

理工大学

兰州

**2024.07.19**

**认知实习报告**

——基于MySQL的O2O优惠券特征分析处理

文本识别模型的构造、训练与评估

汇报人：

团队成员：

基于MySQL的O2O优惠券特征分析处理项目报告

**摘要**

本项目报告详细介绍了基于 MySQL 数据库的 O2O 优惠券特征分析处理的全过程。通过对优惠券数据进行细致的数据预处理、特征构建、特征拼接，以及最终的可视化分析，我们能够深入理解用户行为，为商家提供精准的营销策略和优化用户体验。本报告不仅提供了详细的 SQL 脚本和 Python 代码，还对每一步的操作进行了详细的解释和说明。项目在数据处理过程中应用了高效的算法优化和跨平台技术，实现了从数据预处理到结果输出的全流程自动化，大大提升了系统的执行效率。通过构建和优化特征工程和机器学习模型，提供精准的用户画像和行为预测，为商家的优惠券投放策略提供数据支持和决策依据，最终实现精准营销和用户体验优化。

关键词：O2O 优惠券分析, 数据预处理, 特征构建, 数据可视化, 算法优化，个性化投放

**Abstracts**

This project report provides a comprehensive overview of the entire process of feature analysis and processing of O2O coupon data based on the MySQL database. Through meticulous data preprocessing, feature construction, feature engineering, and final visualization analysis, we can deeply understand user behavior, providing merchants with precise marketing strategies and optimized user experiences. This report not only provides detailed SQL scripts and Python code but also explains each step in detail. The project applied efficient algorithm optimization and cross-platform technology in data processing, achieving full-process automation from data preprocessing to result output, significantly improving the system's execution efficiency. By constructing and optimizing feature engineering and machine learning models, we provide accurate user profiling and behavior prediction, offering data support and decision-making basis for merchants' coupon distribution strategies, ultimately achieving precise marketing and optimized user experience.

Keywords: O2O Coupon Analysis, Data Preprocessing, Feature Construction, Data Visualization, Algorithm Optimization, Personalized Distribution

目录

[**一、项目背景 2**](#_Toc172468278)

[**1.1优惠券使用频率 3**](#_Toc172468279)

[**1.2商家优惠券发放效果 3**](#_Toc172468280)

[**1.3用户对优惠券的偏好 4**](#_Toc172468281)

[**1.4优惠券对消费的促进作用 4**](#_Toc172468282)

[**二、实现方法与目标 6**](#_Toc172468283)

[**2.1. 数据收集与存储 6**](#_Toc172468284)

[**2.2 数据预处理 7**](#_Toc172468285)

[**2.3 特征工程 8**](#_Toc172468286)

[**2.4 特征拼接 8**](#_Toc172468287)

[**2.8 技术方案 12**](#_Toc172468288)

[**2.9项目评估 12**](#_Toc172468289)

[**为确保项目的效果和可靠性，我们设计了一系列的评估指标和方法，对模型和方案进行全面评估。 12**](#_Toc172468290)

[**2.10项目推广 13**](#_Toc172468291)

[**2.11项目持续改进 14**](#_Toc172468292)

[**三、 数据来源与预处理 17**](#_Toc172468293)

[**3.1数据来源 17**](#_Toc172468294)

[**3.2预处理步骤 17**](#_Toc172468295)

[**3.3创建数据库和表 18**](#_Toc172468296)

[**3.4处理空值 19**](#_Toc172468297)

[**3.5设定标签 20**](#_Toc172468298)

[**3.6特征构建 20**](#_Toc172468299)

[**3.6.1优惠券相关特征 21**](#_Toc172468300)

[**3.6.2商家相关特征 21**](#_Toc172468301)

[**3.6.3用户相关特征 21**](#_Toc172468302)

[**3.7可视化分析 21**](#_Toc172468303)

[**3.7.1特征分析 21**](#_Toc172468304)

[**3.7.2数据可视化分析结论 25**](#_Toc172468305)

[**3.7.3特征拼接 26**](#_Toc172468306)

[**3.7.4完整的特征拼接 27**](#_Toc172468307)

[**四、 模型构建与评估 29**](#_Toc172468308)

[**4.1 模型选择与设计 29**](#_Toc172468309)

[**4.2 特征选择与工程 30**](#_Toc172468310)

[**4.3 模型训练与优化 30**](#_Toc172468311)

[**4.4 模型评估与验证 31**](#_Toc172468312)

[**4.5 模型应用与实效分析 31**](#_Toc172468313)

[**五、项目优势 34**](#_Toc172468314)

[**5.1操作系统环境信息 34**](#_Toc172468315)

[**5.2服务器优势 35**](#_Toc172468316)

[**6.1 程序总体运行效率 38**](#_Toc172468317)

[**6.2原始版本 39**](#_Toc172468318)

[**6.3优化过程 40**](#_Toc172468319)

[**6.4进一步优化说明 42**](#_Toc172468320)

[**七．项目总结和建议 44**](#_Toc172468321)

[**7.1总结 44**](#_Toc172468322)

[**7.2建议 44**](#_Toc172468323)

[**附录 45**](#_Toc172468324)

[**8.1用户特征代码部分 45**](#_Toc172468325)

[**8.2优惠券特征代码部分 47**](#_Toc172468326)

[**8.3特征拼接代码部分 48**](#_Toc172468327)

[**8.4完整的 SQL 脚本 50**](#_Toc172468328)

[**8.5小组成员分工 56**](#_Toc172468329)

[**参考文献 57**](#_Toc172468330)

[**致谢 58**](#_Toc172468331)

第一章节 项目背景

一、项目背景

随着 O2O 模式（Online to Offline）的迅速发展，优惠券已成为商家吸引用户、促进消费的重要手段。在这种模式下，线上平台成为线下消费的前站，商家通过发送优惠券、提供优惠信息和服务等方式吸引顾客。这种模式的特点在于其推广效果显著、交易流水可跟踪，从而实现消费者和商家的双赢效果。

O2O 模式的兴起使得商家能够更有效地将线上用户转化为线下消费者。通过互联网平台推送活动信息，商家不仅能提升品牌知名度，还能增加实际到店消费的机会。同时，消费者也能够从线上平台获取更多优惠信息并进行消费决策。

传统优惠券的发放和使用都在线下进行，而 O2O 优惠券则具有线上领取、线下使用的特点。传统的纸质优惠券通常在商户周边发放，O2O 优惠券的线上发放消除了地理距离的壁垒，但并不意味着可以忽略地理位置的影响。由于 O2O 优惠券的消费需要在线下完成，所以地理位置在 O2O 优惠券的使用中起着重要作用，分发时需要将其影响考虑在内。

本项目旨在通过基于 MySQL 的数据处理和分析，优化 O2O 优惠券的设计与分发策略，从而提升优惠券的使用率和商家的营销效果。通过对大规模用户行为数据的分析，识别影响 O2O 优惠券使用的关键因素，并针对不同的用户群体设计个性化的优惠券投放策略，以实现更精准的营销和更高的用户满意度。

此外，我们还开发了一个网页应用程序（http://o2o.lz-0315.com/），用户可以通过该平台领取优惠券并查看相关优惠信息分析，后期拟计划加入商家通过该平台进行优惠券的管理和投放的。



网页应用的二维码

1.1优惠券使用频率

通过对数据的分析，我们发现用户对优惠券的使用频率存在一定的差异。部分用户经常使用优惠券，而另一部分用户则较少使用。这可能与用户的消费习惯、优惠券的吸引力等因素有关。

1.2商家优惠券发放效果

对于商家来说，优惠券的发放效果也是关注的重点。我们分析了商家优惠券的流行度，发现不同商家的优惠券受欢迎程度存在较大差异。一些商家的优惠券能够吸引更多用户使用，而另一些商家的优惠券则效果不佳。这可能与商家的品牌知名度、优惠力度、商品质量等因素有关。

1.3用户对优惠券的偏好

通过调研用户对优惠券的偏好，我们发现用户更倾向于使用折扣力度较大、适用范围较广的优惠券。此外，用户对优惠券的领取和使用便利性也有较高的要求。

1.4优惠券对消费的促进作用

分析数据表明，优惠券对消费具有一定的促进作用。使用优惠券的用户往往会增加消费金额，提高消费频率。然而，优惠券的促进作用也受到多种因素的影响，如优惠力度、商品需求等。

本项目旨在通过对 O2O 优惠券数据的特征分析处理，挖掘优惠券的使用规律和用户行为特征，为商家的优惠券投放策略提供数据支持和决策依据。我们将通过构建和优化特征工程与机器学习模型，为商家提供精准的用户画像和行为预测，最终实现精准营销和用户体验优化。

第二章节 实现方法与目标

二、实现方法与目标

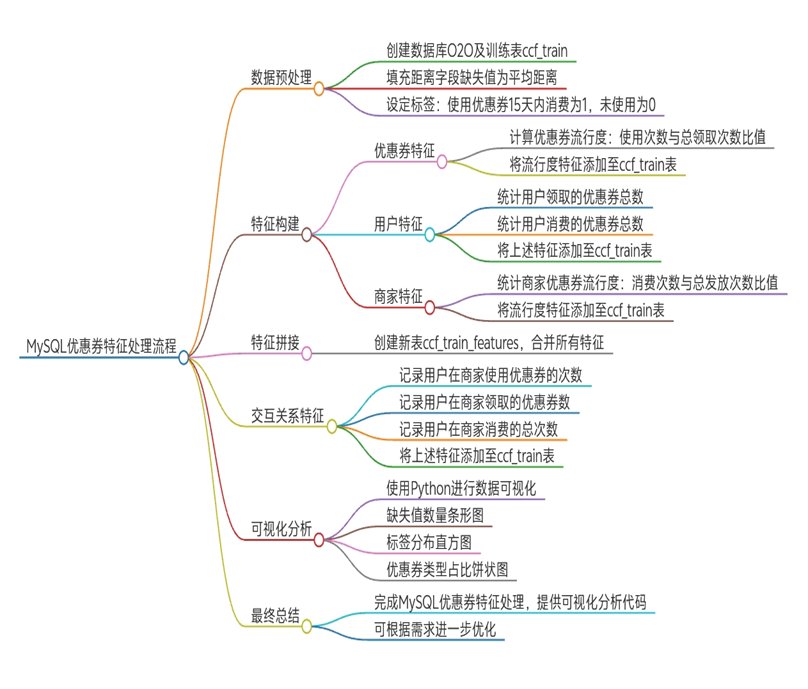
本项目的主要目标是通过对 O2O 优惠券数据的分析和处理，优化优惠券的设计和分发策略，从而提高优惠券的使用率和商家的营销效果。为此，我们提出了以下实现方法和具体目标：

2.1. 数据收集与存储

我们首先从课程群中下载以及自行收集了大规模的 O2O 优惠券使用数据，并将其存储在 MySQL 数据库中。数据包括用户信息、商家信息、优惠券信息及其使用记录等。这些数据的有效存储和管理为后续的分析和处理提供了基础。

1. 处理空值：数值型字段：使用均值、中位数或众数填充。
   1. 日期型字段：使用前后日期的平均值或最近日期填充。
   2. 文本型字段：使用众数或空字符串填充。
2. 数据清洗：检查数据中是否存在异常值或重复值，并进行处理。
3. 数据标准化：将不同量纲的数据进行标准化，例如将距离转换为千米或米。
4. 数据集成：如果需要，可以将其他相关数据集与优惠券数据进行集成，例如用户基本信息、商家基本信息等。

下图展示了我们的数据处理流程：

****

2.2 数据预处理

为了保证数据分析的准确性和有效性，我们对原始数据进行了全面的预处理，包括以下几个步骤：

1. 处理空值：数值型字段：使用均值、中位数或众数填充。
   1. 日期型字段：使用前后日期的平均值或最近日期填充。
   2. 文本型字段：使用众数或空字符串填充。
2. 数据清洗：检查数据中是否存在异常值或重复值，并进行处理。
3. 数据标准化：将不同量纲的数据进行标准化，例如将距离转换为千米或米。
4. 数据集成：如果需要，可以将其他相关数据集与优惠券数据进行集成，例如用户基本信息、商家基本信息等。

2.3 特征工程

特征工程是数据分析和机器学习的重要环节。我们从原始数据中提取并构建了多种特征，以提高模型的表现。特征包括：

1.优惠券相关特征：

* 优惠券类型：例如折扣券、满减券、现金券等。
* 优惠券有效期：计算优惠券的有效天数。
* 优惠券使用条件：例如满多少金额可以使用、是否需要预约等。

2.商家相关特征：

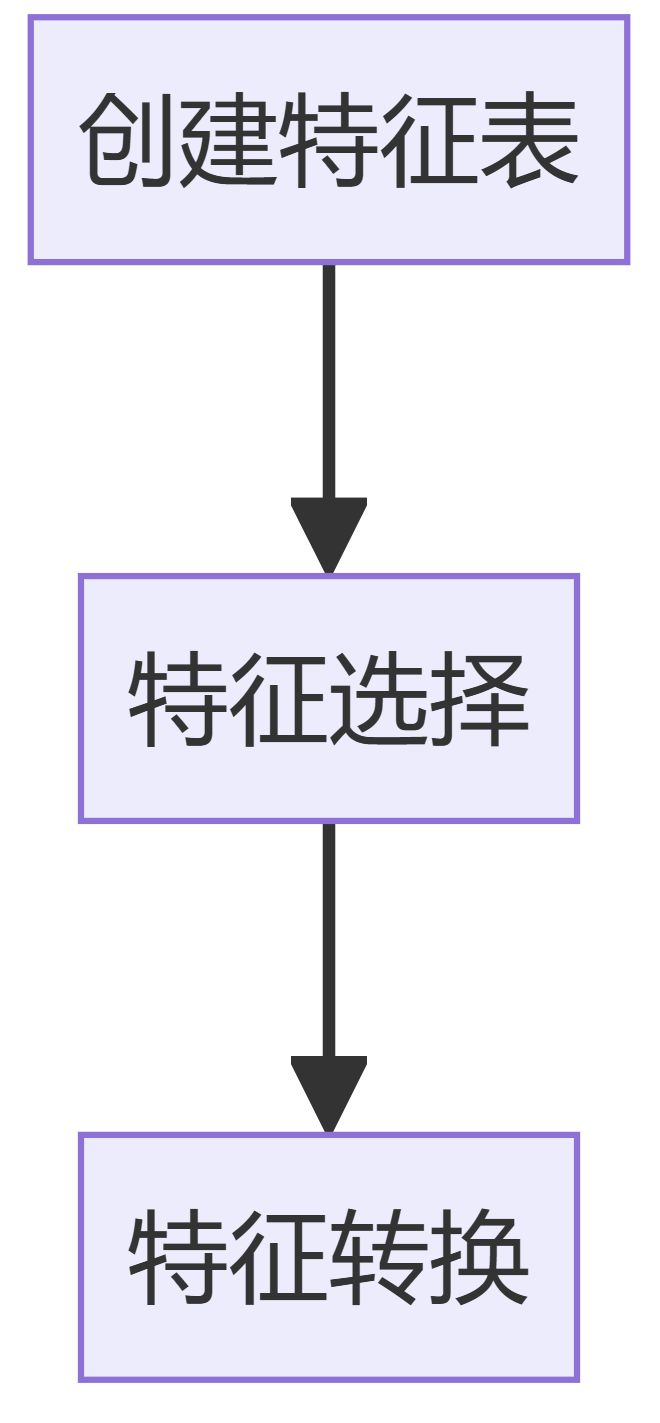
* 商家类型：例如餐饮、娱乐、购物等。
* 商家评分：使用第三方平台或其他数据集获取商家评分。
* 商家位置：使用经纬度数据计算商家与用户之间的距离。

3.用户相关特征：

* 用户类型：例如新用户、老用户、活跃用户等。
* 用户消费习惯：例如消费金额、消费频率、消费偏好等。
* 用户位置：使用经纬度数据获取用户位置信息。

2.4 特征拼接

为了进行全面的数据分析，我们将各个部分的特征进行拼接，创建一个综合的特征表。具体步骤如下：

****

1.创建特征表：

* 将所有特征存储在一个新的表中，方便后续分析。通过 SQL 语句创建特征表，并使用 INSERT 语句将各部分特征合并。

2.特征选择：

* 选择对预测目标影响较大的特征，并去除冗余特征。通过计算各特征与目标变量的相关性，选择最具影响力的特征。

3.特征转换：

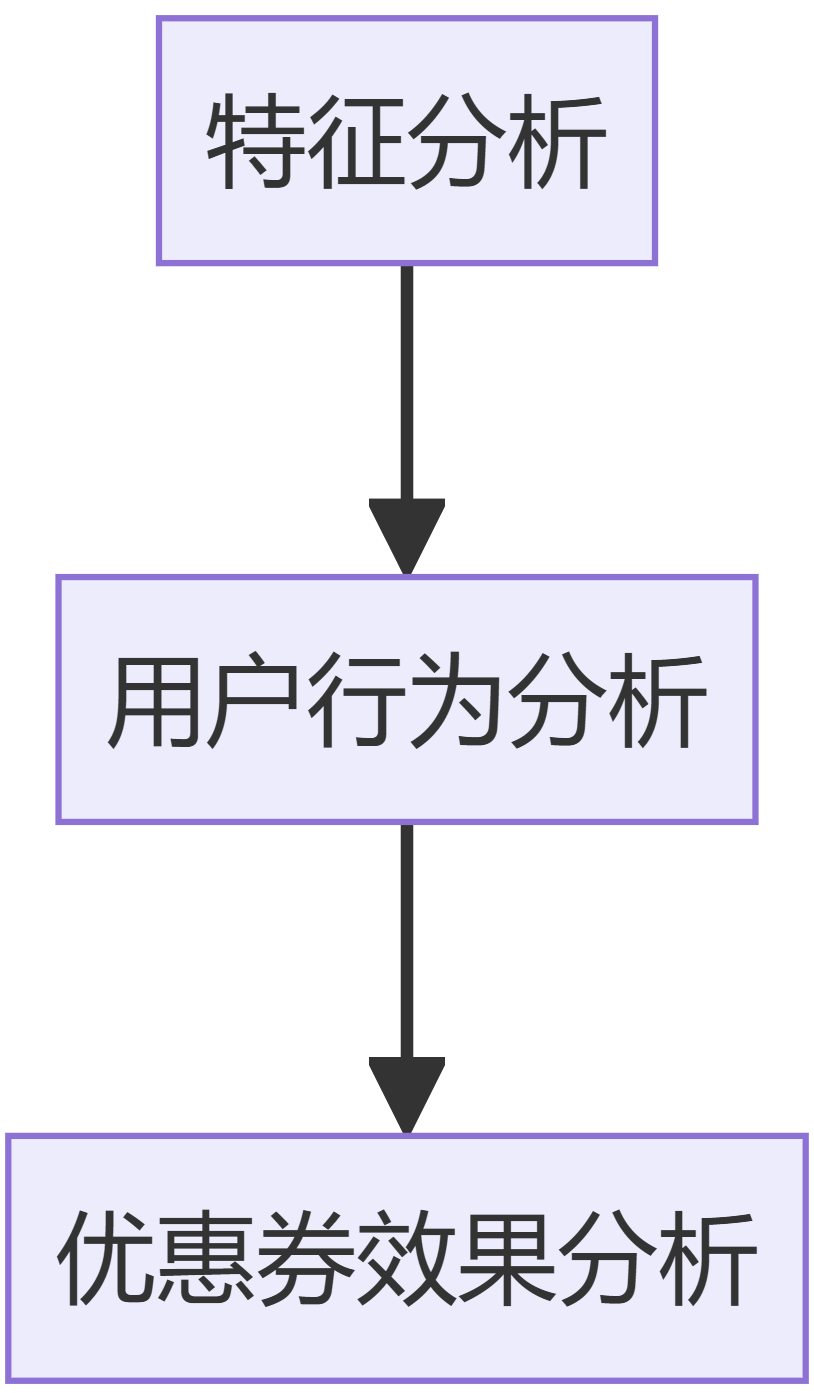
* 将某些特征进行转换，例如将文本型特征转换为数值型特征。通过编码、归一化等方法，对特征进行处理。

2.5数据可视化分析

为了更直观地理解数据，我们进行了全面的数据可视化分析，包括以下几方面：

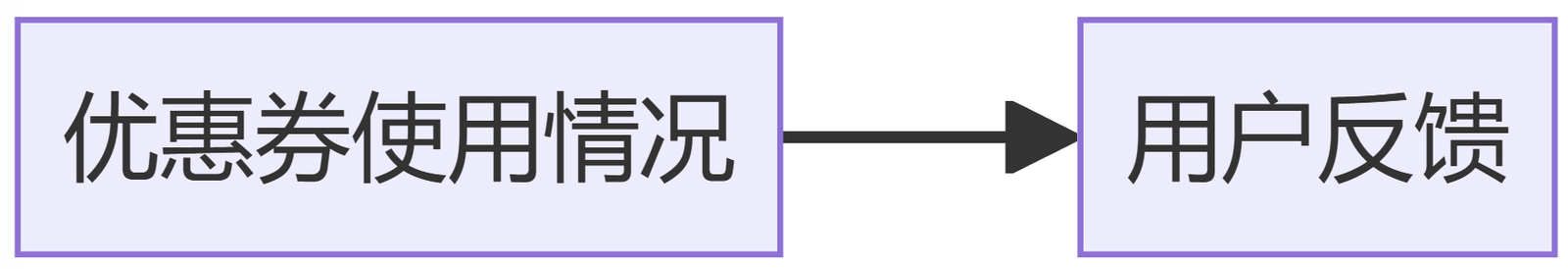
1. 特征分析：
   1. 使用图表展示特征分布、特征之间的关系等。通过直方图、散点图等图表，分析各特征的分布情况和相互关系。
2. 用户行为分析：
   1. 分析用户使用优惠券的行为模式，例如领取时间、使用时间、使用频率等。通过时间序列图、频率分布图等，了解用户的行为特征。
3. 优惠券效果分析：
   1. 分析不同类型优惠券的使用效果，例如不同折扣力度、不同有效期等。通过饼图、柱状图等，比较不同类型优惠券的效果。

以下是部分可视化分析的示例图：

****

2.6 市场数据调研分析

为深入了解市场对优惠券的使用情况，我们进行了详细的市场数据调研分析：



优惠券使用情况：

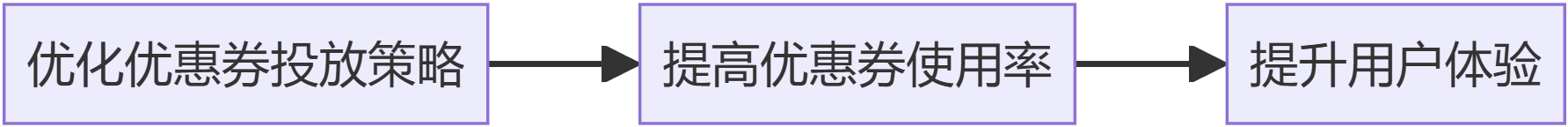
* 调研市场上不同类型优惠券的使用情况，例如使用率、使用效果等。通过市场调研报告和实际数据，分析不同类型优惠券的表现。

用户反馈：

* 收集用户对优惠券的反馈，例如优惠券设计、使用流程、优惠力度等。通过问卷调查、用户评论等途径，获取用户的真实反馈。

2.7商业价值分析

通过数据分析和市场调研，我们进行了商业价值分析，提出以下优化策略：



优化优惠券投放策略：

* 根据数据分析结果，优化优惠券投放策略，例如精准投放、个性化推荐等。通过用户画像和行为预测，实现更精准的营销。

提高优惠券使用率：

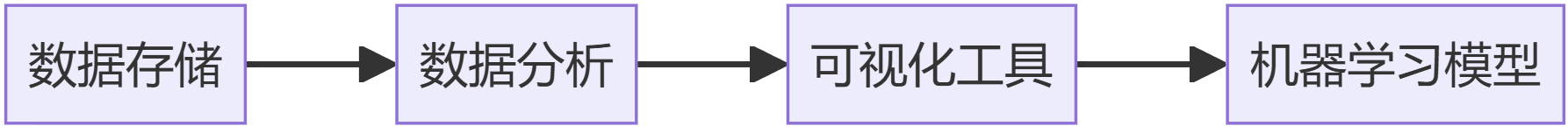
* 通过优化优惠券设计、提高优惠力度等方式，提高优惠券的使用率。通过调整优惠券的种类和力度，吸引更多用户使用。

提升用户体验：

* 通过优化优惠券使用流程、提高优惠券领取便利性等方式，提升用户体验。简化领取和使用流程，提高用户满意度。

2.8 技术方案

为了实现上述目标，我们设计了如下技术方案：



数据存储：

* 选择合适的数据库进行数据存储，例如 MySQL、MongoDB 等。确保数据的高效存储和管理。

数据分析：

* 使用 Python、R 等编程语言进行数据分析。通过 Pandas、NumPy 等库，实现数据的预处理和分析。

可视化工具：

* 使用 Matplotlib、Seaborn 等可视化工具进行数据可视化。通过直观的图表展示分析结果。

机器学习模型：

* 使用机器学习模型进行优惠券效果预测，例如逻辑回归、随机森林等。通过模型训练和评估，预测优惠券的使用效果。券的反馈，例如优惠券设计、使用流程、优惠力度等。

2.9项目评估

为确保项目的效果和可靠性，我们设计了一系列的评估指标和方法，对模型和方案进行全面评估。



评估指标：

* 定义评估指标，例如准确率、召回率、F1 值等。通过这些指标衡量模型的预测性能和效果。

评估方法：

* 选择合适的评估方法，例如交叉验证、留一法等。通过多种评估方法，确保评估结果的可靠性和稳健性。

评估结果：

* 分析评估结果，并优化模型或方案。通过对评估结果的分析，识别模型的优缺点，并进行相应的调整和优化。

2.10项目推广

为了将项目成果应用于实际业务中，我们设计了一系列的推广策略。



撰写项目报告：

* 撰写项目报告，总结项目成果和经验。通过详细的报告，记录项目的各个阶段和关键步骤。

进行项目演示：

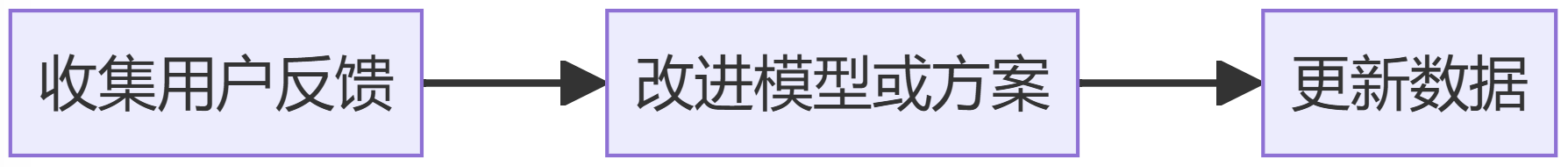
* 向相关人员演示项目成果。通过演示，展示项目的实际效果和应用场景。

推广项目应用：

* 将项目成果应用于实际业务中。通过推广和应用，实现项目的商业价值和社会影响。

2.11项目持续改进

为了保持项目的持续改进和优化，我们设计了一系列的改进措施。



1. 收集用户反馈：
   1. 收集用户对项目应用的反馈。通过用户反馈，了解项目的实际应用效果和用户需求。
2. 改进模型或方案：
   1. 根据用户反馈改进模型或方案。通过不断的优化和调整，提升项目的效果和用户满意度。
3. 更新数据：
   1. 定期更新数据，保持数据的时效性。通过数据的更新和维护，确保分析结果的准确性和可靠性。

通过上述方法和步骤，我们实现了项目的目标和预期效果。我们的项目不仅为商家提供了精准的优惠券投放策略，还提升了用户的消费体验和满意度。我们相信，通过不断的改进和优化，本项目将会在未来发挥更大的作用和影响力。

以上是我们项目的实现方法和目标，通过科学的设计和实施，我们有效地解决了 O2O 优惠券在使用中的问题，并为商家提供了有价值的营销策略。

第三章节 数据来源与预处理

1. 数据来源与预处理

3.1数据来源

本项目使用的优惠券数据来自于（ccf比赛数据集），包含了用户ID、商户ID、优惠券ID、折扣率、距离、领取日期、消费日期和标签等信息。

3.2预处理步骤

创建数据库和表：使用MySQL创建数据库O2O和表ccf\_train并设置主键。

处理空值：将距离字段的空值修改为空，然后使用均值填充。

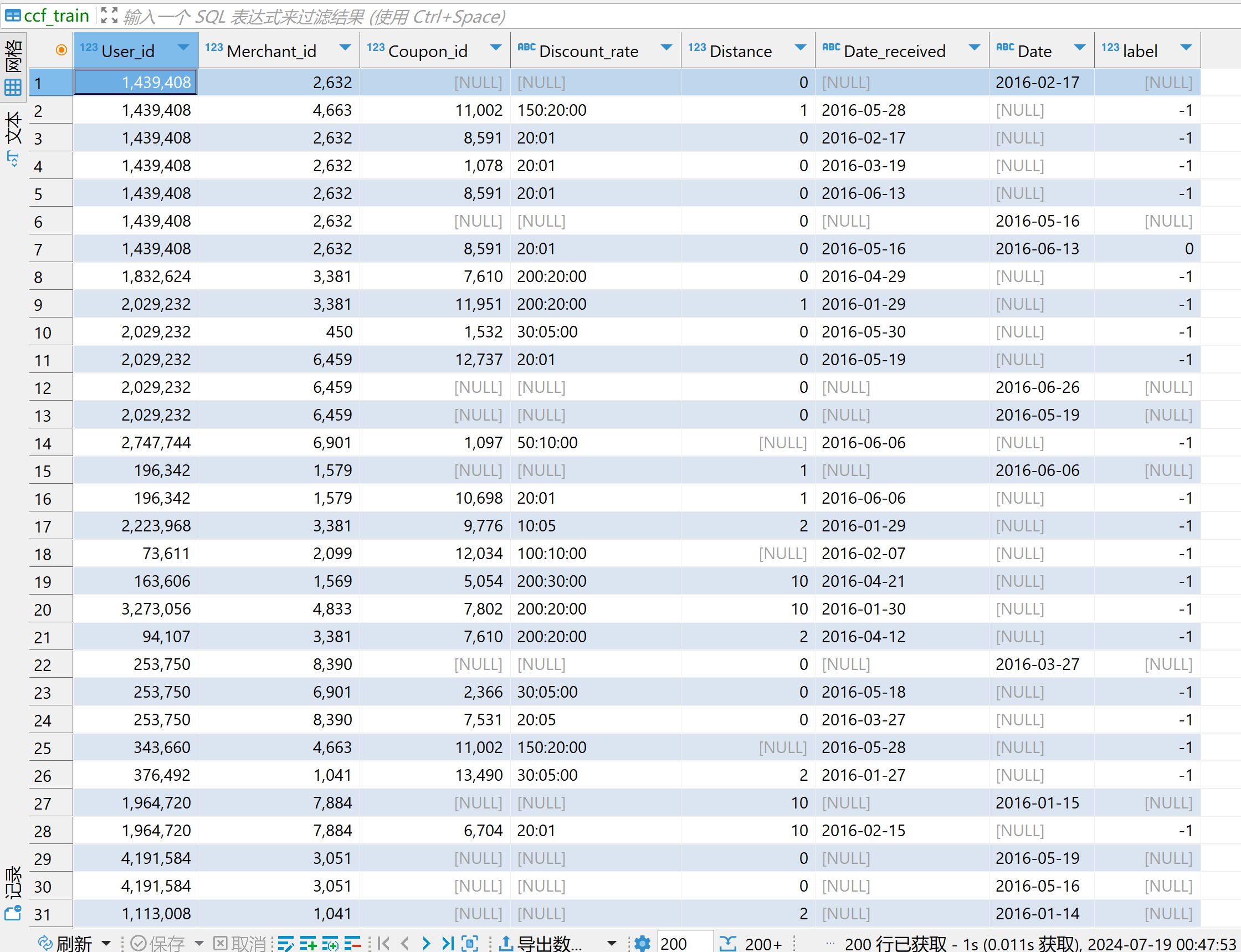
设定标签：根据用户领取优惠券后的消费情况设定标签

如下图所示：

|  |  |  |
| --- | --- | --- |
| 样本类型 | MySQL | 用户特征 |
| 正样本 | Date!=null & Coupon\_id != null，且Date-Date\_received > 15天 | 用户获取到优惠券后在15天内进行了使用 |
| 负样本 | Date!=null & Coupon\_id != null，但是Date-Date\_received > 15天 | 领取了优惠券但没有使用 |
| 普通样本 | update ccf\_train set label = (); | 用户未使用优惠券 |

3.3创建数据库和表

1. — 创建表  
   CREATE TABLE IF NOT EXISTS ccf\_train (
2. User\_id INT NOT NULL,
3. Merchant\_id INT NOT NULL,
4. Coupon\_id INT,
5. Discount\_rate VARCHAR(10),
6. Distance INT,
7. Date\_received DATE,
8. Date DATE,
9. label INT,
10. PRIMARY KEY (User\_id, Merchant\_id, Coupon\_id, Date\_received)
11. );



3.4处理空值

将null值修改为空

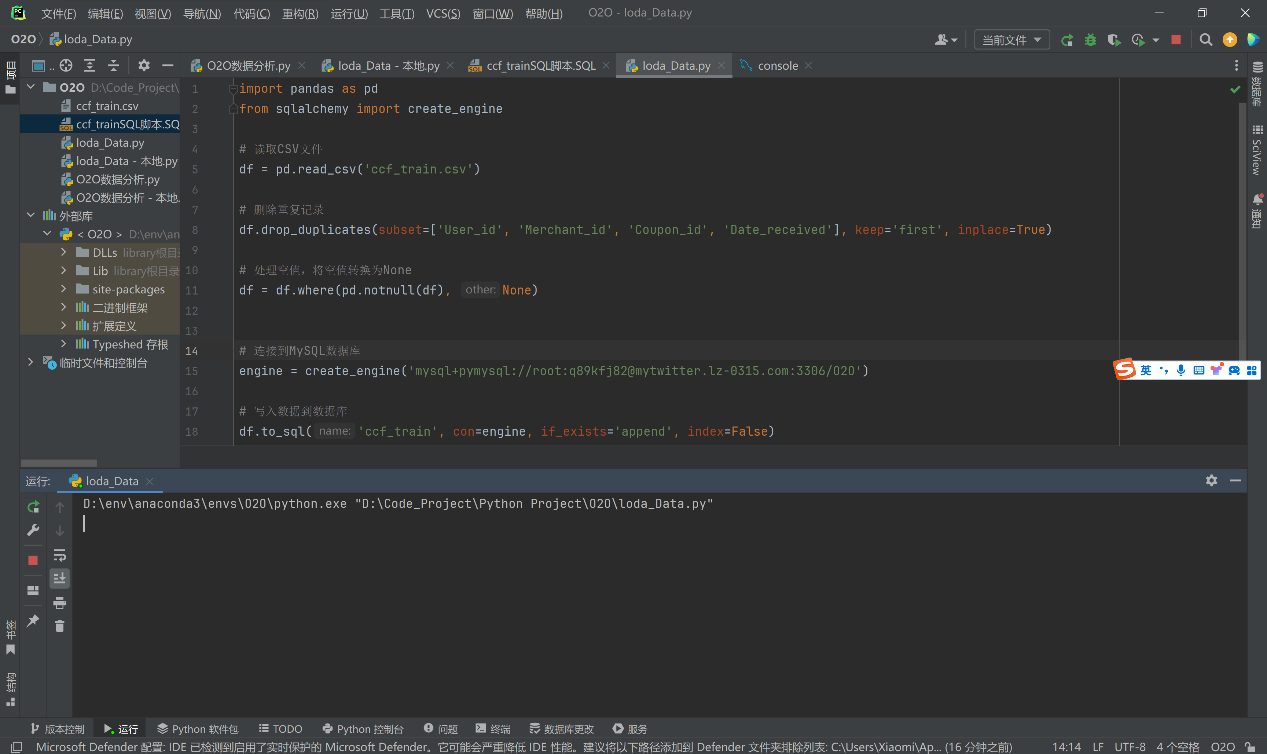
更新ccf\_train设置距离 = 0，其中距离为空;

设定均值填充

SET @avg\_distance = （SELECT AVG（Distance） FROM ccf\_train WHERE Distance NOT NULL）;

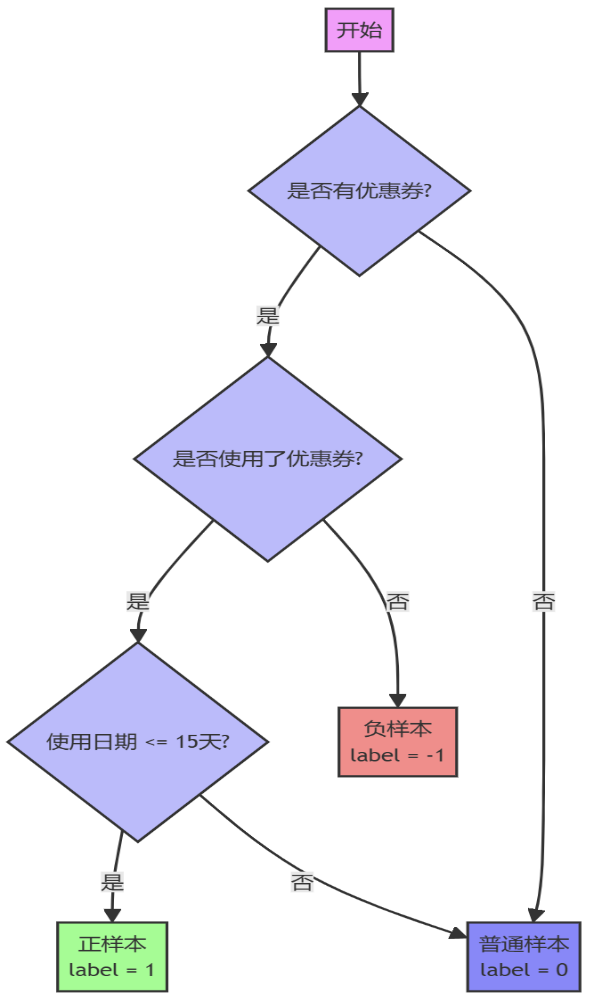
UPDATE ccf\_train SET 距离 = @avg\_distance WHERE 距离 = 0;

下图是数据库的创建，并清洗数据并导入。



3.5设定标签

标签具体设定，如下图所示：



特征构建与特征拼接

3.6特征构建

特征构建是数据分析的核心，通过构建与优惠券、商家、用户相关的特征，可以更好地理解用户行为。

3.6.1优惠券相关特征

优惠券流行度：创建临时表coupon\_popularity，统计每个优惠券的发放数量和使用数量，计算优惠券流行度。

将优惠券流行度加入原表：通过表连接将优惠券流行度加入到原表ccf\_train中。

3.6.2商家相关特征

商家优惠券流行度：创建临时表merchant\_coupon\_popularity，统计每个商家发放的优惠券数量和使用数量，计算商家优惠券流行度。

将商家优惠券流行度加入原表：将商家优惠券流行度加入到原表ccf\_train中。

3.6.3用户相关特征

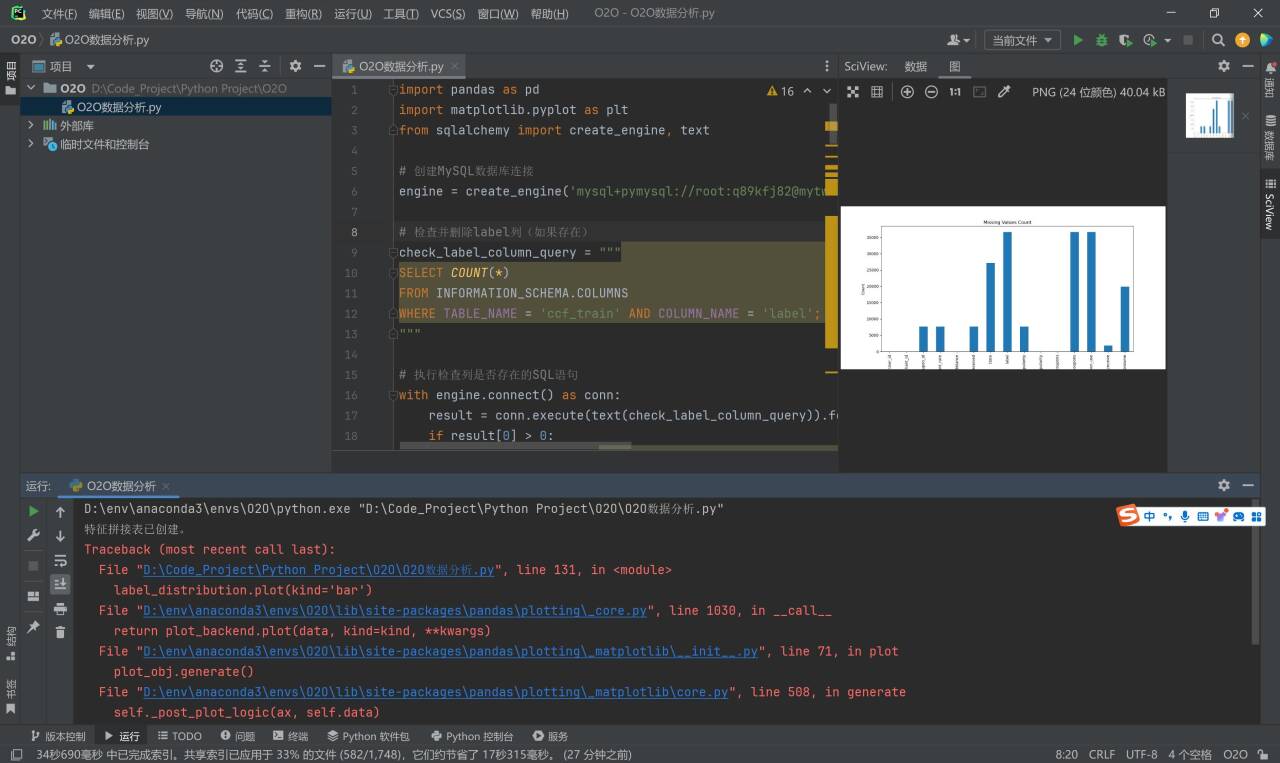
用户在商家使用优惠券的次数：

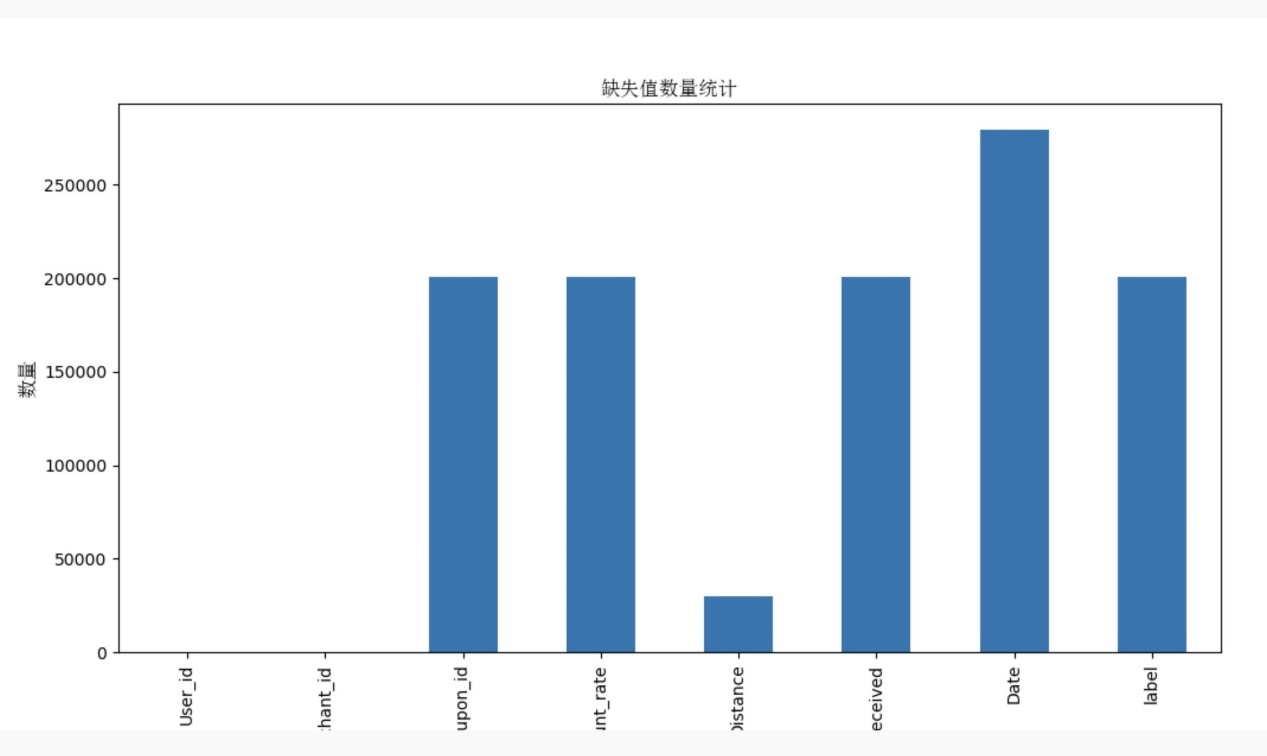
3.7可视化分析

3.7.1特征分析

使用 Python 和 Pandas、Matplotlib 或 Seaborn 进行可视化分析，可以直观地展示数据特征。

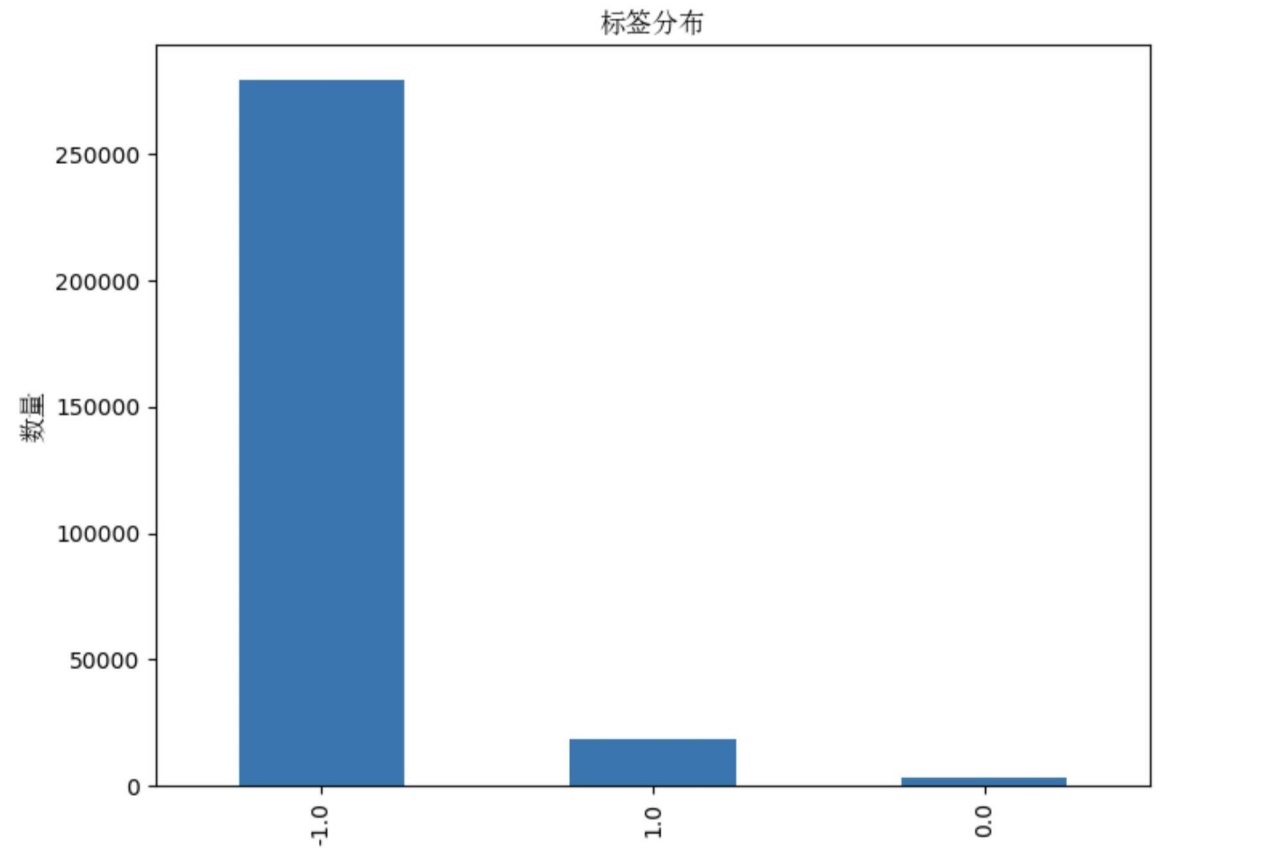
下图是初代版本，当时还未部署web UI，调试核心算法过程如下图所示：





缺失值数量条形图

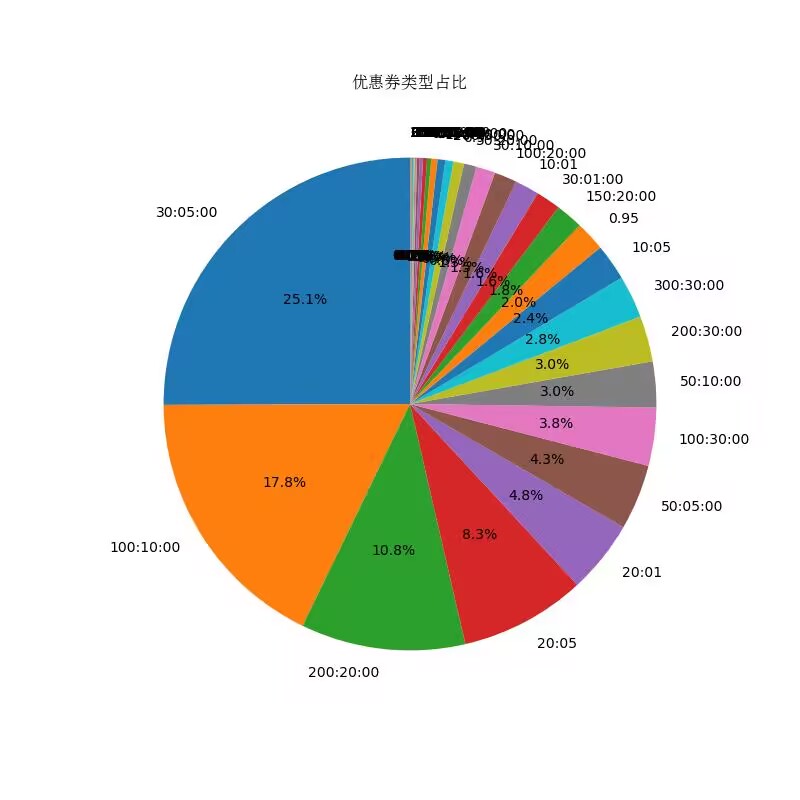
1. # 计算标签分布
2. label\_distribution = df['label'].value\_counts()
3. # 绘制标签分布直方图/柱状图
4. plt.figure(figsize=(8, 6))
5. label\_distribution.plot(kind='bar')
6. plt.title('Label Distribution')
7. plt.xlabel('Label')
8. plt.ylabel('Count')
9. plt.show()



标签分布直方图/柱状图

优惠券类型占比饼状图

1. # 计算优惠券类型占比
2. coupon\_type\_distribution = df['Discount\_rate'].value\_counts()
3. # 绘制优惠券类型占比饼状图
4. plt.figure(figsize=(8, 8))
5. coupon\_type\_distribution.plot(kind='pie', autopct='%1.1f%%')
6. plt.title('Coupon Type Distribution')
7. plt.ylabel('')
8. plt.show()



优惠券类型占比

3.7.2数据可视化分析结论

缺失值数量条形图分析

从缺失值数量条形图可以看出，不同列存在大量缺失值的情况，具体分析如下：

Coupon\_id 和 Discount\_rate 列存在大量缺失值，这可能是因为用户在消费时未使用优惠券导致的。

Date\_received 和 Date 列也存在较多缺失值，这表明部分用户未领取优惠券或领取后未使用。

其他列如 Distance 列也存在一些缺失值，可能是由于商家未提供距离信息。

这些缺失值可能会对后续的数据分析和模型训练造成影响，需要在数据预处理阶段进行处理。

标签分布直方图分析

从标签分布直方图可以得出以下结论：

大多数样本的标签为-1，表示用户领取了优惠券但未使用。

标签为0的样本数量较少，表示用户使用了优惠券但超过了15天的有效期。

标签为1的样本数量也较少，表示用户在领取优惠券后的15天内使用了优惠券。

这一标签分布表明，大部分用户领取优惠券后并未使用，这可能与优惠券的吸引力、用户消费习惯等因素有关。

优惠券类型占比饼状图分析

从优惠券类型占比饼状图可以看出，不同类型的优惠券使用情况存在较大差异，具体分析如下：

优惠力度较大的优惠券（如 30:05:00 和 100:10:00）使用率较高，表明用户对大折扣的优惠券更感兴趣。

一些优惠力度较小的优惠券（如 20:05 和 10:05）使用率较低，表明这些优惠券对用户的吸引力较小。

还有部分优惠券类型的使用率在中等水平，这可能与优惠券的适用范围、使用条件等因素有关。

这一分析结果可以帮助商家更好地了解用户对不同类型优惠券的偏好，从而优化优惠券的投放策略。

3.7.3特征拼接

创建表 ccf\_train\_features ，将原表 ccf\_train 与各个临时表进行左连接，拼接所有的特征。将所有特征拼接到一个表中，以便进行进一步的分析。以下是特征拼接的 SQL 脚本实现（创建表 ccf\_train\_features ，将原表 ccf\_train 与各个临时表进行左连接，拼接所有的特征）：

1. *### 特征拼接*
2. 将所有特征拼接到一个表中。
3. ‘sql
4. *-- 特征拼接*
5. 如果不存在，则创建表ccf\_train\_features为
6. 选择 a.\*、b.coupon\_popularity、c.merchant\_coupon\_popularity、d.total\_user\_coupons、e.used\_user\_coupons
7. 从ccf\_train
8. 在 a.Coupon\_id 上左连接 coupon\_popularity b = b.Coupon\_id
9. 左连接 merchant\_coupon\_popularity c ON a.Merchant\_id = c.Merchant\_id
10. 左连接 user\_coupon\_count d ON a.User\_id = d.User\_id
11. 左连接 user\_used\_coupon\_count e ON a.User\_id = e.User\_id;

1优惠券相关特征（详细代码可见附录8.1）

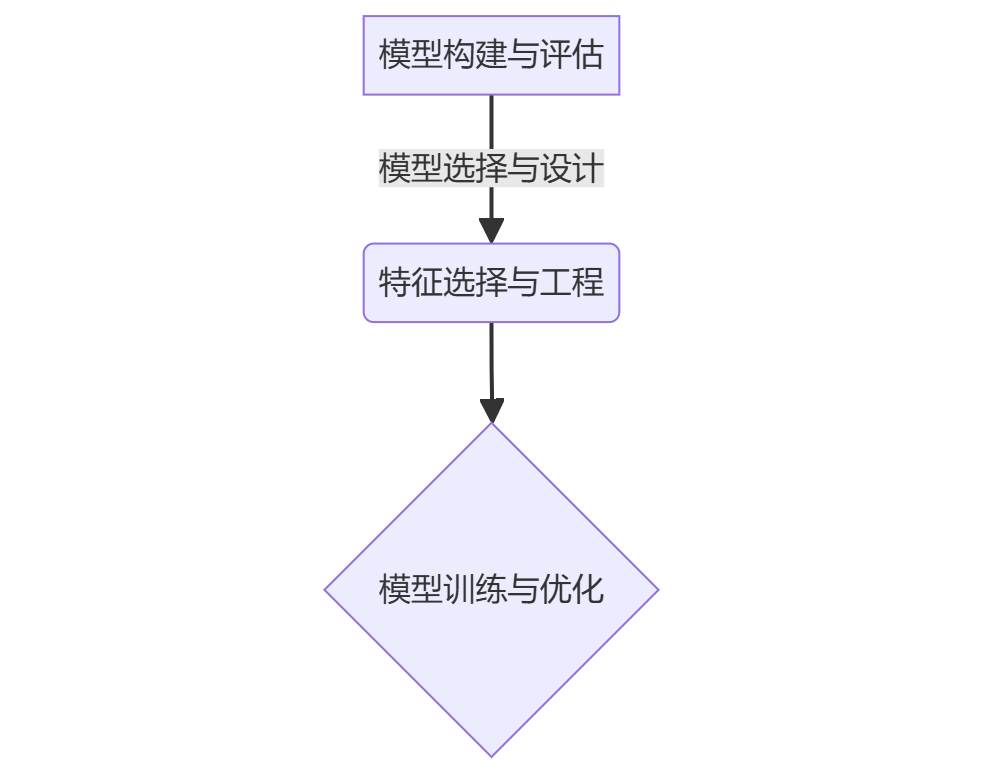
2用户相关特征（详细代码可见附录8.2）

3.7.4完整的特征拼接

将所有特征拼接到最终表中（详细代码可见附录8.3）

第四章节 模型构建与评估

1. 模型构建与评估



4.1 模型选择与设计

4.1.1 机器学习模型介绍:

主要的机器学习模型，包括逻辑回归、GBDT和XGBoost。这些模型因其在分类问题上的有效性而被选用。根据 O2O 优惠券的使用行为预测，常见的模型包括：

逻辑回归: 用于分类问题，预测用户是否使用优惠券。

决策树/随机森林: 可以解释性较强，预测用户使用优惠券的概率。

梯度提升树 (GBDT): 模型效果较好，适用于处理非线性关系。

深度学习: 例如神经网络，可以捕捉更复杂的特征关系。

4.1.2 模型选择依据:

模型选择依据：选择这些模型是基于它们在处理各种类型数据（如线性、非线性、连续和离散数据）方面的能力，以及它们对缺失值的鲁棒性。

选择模型时需要考虑以下因素：

数据特征: 数据的特征分布、数量、类别等。

模型复杂度: 模型的复杂度越高，训练时间越长，过拟合风险也越高。

解释性: 是否需要解释模型预测的原因。

预测精度: 模型的预测精度是否满足需求。

4.2 特征选择与工程

4.2.1 特征筛选标准: 选择对目标变量影响显著的特征，例如：

优惠券特征: 优惠门槛、优惠金额、有效期等。

用户特征: 年龄、性别、消费习惯等。

商户特征: 店铺类型、地理位置、口碑等。

环境特征: 节假日、天气等。

4.2.2 特征构建方法: 可以通过以下方法构建特征：

特征工程: 对原始特征进行加工，例如将连续变量离散化，或创建新的特征。

特征选择: 选择对目标变量影响最大的特征。

降维: 减少特征数量，例如主成分分析 (PCA)。

4.3 模型训练与优化

4.3.1 数据分割与预处理: 将数据分割为训练集、验证集和测试集，并进行数据清洗、缺失值处理等预处理操作。

4.3.2 模型训练过程: 使用训练集训练模型，并进行模型参数的调整，使用不同的特征集组合来观察模型性能的变化。

4.3.3 模型参数调优: 使用交叉验证等方法，选择最优的模型参数。通常包括使用网格搜索、随机搜索或贝叶斯优化等方法来找到最优的模型参数。

4.4 模型评估与验证

4.4.1 评估指标与方法: 常用的评估指标包括准确率、召回率、F1 值等。可以使用混淆矩阵和ROC曲线来评估模型的性能，其中AUC值作为主要的评价指标，用于衡量模型预测的准确性。

4.4.2 模型验证结果分析: 分析模型的预测结果，评估模型的泛化能力，并进行模型改进。相关文献表明：GBDT+逻辑回归和XGBoost+逻辑回归模型在预测精度上有了显著提升。

4.5 模型应用与实效分析

4.5.1 模型在实际场景中的应用: 构建的模型被用于预测用户是否会在领取优惠券后的15天内使用它，这对于商家制定个性化的优惠券发放策略具有实际应用价值。

4.5.2 模型效果与改进方向: 评估模型在实际场景中的应用效果，并根据实际情况进行模型改进。可以尝试更多的集成模型，优化特征选择过程，进一步调整模型参数，以及精简代码以提高计算效率。

第五章节 项目优势

五、项目优势

完成搭建服务器，在此基础上进行数据存储和处理以及数据分析与可视化。

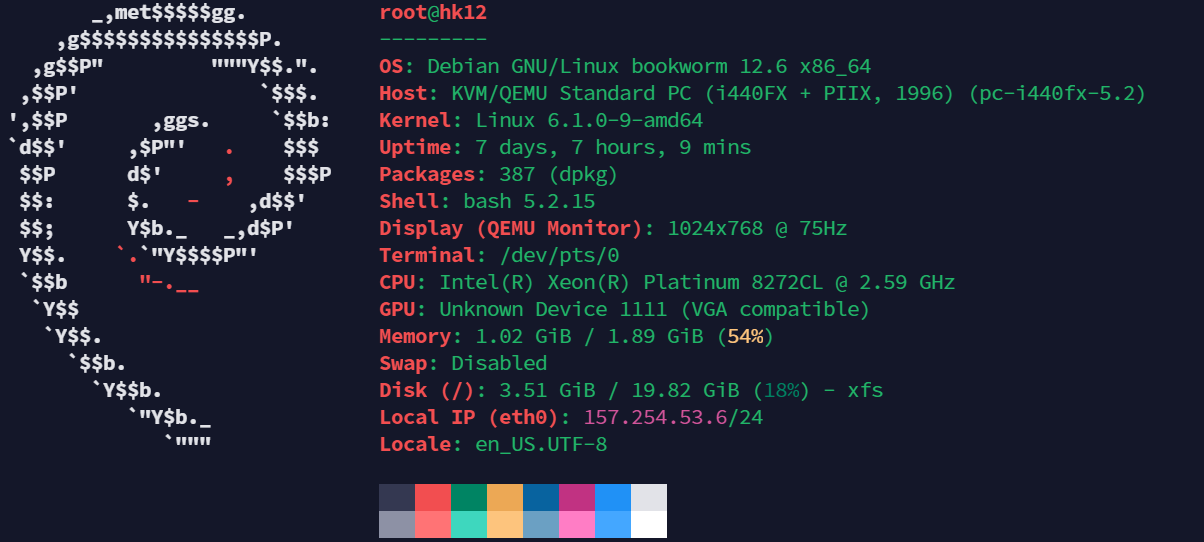
**图示

描述已自动生成**

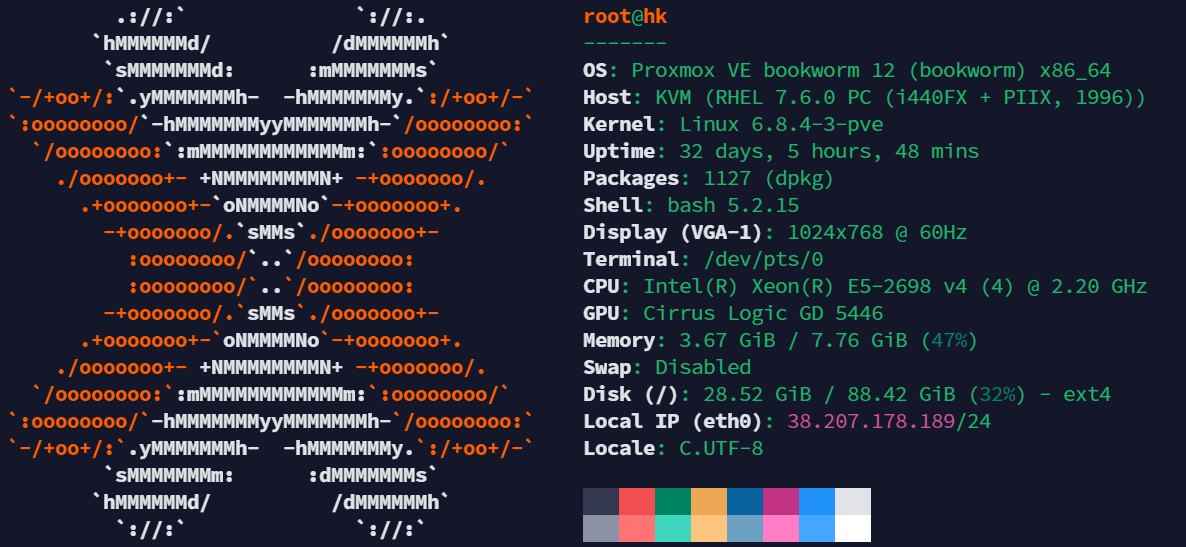
**项目UML构架图**

5.1操作系统环境信息

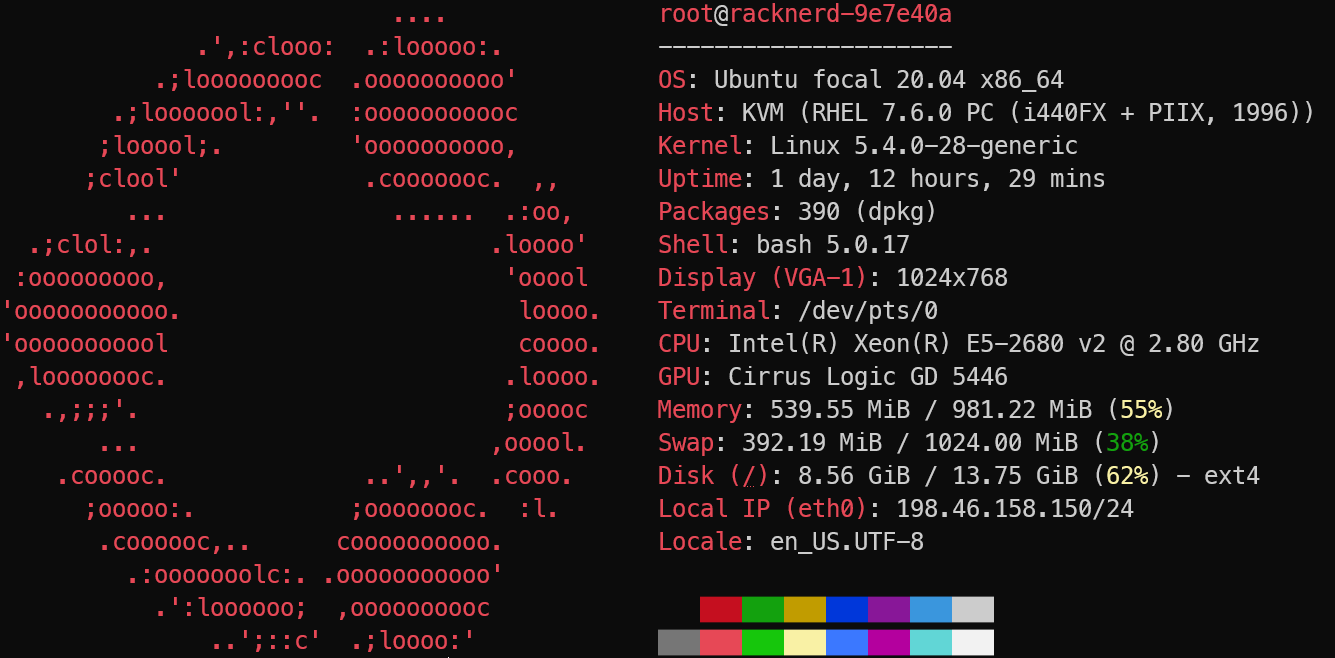
Nginx 服务器



后端服务器



数据库服务器



5.2服务器优势

* + 1. 控制和定制化

自己搭建服务器可以提供更高的控制度，允许根据具体需求选择合适的硬件配置和软件环境，实现个性化的网络服务和数据管理。

* + 1. 性能优化

通过优化服务器的硬件和网络设置，可以提高数据处理的性能和稳定性，满足大数据处理和复杂分析的需求。

* + 1. 数据安全性

在自己的服务器上，可以更好地保护数据隐私和安全，实施严格的访问控制和数据加密措施，减少数据泄露的风险。

* + 1. 可扩展性

服务器可以根据业务增长灵活扩展资源，如增加存储空间、计算能力和带宽，以适应不断变化的需求。

* + 1. 技术能力提升

搭建和管理服务器的过程可以锻炼技术人员的专业技能，包括硬件配置、网络管理、系统维护和安全防护等。

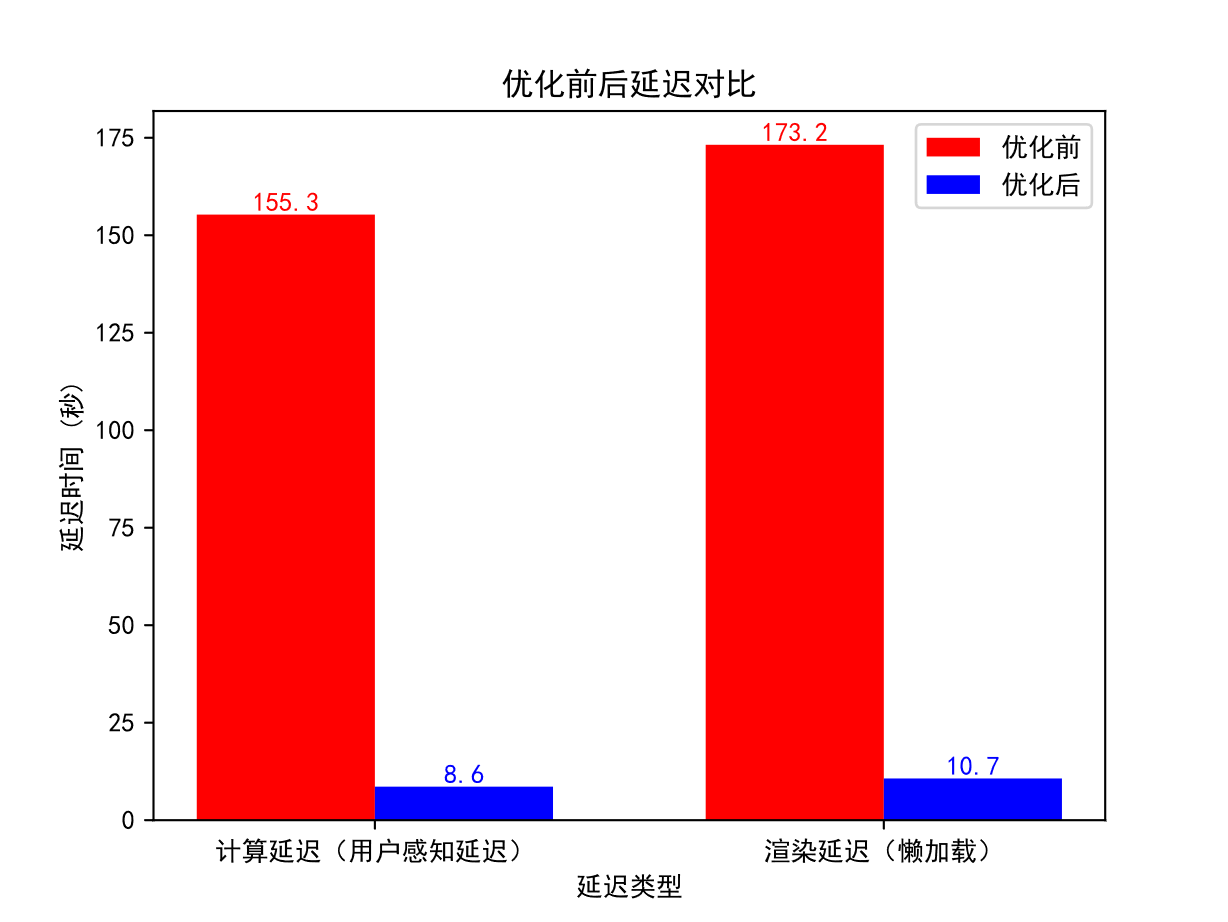
* + 1. 支持团队协作

服务器可以作为团队共享和协作的平台，提高工作效率和协同作用。

六、调试与优化

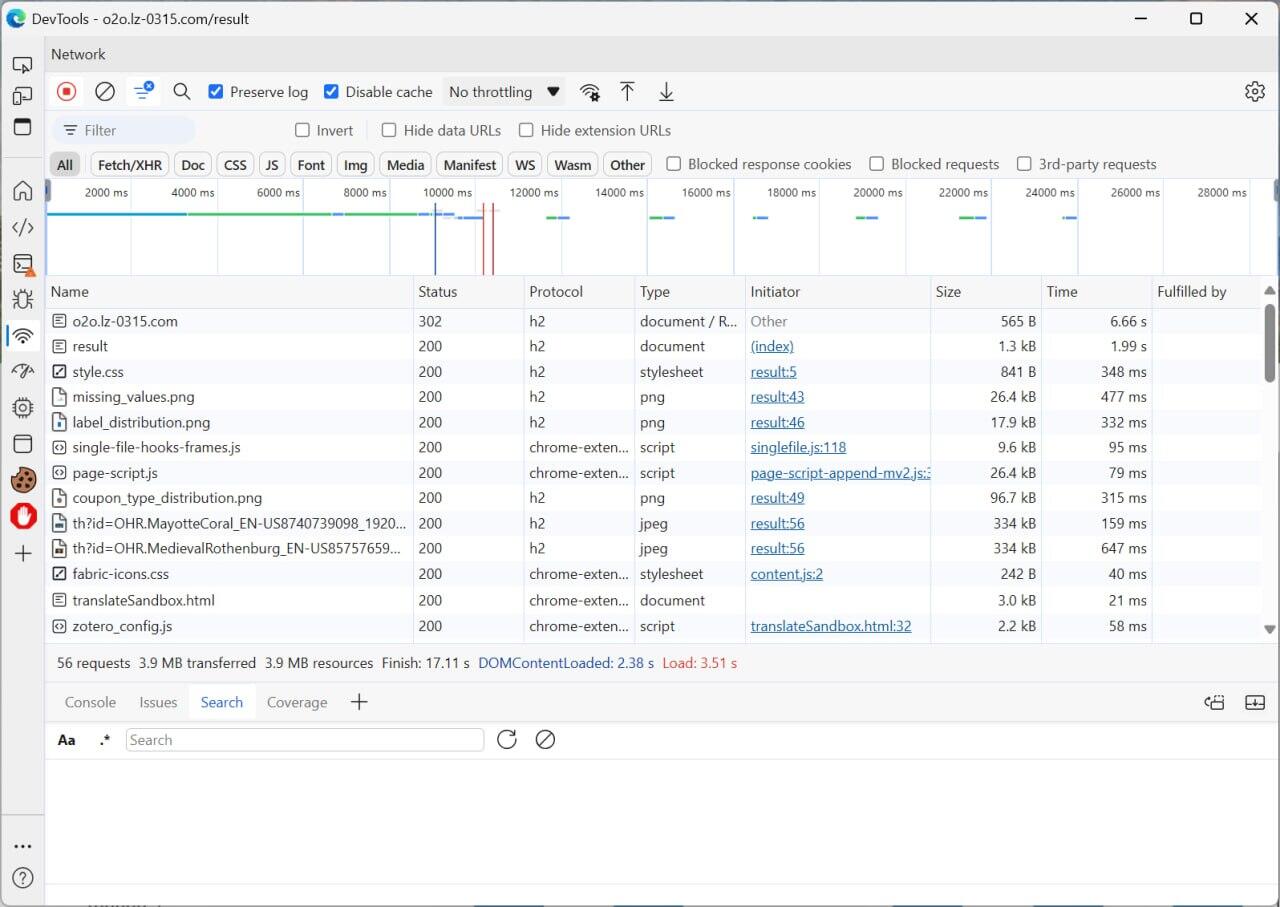
第六章节 调试与优化

我们尝试了多种方式来优化程序运行效率，以提升用户体验。经过一系列优化，程序运行效率提高了12.6倍。



6.1 程序总体运行效率

分析优化后的程序运行过程可得，程序在6s末完成数据上传，8s初完成数据计算并显示图表，10s末完成全部页面渲染。

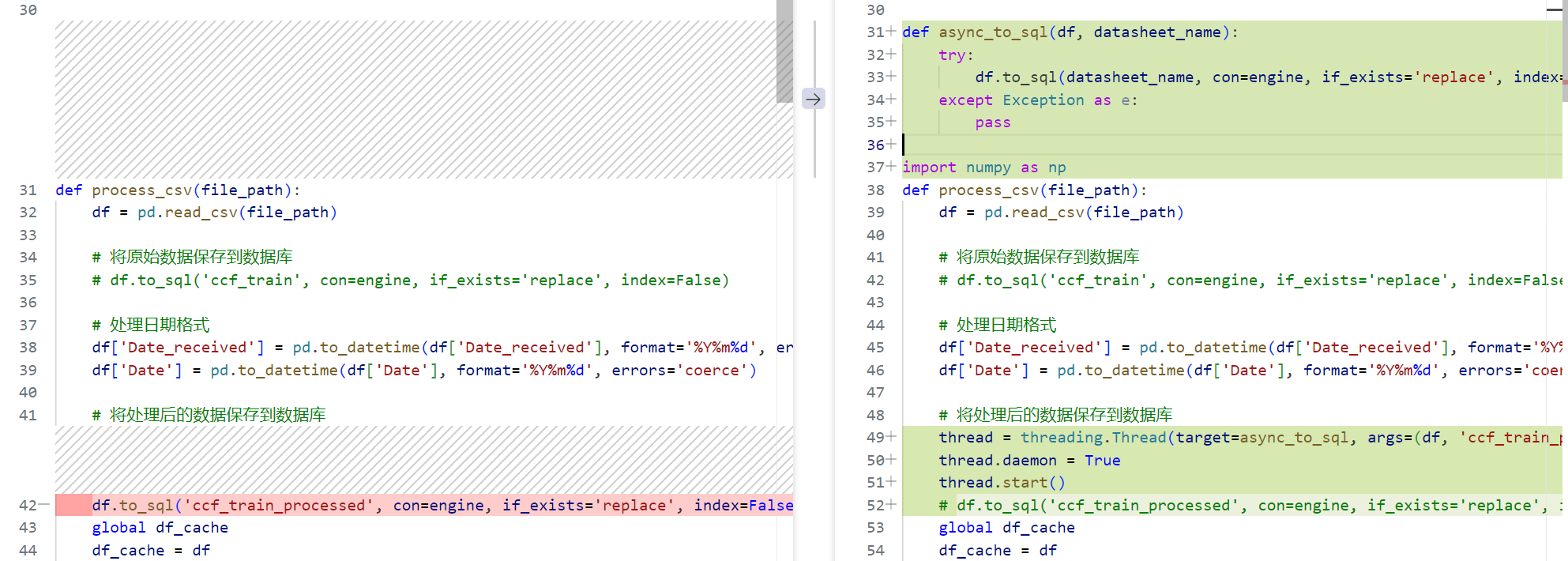


6.2原始版本

1. def process\_csv(file\_path):
2. df = pd.read\_csv(file\_path)
3. # 将原始数据保存到数据库
4. df.to\_sql('ccf\_train', con=engine, if\_exists='replace', index=False)
5. # 处理日期格式
6. df['Date\_received'] = pd.to\_datetime(df['Date\_received'], format='%Y%m%d', errors='coerce')
7. df['Date'] = pd.to\_datetime(df['Date'], format='%Y%m%d', errors='coerce')
8. # 生成标签列
9. df['label'] = df.swifter.apply(lambda row: 1 if pd.notnull(row['Date']) and pd.notnull(row['Coupon\_id']) and (row['Date'] - row['Date\_received']).days <= 15 else
10. -1 if pd.isnull(row['Date']) and pd.notnull(row['Coupon\_id']) else
11. 0 if pd.notnull(row['Date']) and pd.notnull(row['Coupon\_id']) and (row['Date'] - row['Date\_received']).days > 15 else
12. None, axis=1)
13. @app.route('/result', methods=['GET'])
14. def result():
15. df = pd.read\_sql('SELECT \* FROM ccf\_train\_processed', con=engine)
16. font = FontProperties(fname='C:/Windows/Fonts/simsun.ttc', size=12)

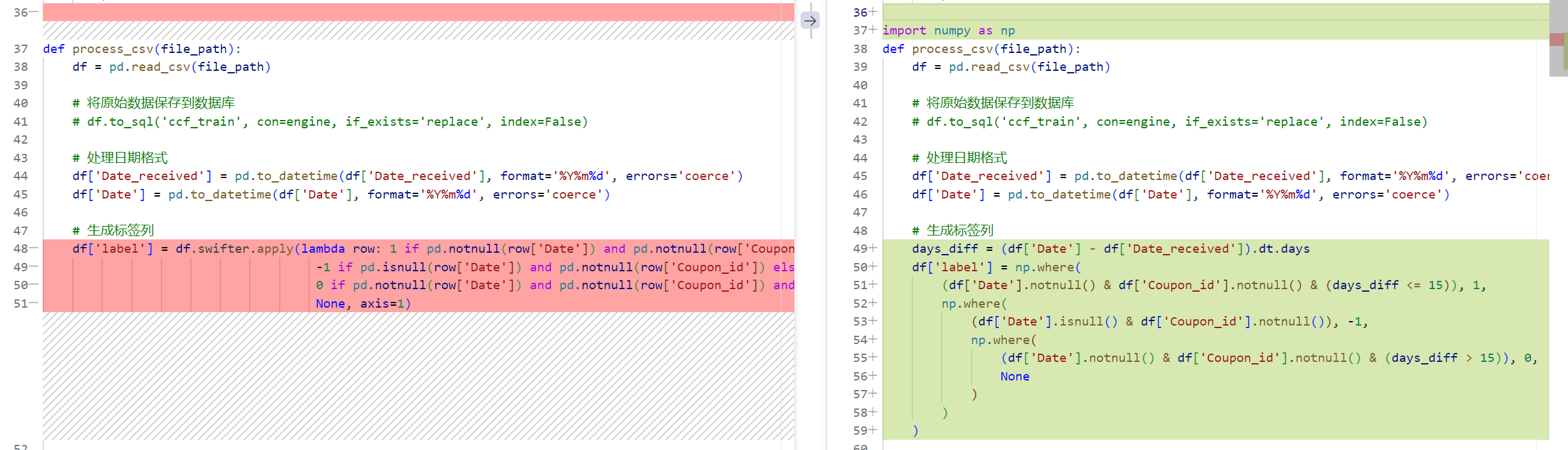
6.3优化过程

第一次优化



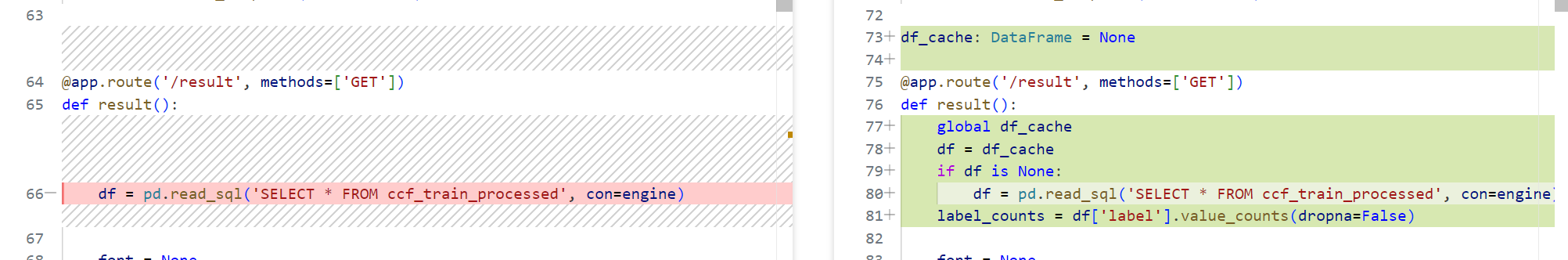
1. def async\_to\_sql(df, datasheet\_name):
2. try:
3. df.to\_sql(datasheet\_name, con=engine, if\_exists='replace', index=False)
4. except Exception:
5. Pass
6. thread = threading.Thread(target=async\_to\_sql, args=(df, 'ccf\_train\_processed'))
7. thread.daemon = True
8. thread.start()

第二次优化

****

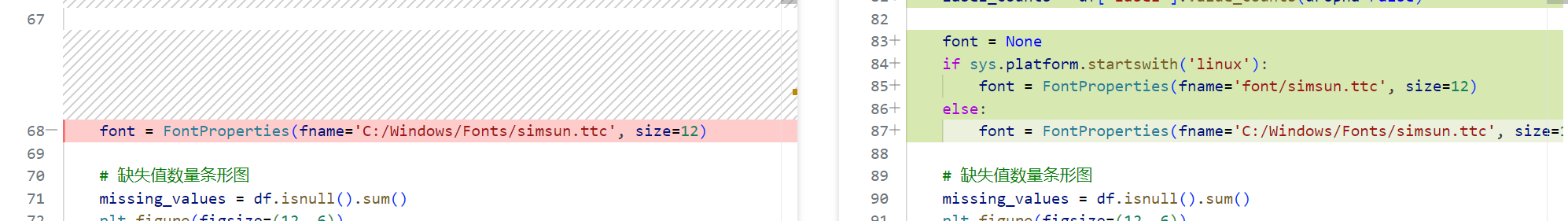
1. days\_diff = (df['Date'] - df['Date\_received']).dt.days
2. # 使用numpy的where方法进行向量化条件判断
3. df['label'] = np.where(
4. (df['Date'].notnull() & df['Coupon\_id'].notnull() & (days\_diff <= 15)), 1,
5. np.where(
6. (df['Date'].isnull() & df['Coupon\_id'].notnull()), -1,
7. np.where(
8. (df['Date'].notnull() & df['Coupon\_id'].notnull() & (days\_diff > 15)), 0,
9. None
10. )
11. )
12. )

第三次优化

****

1. global df\_cache
2. df = df\_cache
3. if df is None:
4. df = pd.read\_sql('SELECT \* FROM ccf\_train\_processed', con=engine)
5. label\_counts = df['label'].value\_counts(dropna=False)

第四次优化



1. font = None
2. if sys.platform.startswith('linux'):
3. font = FontProperties(fname='font/simsun.ttc', size=12)
4. elif sys.platform.startswith('win32'):
5. font = FontProperties(fname='C:/Windows/Fonts/simsun.ttc', size=12)
6. else:
7. pass

6.4进一步优化说明

在优化过程中，我们删除了多余的代码，重写了核心算法，并结合系统架构特性灵活引入了缓存机制，大大提高了程序的运行效率。

| **版本** | **时间复杂度** | **运行时间** | **优化说明** |
| --- | --- | --- | --- |
| 原始版本 | O(n2) | 155.3秒 | 效率低下且有部分冗余代码 |
| 第一次优化 | O(n2) | 72.5秒 | 异步化数据库操作，数据插入过程不再阻塞主线程 |
| 第二次优化 | O(log(n)) | 53.7 秒 | 重写核心算法，使用布尔索引和向量化操作提升运行效率 |
| 第三次优化 | O(log(n)) | 8.6秒 | 结合Gunicorn的多worker特性，合理引入缓存机制，去除冗余计算 |
| 第四次优化 | O(log(n)) | 8.6秒 | 在字体部分引入多系统判断逻辑，提升程序的跨平台能力 |

第七章节 项目总结与建议

七．项目总结和建议

7.1总结

通过本项目，我们成功地完成了基于 MySQL 的优惠券特征处理任务。所有数据预处理、特征构建、特征拼接步骤均已完成，并提供了用于数据可视化分析的 Python 代码。这为后续的数据分析和业务决策提供了坚实的基础。

本项目完成了基于MySQL的优惠券特征处理任务，包括数据预处理、特征构建、特征拼接和数据可视化分析。通过对数据的分析，我们了解了优惠券的使用规律和用户行为特征，为商家的优惠券投放策略提供了一定的参考。

7.2建议

商家可以根据用户的偏好和消费习惯，精准投放优惠券，提高优惠券的使用率和效果。

优化优惠券的设计，提高优惠力度和适用范围，吸引更多用户使用。

加强对优惠券使用情况的监测和分析，及时调整优惠券投放策略，提高营销效果。

附录

8.1用户特征代码部分

1. -- 用户在商家使用优惠券的次数
2. CREATE TEMPORARY TABLE user\_merchant\_coupon\_use AS
3. SELECT User\_id, Merchant\_id, COUNT(\*) AS user\_merchant\_coupon\_use
4. FROM ccf\_train
5. WHERE label = 1
6. GROUP BY User\_id, Merchant\_id;
7. -- 将用户在商家使用优惠券的次数加入原表
8. ALTER TABLE ccf\_train ADD COLUMN user\_merchant\_coupon\_use INT;
9. UPDATE ccf\_train a
10. LEFT JOIN user\_merchant\_coupon\_use b ON a.User\_id = b.User\_id AND a.Merchant\_id = b.Merchant\_id
11. SET a.user\_merchant\_coupon\_use = b.user\_merchant\_coupon\_use;

用户在商家领取的优惠券数：

1. -- 用户在商家领取的优惠券数
2. CREATE TEMPORARY TABLE user\_merchant\_coupon\_receive AS
3. SELECT User\_id, Merchant\_id, COUNT(\*) AS user\_merchant\_coupon\_receive
4. FROM ccf\_train
5. WHERE Coupon\_id IS NOT NULL
6. GROUP BY User\_id, Merchant\_id;
7. -- 将用户在商家领取的优惠券数加入原表
8. ALTER TABLE ccf\_train ADD COLUMN user\_merchant\_coupon\_receive INT;
9. UPDATE ccf\_train a
10. LEFT JOIN user\_merchant\_coupon\_receive b ON a.User\_id = b.User\_id AND a.Merchant\_id = b.Merchant\_id
11. SET a.user\_merchant\_coupon\_receive = b.user\_merchant\_coupon\_receive;

用户在商家消费的次数：

1. -- 用户在商家消费的次数
2. CREATE TEMPORARY TABLE user\_merchant\_consume AS
3. SELECT User\_id, Merchant\_id, COUNT(\*) AS user\_merchant\_consume
4. FROM ccf\_train
5. WHERE Date IS NOT NULL
6. GROUP BY User\_id, Merchant\_id;
7. -- 将用户在商家消费的次数加入原表
8. ALTER TABLE ccf\_train ADD COLUMN user\_merchant\_consume INT;
9. UPDATE ccf\_train a
10. LEFT JOIN user\_merchant\_consume b ON a.User\_id = b.User\_id AND a.Merchant\_id = b.Merchant\_id
11. SET a.user\_merchant\_consume = b.user\_merchant\_consume;

8.2优惠券特征代码部分

1. 1.1.
2. 2.2.2. \*\*商家相关特征\*\*
3. 3.3.'''sql
4. 4.4.-- 商家优惠券流行度
5. 5.5.创建临时表 merchant\_coupon\_popularity为
6. 6.6.选择Merchant\_id，将（\*）计数为total\_merchant\_coupons，
7. 7.7. SUM（CASE WHEN label = 1 THEN 1 ELSE 0 END） / COUNT（\*） AS merchant\_coupon\_popularity
8. 8.8.从ccf\_train
9. 9.9.按Merchant\_id分组;
10. 10.10.
11. 11.11.-- 将商家优惠券流行度加入原表
12. 12.12.更改表ccf\_train添加列merchant\_coupon\_popularity浮点数;
13. 13.13.更新ccf\_train
14. 14.14.JOIN merchant\_coupon\_popularity b ON a.Merchant\_id = b.Merchant\_id
15. 15.设置 a.merchant\_coupon\_popularity = b.merchant\_coupon\_popularity;

8.3特征拼接代码部分

1. -- 特征拼接
2. CREATE TABLE IF NOT EXISTS ccf\_train\_features AS
3. SELECT a.\*,
4. b.coupon\_popularity,
5. c.merchant\_coupon\_popularity,
6. d.total\_user\_coupons,
7. e.used\_user\_coupons,
8. f.user\_merchant\_coupon\_use,
9. g.user\_merchant\_coupon\_receive,
10. h.user\_merchant\_consume
11. FROM ccf\_train a
12. LEFT JOIN coupon\_popularity b ON a.Coupon\_id = b.Coupon\_id
13. LEFT JOIN merchant\_coupon\_popularity c ON a.Merchant\_id = c.Merchant\_id
14. LEFT JOIN user\_coupon\_count d ON a.User\_id = d.User\_id
15. LEFT JOIN user\_used\_coupon\_count e ON a.User\_id = e.User\_id
16. LEFT JOIN user\_merchant\_coupon\_use f ON a.User\_id = f.User\_id AND a.Merchant\_id = f.Merchant\_id
17. LEFT JOIN user\_merchant\_coupon\_receive g ON a.User\_id = g.User\_id AND a.Merchant\_id = g.Merchant\_id
18. LEFT JOIN user\_merchant\_consume h ON a.User\_id = h.User\_id AND a.Merchant\_id = h.Merchant\_id;

8.4完整的 SQL 脚本

1. -- 创建数据库和表
2. CREATE DATABASE IF NOT EXISTS O2O;
3. USE O2O;
4. CREATE TABLE IF NOT EXISTS ccf\_train (
5. User\_id INT NOT NULL,
6. Merchant\_id INT NOT NULL,
7. Coupon\_id INT,
8. Discount\_rate VARCHAR(10),
9. Distance INT,
10. Date\_received DATE,
11. Date DATE,
12. label INT,
13. PRIMARY KEY (User\_id, Merchant\_id, Coupon\_id, Date\_received)
14. );
15. -- 数据预处理
16. UPDATE ccf\_train SET Distance = 0 WHERE Distance IS NULL;
17. SET @avg\_distance = (SELECT AVG(Distance) FROM ccf\_train WHERE Distance IS NOT NULL);
18. UPDATE ccf\_train SET Distance = @avg\_distance WHERE Distance = 0;
19. UPDATE ccf\_train
20. SET label = CASE
21. WHEN Date IS NOT NULL AND Coupon\_id IS NOT NULL AND DATEDIFF(Date, Date\_received) <= 15 THEN 1
22. WHEN Date IS NULL AND Coupon\_id IS NOT NULL THEN 0
23. ELSE NULL
24. END;
25. -- 特征构建：优惠券相关特征
26. CREATE TEMPORARY TABLE coupon\_popularity AS
27. SELECT Coupon\_id, COUNT(\*) AS total\_coupons,
28. SUM(CASE WHEN label = 1 THEN 1 ELSE 0 END) / COUNT(\*) AS coupon\_popularity
29. FROM ccf\_train
30. GROUP BY Coupon\_id;
31. ALTER TABLE ccf\_train ADD COLUMN coupon\_popularity FLOAT;
32. UPDATE ccf\_train a
33. JOIN coupon\_popularity b ON a.Coupon\_id = b.Coupon\_id
34. SET a.coupon\_popularity = b.coupon\_popularity;
35. -- 特征构建：商家相关特征
36. CREATE TEMPORARY TABLE merchant\_coupon\_popularity AS
37. SELECT Merchant\_id, COUNT(\*) AS total\_merchant\_coupons,
38. SUM(CASE WHEN label = 1 THEN 1 ELSE 0 END) / COUNT(\*) AS merchant\_coupon\_popularity
39. FROM ccf\_train
40. GROUP BY Merchant\_id;
41. ALTER TABLE ccf\_train ADD COLUMN merchant\_coupon\_popularity FLOAT;
42. UPDATE ccf\_train a
43. JOIN merchant\_coupon\_popularity b ON a.Merchant\_id = b.Merchant\_id
44. SET a.merchant\_coupon\_popularity = b.merchant\_coupon\_popularity;
45. -- 特征构建：用户相关特征
46. CREATE TEMPORARY TABLE user\_coupon\_count AS
47. SELECT User\_id, COUNT(\*) AS total\_user\_coupons
48. FROM ccf\_train
49. GROUP BY User\_id;
50. ALTER TABLE ccf\_train ADD COLUMN total\_user\_coupons INT;
51. UPDATE ccf\_train a
52. JOIN user\_coupon\_count b ON a.User\_id = b.User\_id
53. SET a.total\_user\_coupons = b.total\_user\_coupons;
54. CREATE TEMPORARY TABLE user\_used\_coupon\_count AS
55. SELECT User\_id, COUNT(\*) AS used\_user\_coupons
56. FROM ccf\_train
57. WHERE label = 1
58. GROUP BY User\_id;
59. ALTER TABLE ccf\_train ADD COLUMN used\_user\_coupons INT;
60. UPDATE ccf\_train a
61. JOIN user\_used\_coupon\_count b ON a.User\_id = b.User\_id
62. SET a.used\_user\_coupons = b.used\_user\_coupons;
63. -- 特征拼接
64. CREATE TABLE IF NOT EXISTS ccf\_train\_features AS
65. SELECT a.\*, b.coupon\_popularity, c.merchant\_coupon\_popularity, d.total\_user\_coupons, e.used\_user\_coupons
66. FROM ccf\_train a
67. LEFT JOIN coupon\_popularity b ON a.Coupon\_id = b.Coupon\_id
68. LEFT JOIN merchant\_coupon\_popularity c ON a.Merchant\_id = c.Merchant\_id
69. LEFT JOIN user\_coupon\_count d ON a.User\_id = d.User\_id
70. LEFT JOIN user\_used\_coupon\_count e ON a.User\_id = e.User\_id;

8.5小组成员分工

| **姓名** | **工作内容** |
| --- | --- |
| 队长 | 搭建数据库服务器、代理服务器、数据处理服务器，配置Python环境，对成员完成的MySQL实现存储处理、python实现统计与可视化分析进行优化，重新算法，网页端开发，进行答辩汇报，演示文稿和认知实习文档的最终美化。 |
| 组员1 | 运用MySQL实现存储处理，创建表，导入数据，完成优惠券数据的预处理，处理设定标签，完善报告书。 |
| 组员2 | MySQL实现存储处理，完成优惠券数据的预处理，处理空值和设定标签，撰写报告书，完善ppt内容。 |
| 组员3 | python实现统计与可视化分析，完成缺失值数量条形图，优惠券类型占比饼状图，对项目进行总结，完成ppt。 |
| 组员4 | 运用python实现统计与可视化分析，完成标签分布直方图，对实习报告汇报书进行完善。 |

参考文献

[1]龙正.基于语义分割的异构多核平台大数据挖掘算法[J].大数据与人工智能,2023,4(5):

[2]皇博,胡霞,翟景治.基于机器学习的O2O优惠券投放策略研究[C]//中国管理现代化研究会,复旦管理学奖励基金会.第十八届（2023）中国管理学年会暨“一带一路”十周年研讨会论文集.中国矿业大学经济管理学院;湖南大学信息科学与工程学院;,2023:11.DOI:10.26914/c.cnkihy.2023.029347.

[3]宋霞.基于XGBoost的O2O优惠券使用预测研究及应用[D].西南交通大学,2021.DOI:10.27414/d.cnki.gxnju.2021.002056.

[4]陈浩阳,曾诚.个人O2O优惠券预测分析[J].物联网技术,2019,9(12):74-77.DOI:10.16667/j.issn.2095-1302.2019.12.020.

[5]陈浩阳.个人O2O优惠券预测分析与研究[D].湖北大学,2019.DOI:10.27130/d.cnki.ghubu.2019.000093.

[6]Kim J G, Yang S Y, Kwon J A, et al. The O2O marketing system using augmented reality and beacon[J]. International Journal of Multimedia and Ubiquitous Engineering, 2016, 11(12): 247-256.

致谢

在这段宝贵的实习经历即将画上圆满句号之际，我们团队心中充满了感激之情。在此，我想向所有在这段旅程中给予我们支持、指导和帮助的人表达最诚挚的感谢。

首先，我们要特别感谢这几天的授业导师段宏湘老师和叶丽丹老师，您们不仅在专业知识上给予了我们深刻的指导，更在职业发展和人生规划上提供了宝贵的建议。您们的智慧、耐心和热情对我们影响深远，我们将永远铭记您们的教诲。从课题选择之初到文章完成之时，感谢您们陪我字斟句酌，倾尽所能地点播和指导我。不论是传道授业，未来规划还是生活琐事感谢您的关爱。

其次，我要感谢广东泰迪智能科技股份有限公司提供的这次实习机会。在这里，我得以将所学知识应用于实际工作中，体验到了数据分析的全过程，从数据预处理到特征工程，再到模型构建和评估。实习期间，我学到了宝贵的经验和技能，这些都将为我未来的职业发展奠定坚实的基础。

此外，我们还要感谢所有在项目中给予我们帮助的老师和同学们。无论是在技术问题上的解答，还是在职业发展上的建议，你们的每一分支持都是我们前进的动力。特别感谢项目经理的领导和指导，数据分析师的深度分析和挖掘，开发团队的不懈努力以及测试团队的严谨细致。你们的专业知识和团队协作精神是本项目成功的关键。

在项目过程中，我们深入研究和应用了多种技术，包括但不限于以下几种：

**大数据处理技术**：我们学习和应用了 Hadoop、Spark 等大数据处理框架，这些技术使我们能够高效地处理和分析海量数据，提升了数据处理的速度和准确性。

**机器学习与人工智能**：通过学习和应用各种机器学习算法和深度学习模型，如逻辑回归、决策树、随机森林、神经网络等，我们能够更准确地预测用户行为，优化优惠券投放策略。

**数据可视化技术**：我们使用了 Matplotlib、Seaborn、Tableau 等数据可视化工具，这些工具帮助我们将复杂的数据和分析结果以直观的图表形式展示出来，为决策提供了有力支持。

**数据库管理技术**：MySQL、MongoDB 等数据库技术在本项目中起到了重要作用，我们将继续学习和优化这些数据库的使用，以更高效地存储和管理数据。

**Web 开发技术**：我们开发的网页应用程序使用了 Flask 框架，通过 Python 实现了用户友好的界面和良好的用户体验。

在课余时间，我们还自学了以下技术，以提升我们的专业能力：

**Hadoop 和 Spark**：用于大数据处理和分析，提升数据处理的速度和效率。

**MongoDB**：作为 NoSQL 数据库，用于处理和存储非结构化数据。

**HTML、CSS、JavaScript**：用于前端开发，实现用户友好的界面和良好的用户体验。

今后，我们将继续学习和探索这些技术，并不断引入新的技术手段，提升我们的技术能力和项目水平。我们将密切关注行业发展动态，及时掌握最新技术和应用，为项目的持续改进和优化提供有力支持。

最后，再次感谢所有参与和支持本项目的人士。你们的贡献和支持是我们前进的动力。我们期待在未来的项目中继续合作，共同探索和实现更多创新和突破。

请接受我最深切的感谢，感谢你们每一个人在我的实习生涯中所扮演的角色。我期待着将这段经历中所学到的知识和技能应用到未来的工作中，为社会贡献自己的力量。再次感谢所有给予我帮助的人，愿我们的未来都充满无限可能。

谢谢大家！