这主要是由于朴素贝叶斯模型认为数据集的各个特征彼此独立。 该前提在实际应用中通常不存在（我们可以首先使用聚类算法对数据集中具有高度相关性的特征进行聚类），这对NBC模型的正确划分和准确性也会产生一定程度的影响。

#### 逻辑回归

优点

* 1. 预测结果是界于0和1之间的概率；
  2. 逻辑回归可以被广泛地应用于具有连续性和一定类别属性的自变量；
  3. 容易使用和解释；

缺点

* 1. 由于模型中预测所得到的计算结果曲线通常呈“S”型，因此从一个log函数(odds)向每个不同类别的概率转化的过程不是线性的，在横轴两端随着​log(odds)值的变化所改变，所对应的概率变化很小，边际值非常小，而中间部分的概率的变化很敏感，很大。这也就导致很多区间的变量变化对目标概率的影响没有区分，无法确定一个阀值。
  2. 对于模型中自变量的多重共线性较为敏感，例如两个模型高度相关的自变量同时被放入到模型中进行训练，这样有可能会直接造成影响较弱的一个自变量回归模型无法满足我们的预测期望。需要通过利用变量聚类和数据分析等方法手段去选择具有国际代表性的输入特征，来降低所有输出特征之间的相关性[9]

#### 随机森林

优点

* 1. 在现有的大多数数据集上，相比其他分类算法有着很大的优点，分类效果良好
  2. 随机森立使得我们能够对很许多非常高维度（特征很多）的数据集进行训练和学习，并且不需要提前在这个时候就对数据集做出任务的特征选择
  3. 在训练结束后，随机森林能够提供出明确的结果，即哪些特征比较重要。
  4. 在训练随机森林模型的时候，对生成误差估计使用的是无偏估计，模型有很强的泛化能力
  5. 模型训练速度非常快，并且可以很简单的实现并行化方法
  6. 在对模型的训练过程中，能够发现到数据集各个不同特征之间存在的的相互影响关系
  7. 对于不数据分布不匀的数据集来说，它可以平衡误差。
  8. 即使数据集中存在一部分的特征缺失，模型仍可以维持较好的准确度。

缺点

* 1. 随机森林往往对一些存在噪音的数据集进行分类时会出现过拟合的现象

#### K-means

K-means算法是一种通过[迭代](https://baike.baidu.com/item/%E8%BF%AD%E4%BB%A3/8415523)的方式对其进行集合求解的聚类算法，它的算法步骤是，预先将数据分为K组，然后随机选取K个值作为初始的聚类中心，然后计算各个对象与各个簇的聚类中心之间的欧式距离，把每个对象分配给距离它最近的聚类中心。聚类中心以及分配给它们的对象就代表一个簇。每分配一个样本，各个聚类的聚类中心会根据聚类中现有的对象被重新计算。这个过程将不断重复直到满足某个终止条件。终止条件是没有（或最小数目）对象被重新分配给不同的簇，没有（或最小数目）聚类中心再发生变化，达到[误差](https://baike.baidu.com/item/%E8%AF%AF%E5%B7%AE/738024)[平方和](https://baike.baidu.com/item/%E5%B9%B3%E6%96%B9%E5%92%8C/783894)局部最小。

优点:

1. 算法原理很简单，算法实现也比较容易，模型训练的收敛速度快。
2. 聚类效果较优。
3. 算法的可解释度比较强。
4. 模型可以调节的超参数仅仅是聚类簇数k值。

缺点:

1. K值的选取不好把握
2. 对于不是凸的数据集比较难收敛
3. 如果各类别的数据分布不平衡，比如不同类别的数据数量严重不平衡，或者各隐含类别的方差差别较大，则聚类效果会很差。
4. 算法使用的是迭代，得到的最终结果只是局部最优而不是全局最优解。
5. 对噪音和异常点比较的敏感。

#### BRICH聚类算法

BIRCH算法可以不用输入K值(簇数)，这点而言与K-Means等其他聚类算法不同。BIRCH算法适用于数据集数量大的情况，这个特点和Mini Batch K-Means是有一定类似的，但是BIRCH算法更适用于聚类簇比较大的情景，而Mini Batch K-Means一般用于聚类簇数适当或者较少的时候。BIRCH除了解决聚类问题还可以额外做一些数据异常点的检测和数据初步按类别分类的预处理。当我们的数据集特征维度特别大，大于20时，BIRCH算法就不太适合，这时用Mini Batch K-Means可能得到更好的模型效果。

BIRCH算法的主要优点有：

1. 节约内存，所有的样本都在磁盘上。
2. 聚类速度非常快，只需要对数据集进行一遍扫描就可以建立CF Tree。
3. 可以识别噪音数据，鲁棒性强。此外还可以对数据集实现初步分类的预处理功能

BIRCH算法的主要缺点有：

1. 聚类算法的结果可能和数据真实的类别分布存在差异.
2. 对特征维度特别高的数据聚类效果很差。此时可以选择Mini Batch K-Means算法
3. 当我们的数据集的分布簇不是非凸的，又或者不是类似于超球体则模型的聚类效果不好。

### 数据建模（机器学习）代码生成模板设计

在自动生成数据模型建模代码之前，我们需要给用户提供代码生成的指导方向，即根据不同的问题需要传入对应的参数。数据建模时我们往往需要知道数据集的特征列、目标列以构建合适的模型解决问题。因此特征列，目标列以及模型类型是代码生成器的必要参数，基于此，我们的代码生成器构建了如下的操作流程：

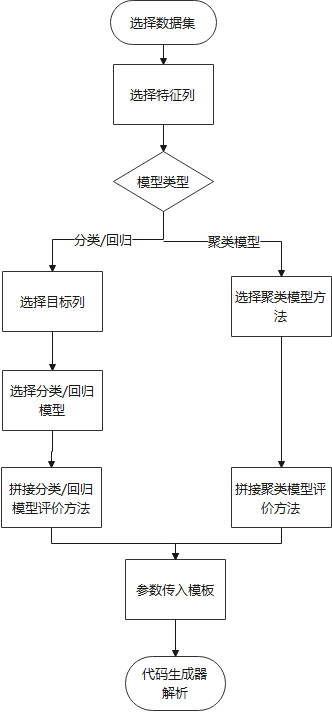


图4-2-4-1 数据建模代码生成器用户使用流程图

用户传入参数后至代码生成器后台，后台将参数解析，并在对应模板中扩充代码。传递的参数主要包括数据集名称，模型类型（分类、回归或者聚类），特征列名称，目标列名称（仅分类，回归任务包含），应用模型（一个或者多个）等参数传至后台数据建模的代码生成的逻辑主要如下图所示。

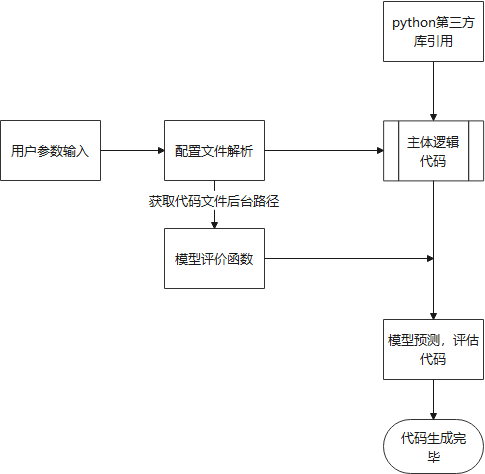


图4-2-4 数据建模代码生成逻辑流程图

首先需要引入建模需要的python第三方库，包括用于数据分析的pandas，numpy，用于机器学习建模的sklearn。

通过读取用户的参数输入获取需要构建的模型类型，进行主体逻辑的代码拼接。主体代码逻辑包括读取数据集文件、选择特征列，目标列，并拆分数据集为训练集与测试集。拆分完成后，将训练集作为模型的输入进行训练，并将训练的好的模型（模型支持多选）以列表的形式保存，作为后续预测模型方法的输入。

根据用户选择的分类/回归/聚类任务下的不同模型，进行主体逻辑拼接。生成器获取模型的映射关系以json形式储存，如下key-value结构所示。其中key为具体模型的名称，value为该python调用sklearn中封装的api的具体代码，在生成代码的过程中会优先导入这部分方法，导入方法完成后会对这部分字符串进行分隔，转换成python中实际运行的语法,配置如下图4-2-4-1所示：

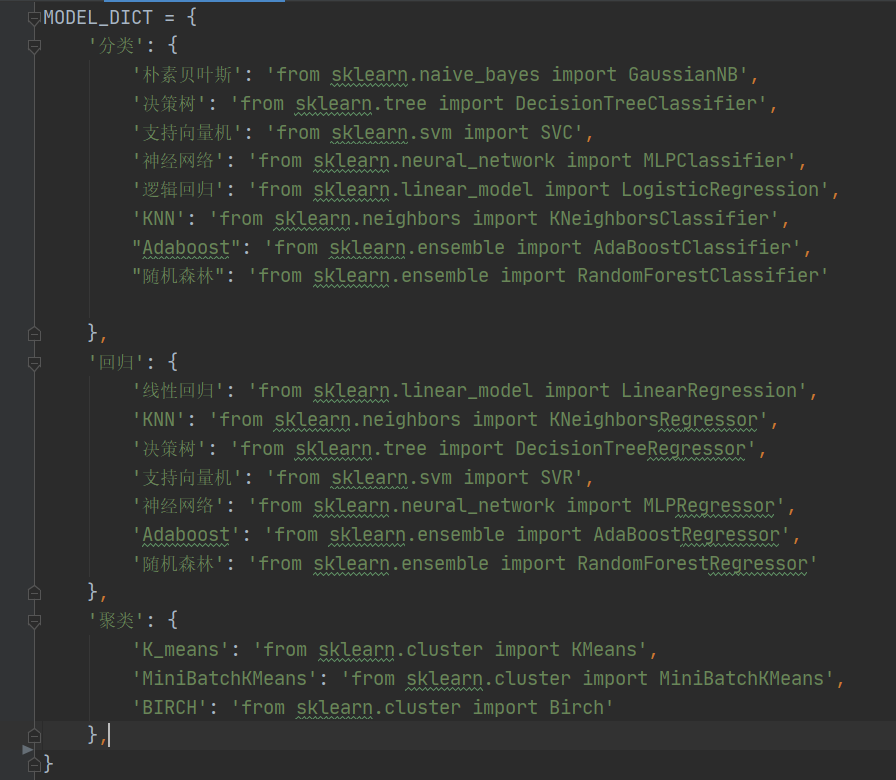


图4-2-4-1 数据建模模型选择关系映射

此外，针对分类模型与回归模型分别有着不同的模型评价参数，计算过程代码以单独模块的形式保存在后台。通过任务类型读取模块文件，将文件内容拼接于主体逻辑代码之后。至此，数据建模的代码生成已经完成。

基于上述流程，数据挖掘模型的交互界面(用户参数选择)原型图如下所示

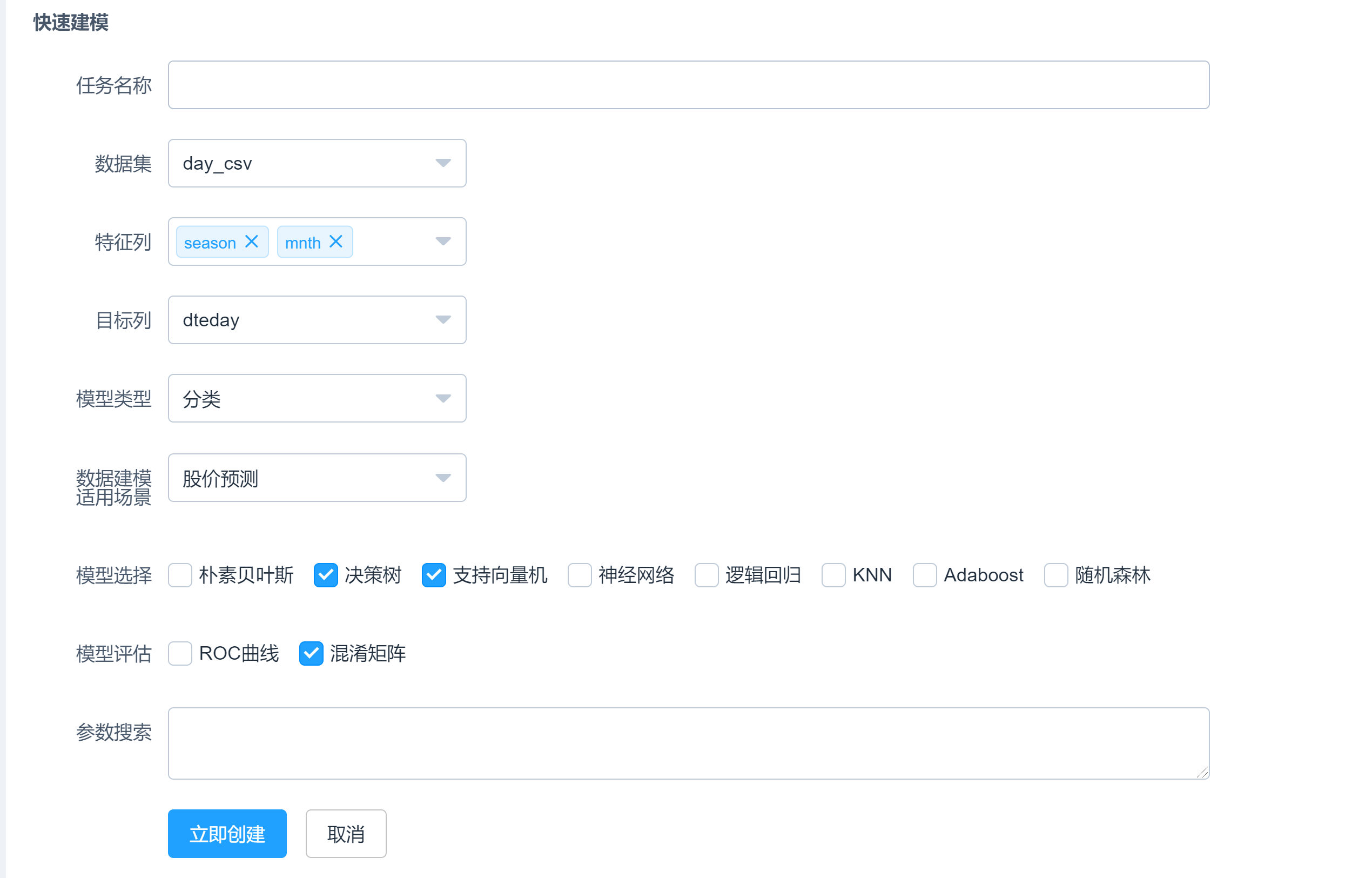


图4-2-4-2 数据挖掘代码生成器模型选择交互界面原型图

代码生成界面的效果如下。

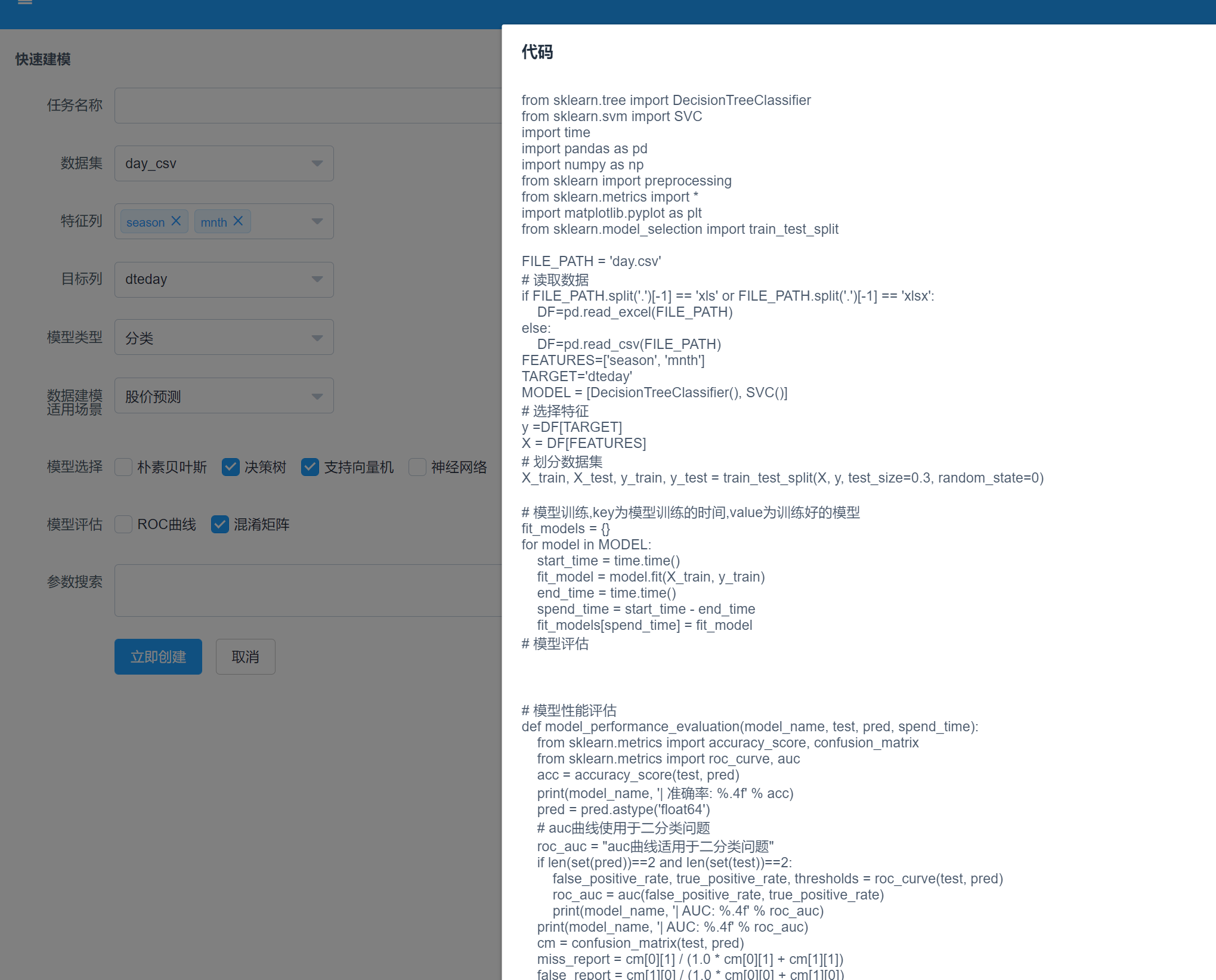


图4-2-4-3 代码生成效果图

在快速建模界面，提供了一个用户可选下拉框，以供用户填写当前分析的数据集适用的场景。下拉框的选项为本系统的内置场景，选项可选股价预测、天气预测、价格预测、属性(动物/植物/疾病)分类等等。该场景分类属性内置为后续功能——根据用户上传数据集能够实现推荐模型做出准备。详情见第5.4章常用模型参数模板化/推荐。

## 本章小结

本章采用基于基于模板的代码生成方法，通过研究数据清洗、数据建模的流程就行关键参数的提取，用以动态嵌入模板生成代码。首先对常用的一些数据清洗方法，计算过程，以及数据清洗方法的模板设计思路做了介绍。接着在已有无异常缺失值的“净”数据的前提下，就常用的回归、建模和聚类的机器学习方法以及分别对应的评价指标做了介绍，并就各种模型方法的优点与缺点做出了详尽的对比。基于模型的理解以及评价指标的不同，将构建模型必要的参数（模型类型，特征列名称，目标列名称，应用模型）进行提取，动态插入到设计好的主体逻辑代码中生成完整代码。 基于对数据挖掘建模代码生成器流程的分析，将界面原型图以及生成代码的呈现样式做了介绍。

# 数据分析系统引擎设计

## 一键生成探索性分析报告

探索性数据分析（EDA）是数据分析流程中的第一步。探索性数据分析的主要实践思想在于，用于通过使用尽可能少的先验假设，探索现有数据（尤其以调查或者观察手段获得的源数据），使用统计图表，方程拟合，计算特征量等。尤其是当我进行数据分析时，各种混乱的“脏数据”通常会无所适从。当我不知道从哪里开始时，了解我现在拥有的数据非常有效。

进行探索性数据分析时，我们在了解各字段的意义以及类型的基础上，可以通过单变量分析、双变量分析、多变量分析等思路进行对应数据的可视化。单变量分析主要看该字段是离散型或者连续型。如果是离散型数据否存在子名称，每个子名称是否存在字段限制，如果是连续型则观察字段的分布情况以及是否存在范围区间;双变量分析重在观察两个变量之间是否存在线性关系，一个变量是否会对另一个变量产生相关影响，变量是否收时间，地区等特定条件的影响等;多变量分析则是在双变量分析的基础上站在更高的维度去观察数据，多变量分析的重点主要是针对数据集中的数值型数据去观察，最经典的做法是去绘制所有数值型变量的热力图，通过热力图查看多个变量之间的相关程度，根据该结果去选择特征进行进一步筛选，分析。

通过上文可以看到，探索性数据分析是一个相对繁杂的过程，其结果对最终数据分析的结果起到了很大一部分的决策作用。然而，当数据维度较大时，人工去进行列与列的比对的复杂度往往是呈几何倍数的增长，因此也容易造成对部分关键特征的疏忽。为了便于用户进行后续数据清洗以及数据建模的操作，本系统在数据预览界面提供“生成报告”的功能以方便一键完成探索性分析的过程。

为了实现一键式探索性分析，后台引擎使用了pandas\_ Profiling库。 pandas\_Profiling基于pandas的DataFrame数据类型，可将其简单，快速地用于探索性数据分析。 对于数据集的每一列，pandas\_ Profiling将提供以下统计信息：

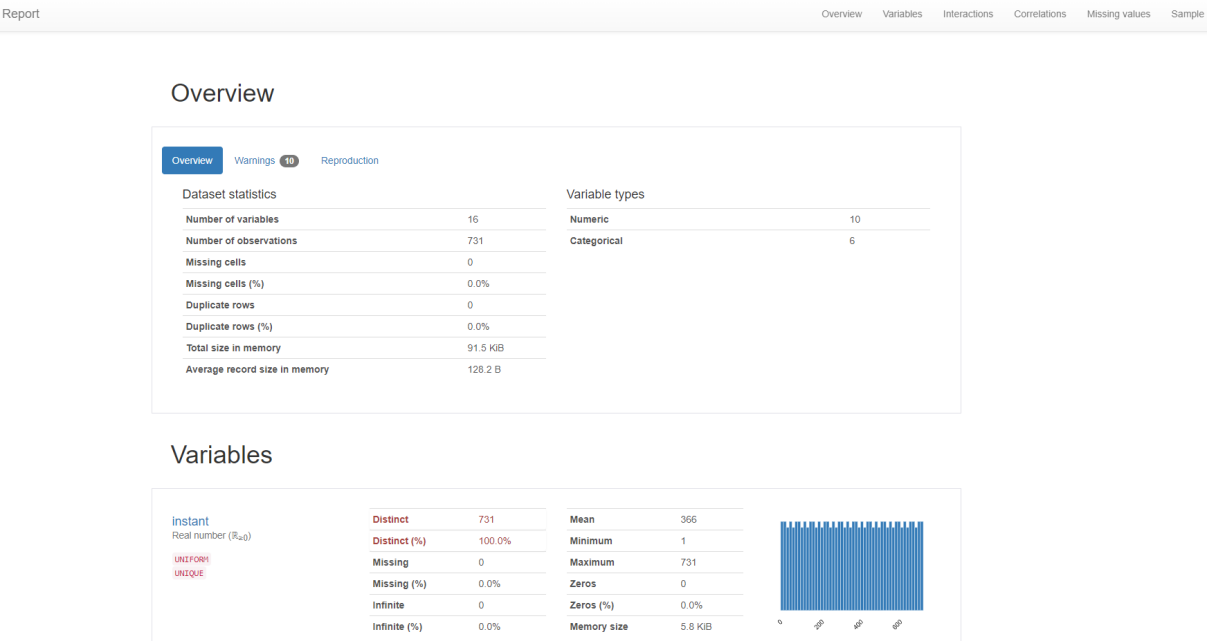
Ⅰ、汇总：数据类型，缺失值统计，唯一值统计，内存占用统计；

Ⅱ、分位数统计：最值，中位数值，范围值，四分位数等待；

Ⅲ、描述性统计：标准偏差，平均值，众数, 绝对中位数偏差，变异系数，峰值，偏度系数等;

Ⅳ、最频繁的值，柱状图/直方图

Ⅴ、相关分析的可视化：突出显示高度相关（强相关）的变量，Spearman，Pearson矩阵相关色标数据分析报告结果预览如下所示



5-1-1 探索性数据分析报告概览

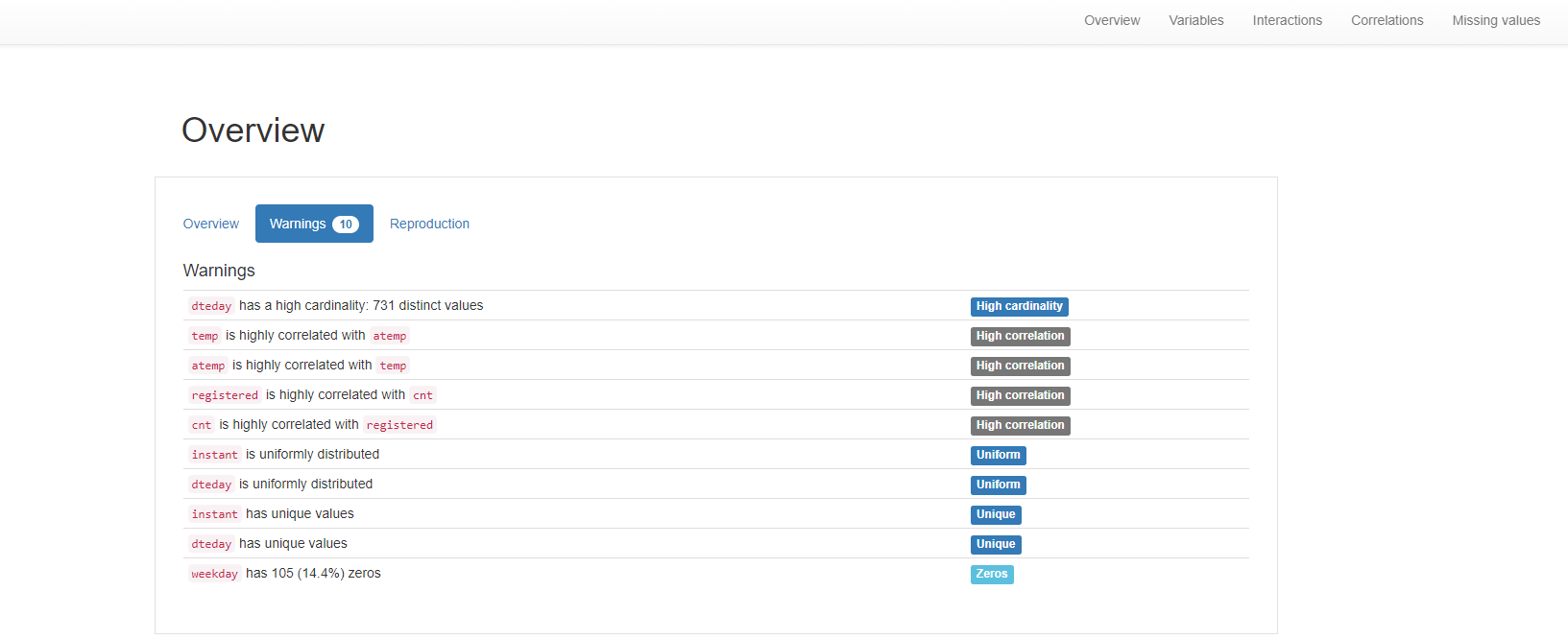


图5-1-2 探索性数据分析关键特征

在报告的概览界面会提供数据集的基本信息如图5-1-1所示。其中Variables统计了数据集中每一列的详细分析情况，图5-1-1中即截取了样例数据集instant列的详情统计。例如图5-1-1点开Warnings Tab页可以看到当前数据集中可能有重要意义的数据如图5-1-2所示，此类特征包括相关性高的两个特征，服从正态分布的特征，特征是否是离散值或者连续值，是否某列特征存在缺失值，或者无意义值（0值，空字符串等）过多的情况。具体的详情可通过右上角的五个页签跳转，查看各列特征的统计情况，分布情况等。

## 数据清洗引擎设计

由于不同的数据集有着不同的清洗步骤和方法，这意味着针对不同来源的数据所遇到的困难也是不同的。因此，固定顺序的数据清洗模板往往很难应用于所有的数据集，数据清洗引擎的功能主要是使得用户能够自定义进行数据清洗规则的排序，同时根据数据清洗规则进行处理，对规则的合理性进行校验，对于能生效的清洗规则，输出对应的新数据，而不仅仅是生成一份看似合理的清洗代码。

所以，数据清洗引擎主要在于规则合理性的校验，当所有清洗规则校验通过时，生成一份新的数据集，即清洗完毕的数据集。对于校验不通过的规则，则解析后台错误信息，在前端给出具体错误原因。基于此，设计如下引擎的工作流程如下。

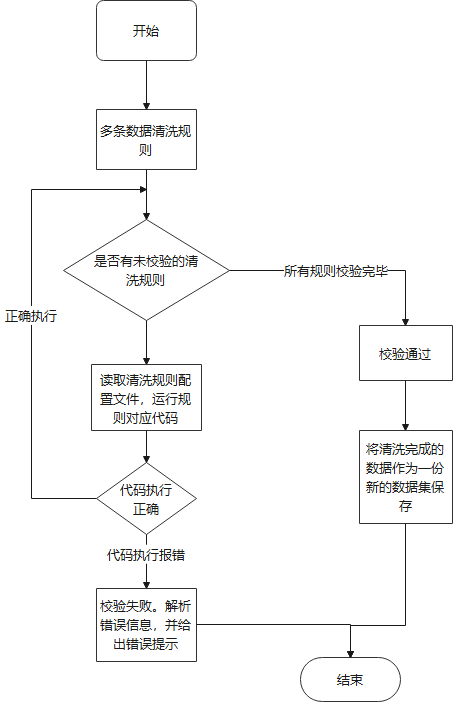


图5-1 数据清洗引擎流程图

## 数据建模（机器学习）引擎设计

在用户上传不同的数据集文件时，引擎通过读取数据集内容判断需要使用的模型。考虑到很多用户在实际应用场景中会对分类模型构建与回归模型的具体使用产生混淆，此外也需要考虑到用户可能产生的误操作，例如，模型类型将回归选择成了分类，由于回归模型的目标列往往是连续的，如若误操作选择成了分类模型，此时模型的输入会将所有的输入数据当作一个类别，从而产生几千上万个“分类类型”，如此可能导致系统性能的高负荷运转，甚至系统崩溃，在影响性能的同时，也使得用户的使用体验变得尤其糟糕。为了解决这个问题，数据建模引擎在代码运行前会对模型构建参数进行检查，这里我们预定义了如下模型检测规则:当检测到用户发起代码生成请求时，首先校验数据的样本总数否大于 50。如果样本小于 50 则提示需要收集更多的样本数据，以确保训练的模型有着足够的输入，从而获得有着良好效果的模型。如果样本数量大于 50， 则需要先判断预测的目标列的数据分布，即数据分布是离散型或者连续型。如果数据是离散型或者是标签类型数据，则判定该目标列是用于分类模型；若数据是连续型，则判定该目标列是一个用于回归模型的特征。当用户上传的数据集没有标签列，则提示用户使用聚类方法找到合适的标签，从而进行后续模型的构建。

当代码生成器运行生成了代码，引擎主要负责运行代码，并根据运行结果生成一份报告，用以展示模型结果。当有多个模型存在时，展示各个模型的结果对比，通过可视化数据表格的形式展示。

由于生成的代码从使用者的角度出发，在用户拿到生成的代码时，读取数据的步骤是通过加载本地数据文件到内存，进行后续建模操作。考虑到数据建模引擎运行在后端，可通过数据库直接读取数据集文件，达到节省服务器资源开销的目的，因此，引擎运行文件前会修改原始生成代码的数据源，即从本地文件系统加载修改为数据库读取。数据建模引擎的工作原理如下

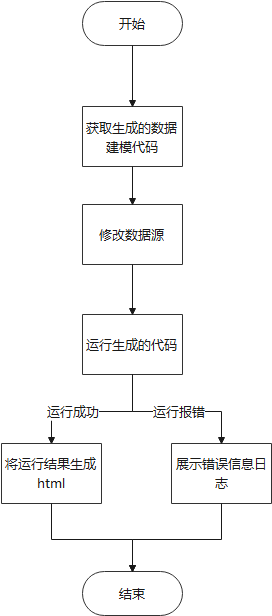
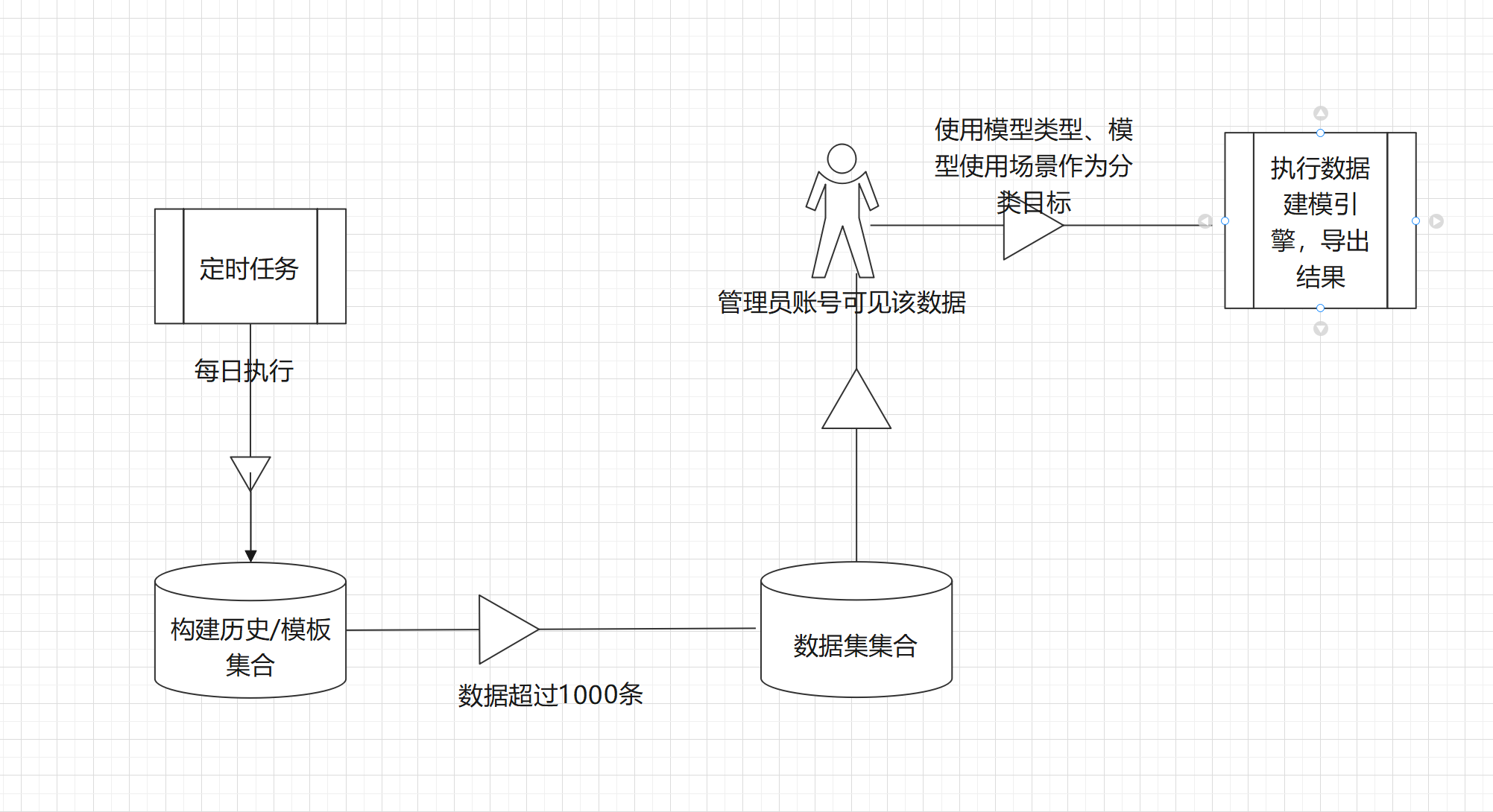


图5-2数据建模引擎流程图

## 常用模型参数模板化/推荐

由于当前系统并没有真实用户使用，因此对于每种模型的构建的频次 暂时无法预知。因此在当前系统实现过程中，在模型构建的地方已提前做好埋点统计，每当用户生成一次代码并执行导出代码操作时，系统会将此次用户的模型构建参数存入数据库中的模板集合。当数据量增长到一定程度时，通过将数据库中的模板集合导出为csv文件，将该csv文件作为一份新的数据集推送至管理员用户的数据集管理界面。通过使用数据建模，将构建学习模型，达到能够根据用户数据集识别出适配的模型。上述过程通过系统预定义定时任务来执行。定时任务每日执行一次，当数据集模板集合数量达到1000条时触发。工作流程如下



**图5-4常用模型参数学习流程图**

根据模型的结果在后期加入数据建模界面定义快速填充参数下按钮，从而使得系统形成闭环，达到推荐参数，智能选择参数的效果。

## 本章小结

在上一章提及的代码生成器已经生成对应功能代码的基础上。本章分别针对数据清洗代码，数据建模代码设计运行引擎。在数据清洗之前，根据数据集内容，后台引擎首先生成探索性数据分析报告，以供分析者决策之用。数据清洗运行引擎主要通过对用户自定义的清洗规则进行逐条校验，将校验失败的规则展示给用户，并给出对应修改建议。数据建模运行引擎通过执行生成代码，并依据模型类型生成对应的模型效果评估图表，当存在多个模型时，给出多个模型的效率对比以及关键评价参数的对比可视化图表，生成一份评估报告。使得自动代码生成器系统拥有更好的用户交互体验，以提升易用性。

此外，本章所涉及的引擎等均会占用较多的服务器资源，造成大量的性能开销。因此在引擎的使用权限上面做了相对应的权限控制，仅系统注册的高级用户可以使用。

# 参考文献

1. [王博](https://kns.cnki.net/KNS8/Detail?sdb=CJFD&sfield=%e4%bd%9c%e8%80%85&skey=%e7%8e%8b%e5%8d%9a&scode=35790922&acode=35790922" \t "knet);[舒新峰](https://kns.cnki.net/KNS8/Detail?sdb=CJFD&sfield=%e4%bd%9c%e8%80%85&skey=%e8%88%92%e6%96%b0%e5%b3%b0&scode=39111263&acode=39111263);[王小银](https://kns.cnki.net/KNS8/Detail?sdb=CJFD&sfield=%e4%bd%9c%e8%80%85&skey=%e7%8e%8b%e5%b0%8f%e9%93%b6&scode=39111305&acode=39111305);[陈锐](https://kns.cnki.net/KNS8/Detail?sdb=CJFD&sfield=%e4%bd%9c%e8%80%85&skey=%e9%99%88%e9%94%90&scode=22002880&acode=22002880). [自动代码生成技术的发展现状与趋势](https://kns.cnki.net/KNS8/Detail?sfield=fn&QueryID=5&CurRec=1&recid=&FileName=XAYD201803001&DbName=CJFDLAST2018&DbCode=CJFD&yx=&pr=&URLID=). [西安邮电大学学报](https://kns.cnki.net/KNS8/Navi?DBCode=CJFD&BaseID=XAYD)[J]
2. 张艳伟,陈正鸣,吕嘉,张霞. 主流代码生成方法的分析与比较[J]. 计算机与现代化,2015(11):103-108.
3. 李陈军. 基于MongoDB的SNS平台设计与实现[D].南京理工大学,2013.
4. 牛岩.作物产量指标综合评价的数据标准化处理[J].农村经济与科技,2017,28(19):16-19.
5. 陈家伟. 支持向量机在人体健康状态预测中的研究与应用[D].中国科学技术大学,2014.
6. 任禹丞,徐超,赵磊,贾静,彭路,周子馨.基于自适应特征权重聚类算法的用电问题分析[J].计算机系统应用,2020,29(01):29-39.
7. 鲁志军,邱雪涛,赵金涛.数据挖掘在交易欺诈侦测中的应用[J].软件产业与工程,2014(05):24-28+38.
8. 刘超. 业界专家的媒体发言对公司股价影响的分析[D].上海师范大学,2016.
9. 潘抱一. 基于动态程序失效执行块的软件错误定位方法研究[D].浙江理工大学,2020.
10. 徐京京. 基于聚类和神经网络的异常数据识别算法研究[D].华北电力大学(北京),2019.

# 致 谢

从17年的秋天进入大学，学会输出第一行“Hello,World”起，便开始了我对计算机世界的探索。经过这四年的学习、研究、探索，对课本教材知识点，以及时下技术热点的理解，想通过毕业设计这样一个途径，以独立完成这样一个完整的代码生成系统的方式，来表达老师们这几年来的悉心栽培的感激之情。短短四年时间，是老师们实验课上一次次断点调试的帮助，是课堂上上至对编程架构的讲解，对前沿科技的深刻见解，下至内存中每个字节的存放，机器语言对1和0 的每一次编排让我被软件工程，数据科学这些技术的优雅的实现方式所折服。此外，也是每一次竞赛，每一次跟着老师做项目的经历，让我对那些教材上看似简短，实则凝练至极的概念有了深层次的领悟。

其次，要感谢我的父母，对我求学路上每一步的鼓励与肯定，是我最坚强的后盾。感谢一起同窗四年的室友，是课堂上共同进步，课下一起研究技术，在数不清多少个熄灯的夜晚，一起攻克学习新知识的瓶颈，完成了一次次自我的突破。是假期一起旅游，游览大江大河，回到学校想通过更加努力的学习去见识更加广阔的世界。

最后我要感谢母校的栽培，给我提供宝贵的学习资源，以及这样一个广阔的平台感谢数计学院。这些经历都是我人生中宝贵的财富，是弥足珍贵的纪念品，好好收起，仍热满怀热情，继续前行。