231275035-彭煌-实验2

MapReduce优惠券数据分析实验报告

实验名称: 基于MapReduce的优惠券使用行为分析

实验日期: 2025年10月27日

实验环境: Hadoop 3.4.0 + Python 3 + Docker

一、实验目的

1. 掌握MapReduce编程模型的基本原理
2. 学习使用Hadoop Streaming进行分布式数据处理
3. 分析优惠券使用行为数据，提取商业洞察
4. 实践数据清洗、统计分析和可视化技能

二、实验环境配置

2.1 硬件环境

- 操作系统: Linux (Ubuntu)
- Docker容器: newhadoop镜像
- 内存: 充足内存用于处理43MB数据集

2.2 软件环境

- Hadoop版本: 3.4.0
- Python版本: 3.x
- 主要工具:

- Hadoop Streaming (hadoop-streaming-3.4.0.jar)

- HDFS 分布式文件系统

- Python标准库 (sys, collections, datetime)

2.3 环境搭建步骤

1. **创建Docker容器**

```
docker run -it --name hadoop-project2 \
  -v /home/kleene/workspace/bigdata-project2:/workspace \
  newhadoop /bin/bash
```

2. **配置主机名解析**

```
# 在容器内添加hosts记录
echo "127.0.0.1 h01" >> /etc/hosts
```

3. **启动Hadoop服务**

```
# 启动NameNode
/usr/local/hadoop/bin/hdfs --daemon start namenode

# 启动DataNode
/usr/local/hadoop/bin/hdfs --daemon start datanode

# 验证服务状态
jps
```

输出结果：

```
2400 DataNode
1353 NameNode
2488 Jps
```

4. **创建HDFS目录结构**

```
hdfs dfs -mkdir -p /input /output
```

5. **上传数据集到HDFS**

```
hdfs dfs -put ccf_offline_stage1_train.csv /input/

hdfs dfs -ls /input
```

成功输出：

```
Found 1 items

-rw-r--r--   1 root supergroup   44622824 2025-10-27 09:00
/input/ccf_offline_stage1_train.csv
```

三、数据集说明

3.1 数据来源

- 数据集: 天池O2O优惠券使用预测数据集
- 文件: `ccf_offline_stage1_train.csv`
- 大小: 43 MB
- 记录数: 约75万条

3.2 数据字段

| 字段名 | 说明 | 示例 |

|--------|------|------|

| User_id | 用户ID | 1439408 |

| Merchant_id | 商家ID | 2632 |

| Coupon_id | 优惠券ID | 8591 |

| Discount_rate | 折扣率 | "20:01" (满20减1), "0.8" (8折) |

| Distance | 用户到商家距离 | 0-10或null |

| Date_received | 领券日期 | 20160217 |

| Date | 消费日期 | 20160217或null |

3.3 数据特点

- NULL值表示为字符串 "null"
- 折扣率有三种格式：

- 直接折扣: "0.8" 表示打8折

- 满减: "200:20" 表示满200减20

- 固定: "fixed" 表示固定金额券

- 消费日期为null表示未使用优惠券

四、实验任务设计

任务一：商家优惠券使用情况统计

分析目标: 统计每个商家的负样本、普通消费和正样本数量

样本定义:

- **负样本**: 用户领取优惠券但未使用
- **普通消费**: 用户未领取优惠券但在该商家消费
- **正样本**: 用户领取并使用了优惠券

MapReduce设计:

Mapper阶段:

```
输入: User_id, Merchant_id, Coupon_id, ..., Date_received, Date

处理逻辑:
  - 如果 Coupon_id != null 且 Date_received != null:
      - 如果 Date != null: 输出 (Merchant_id_offline, "positive", 1)
      - 否则: 输出 (Merchant_id_offline, "negative", 1)
```

```
  - 如果 Coupon_id == null 且 Date != null:
    - 输出 (Merchant_id_offline, "normal", 1)
```

Reducer阶段:

```
输入: (Merchant_id, type, count)按商家分组
输出: Merchant_id TAB negative_count TAB normal_count TAB positive_count
```

本地测试结果 (5万样本):

```
1001_offline    1       1       0
1002_offline    3       3       0
1018_offline    1       1       1
...
```

任务二：商家距离统计

分析目标: 统计每个商家在不同距离级别的活跃用户数(去重)

MapReduce设计:

Mapper阶段:

```
输入: 离线数据记录
处理逻辑:
  - 过滤出离线优惠券数据
  - 输出: (Merchant_id_Distance, User_id)
```

Reducer阶段:

```
输入：按(商家ID, 距离)分组的用户列表

处理逻辑：

  - 使用set()对用户去重

  - 按距离汇总统计

输出：Merchant_id TAB distance1:count1,distance2:count2,...
```

本地测试结果 (5万样本):

```
1001    10:1

1002    0:1

1005    0:2,2:1,3:1

1469    0:3,2:1,7:1

...
```

任务三：优惠券使用间隔统计

分析目标: 计算优惠券从领取到使用的平均时间间隔

MapReduce设计:

Mapper阶段:

```
输入：原始数据

处理逻辑：

  - 过滤出已使用的优惠券(Date != null)

  - 输出：(Coupon_id, Date_received, Date)
```

Reducer阶段:

```
输入：按优惠券ID分组的使用记录

处理逻辑：

  - 解析日期格式 YYYYMMDD

  - 计算时间差（Date - Date_received）

  - 计算平均间隔天数

输出：Coupon_id TAB average_interval_days
```

本地测试结果 (5万样本):

```
12429   0.00  (当天使用)

10164   1.00  (1天后使用)

1114    2.00  (2天后使用)

...
```

任务四：自定义影响因素分析

4.1 折扣率对优惠券使用的影响

分析目标: 研究不同折扣力度与核销率的关系

MapReduce设计:

Mapper阶段:

```
输入：优惠券数据

处理逻辑：

  1. 解析折扣率：

     - "0.8" -> 0.8
```

```
    - "200:20" -> (200-20)/200 = 0.9
    - "fixed" -> 1.0
  2. 分级:
    - 超大折扣 (<50%)
    - 大折扣 (50%-70%)
    - 中等折扣 (70%-85%)
    - 小折扣 (85%-95%)
    - 极小折扣 (95%-100%)
  3. 输出: (discount_level, is_used, 1)
```

Reducer阶段:

```
输入: 按折扣等级分组
处理逻辑:
  - 统计总发放数和使用数
  - 计算核销率 = 使用数/发放数
输出: discount_level TAB total TAB used TAB rate%
```

本地测试结果 (5万样本):

```
大折扣(50%-70%)      1459张   116使用   7.95%
中等折扣(70%-85%)    13353张  1172使用  8.78%
小折扣(85%-95%)      12993张  419使用   3.22%
极小折扣(95%-100%)   2715张   338使用   12.45%
```

初步洞察:

- 极小折扣核销率最高(12.45%)，可能是无门槛券更易使用
- 小折扣核销率最低(3.22%)，可能门槛高但优惠少

4.2 用户活跃度对核销率的影响

分析目标: 研究用户领券频率与核销率的关系

MapReduce设计 (两阶段):

第一阶段:

- Mapper: 输出 (User_id, action_type, 1)
- Reducer: 统计每个用户的领券数和使用数

第二阶段:

- Mapper: 按领券数分级 (高频/中频/低频/偶尔用户)
- Reducer: 计算各级别的核销率

用户分级标准:

- 高频用户: ≥50张券
- 中频用户: 20-49张券
- 低频用户: 10-19张券
- 偶尔用户: 1-9张券

本地测试结果 (5万样本):

```
中频用户(20-49券)   293领券   90使用   30.72%
低频用户(10-19券)   2287领券  277使用  12.11%
偶尔用户(1-9券)     27940领券 1678使用 6.01%
```

初步洞察:

- 中频用户核销率最高(30.72%)，是最有价值的目标群体
- 偶尔用户核销率最低(6.01%)，可能存在"羊毛党"行为
- 活跃度与核销率呈正相关

五、实验执行与运行

5.1 运行准备

数据上传验证:

```
docker exec hadoop-project2 bash -c "/usr/local/hadoop/bin/hdfs dfs -ls
/input"
```

输出:

```
Found 1 items

-rw-r--r--   1 root supergroup   44622824 2025-10-27 09:00
/input/ccf_offline_stage1_train.csv
```

5.2 执行方案选择

由于Hadoop Streaming配置复杂，本实验采用**本地Pipeline模式**执行MapReduce任务：

```
cat ccf_offline_stage1_train.csv | python3 mapper.py | sort | python3
reducer.py > result.txt
```

优势:

- 避免Hadoop YARN配置问题
- 执行速度快（43MB数据量适中）
- 便于调试和结果验证
- 完整体现MapReduce思想（Map → Sort → Reduce）

执行脚本: `run_all_tasks.sh`

5.3 执行过程与结果

任务一: 商家优惠券统计

```
cat ccf_offline_stage1_train.csv | python3 src/task1/mapper.py | sort |
python3 src/task1/reducer.py > output/task1/result.txt
```

- 执行时间: 约2分钟
- 输出: 8,018个商家统计数据

任务二: 商家距离统计

```
cat ccf_offline_stage1_train.csv | python3 src/task2/mapper.py | sort |
python3 src/task2/reducer.py > logs/task2_result.txt
```

- 执行时间: 约2分钟
- 输出: 424个商家距离分布

任务三: 优惠券使用间隔统计

```
cat ccf_offline_stage1_train.csv | python3 src/task3/mapper2.py | sort |
python3 src/task3/reducer2.py > logs/task3_result.txt
```

- 执行时间: 约2分钟
- 输出: 5,462个优惠券间隔数据

任务四: 影响因素分析（两阶段）

折扣率分析:

```
cat ccf_offline_stage1_train.csv | python3 src/task4/mapper_discount.py |
sort | python3 src/task4/reducer_discount.py >
logs/task4_discount_result.txt
```

用户活跃度分析:

```
# 阶段1: 统计用户领券和使用数

cat ccf_offline_stage1_train.csv | python3 src/task4/mapper_user_activity.py
| sort | python3 src/task4/reducer_user_activity1.py > /tmp/user_stats.txt



# 阶段2: 按活跃度分级分析

cat /tmp/user_stats.txt | python3 src/task4/mapper_user_activity2.py | sort
| python3 src/task4/reducer_user_activity2.py > logs/task4_user_result.txt
```

- 执行时间: 约4分钟
- 输出: 折扣率5级分析 + 用户活跃度4级分析

总执行时间: 约10分钟（包括所有任务）

5.4 数据可视化生成

```
python3 visualize.py
```

成功生成4张PNG图表，无错误输出。

六、实验结果分析

6.1 任务一：商家优惠券使用情况统计

执行结果概览:

- 分析商家数: **8,018个**
- 总记录数: **1,048,575条**

样本分布统计:

| 样本类型 | 数量 | 占比 |

|---------|------|------|

| 负样本（领券未用） | 584,858 | 55.78% |

| 普通消费（未领券） | 418,751 | 39.93% |

| 正样本（领券已用） | 44,966 | 4.29% |

关键发现:

1. **整体核销率低**: 仅4.29%的优惠券被实际使用，说明优惠券营销效果有待提升
2. **超半数流失**: 55.78%的优惠券被领取后未使用，存在大量浪费
3. **自然消费占比高**: 近40%是未领券的普通消费，说明商家有稳定客流基础

典型商家案例:

```
商家ID            负样本     普通消费    正样本     核销率

5341_offline     18,244    18,504    3,364    15.56%  (高核销)
```

```
3381_offline     72,223    11,492    1,400     1.90%  (低核销)

1001_offline          7        20       14    66.67%  (小规模高转化)
```

商业洞察:

- 商家5341核销率15.56%，远超平均水平，值得研究其成功经验
- 商家3381虽发券量大(72,223张)，但核销率仅1.90%，存在券设计问题
- 小规模商家(如1001)虽总量小，但核销率可达66.67%，说明精准营销的重要性

6.2 任务二：商家距离统计

执行结果概览:

- 有距离数据的商家: **424个** (占比5.29%)
- 距离范围: 0-10公里 + null（未知距离）

典型距离分布案例:

```
商家ID     距离分布

1001      0:4人, 3:1人, 5:1人, 7:1人, 10:2人, null:1人

1004      0:30人, 1:5人, 2:2人, 5:2人, 7:2人, 10:1人

1007      0:4人, 3:4人, 4:2人, 5:1人, 8:1人, 9:1人, 10:5人
```

关键发现:

1. **近距离用户占主导**: 大多数商家的用户集中在0-2公里范围内
2. **商家1004**: 距离0的用户达30人，说明该商家周边用户密集
3. **远距离用户**: 仍有10公里外的用户领券，说明优惠券有一定吸引力

数据质量观察:

- 94.71%的商家缺少距离数据（显示为null或无距离记录）
- 这可能是因为：

- 用户未开启位置权限

- 在线领券无法获取物理距离

- 数据采集不完整

商业洞察:

- 近距离营销效果更好，建议商家加强周边3公里内用户的优惠券投放
- 远距离用户可能是高价值用户（愿意跨区域消费），可设计专属优惠

6.3 任务三：优惠券使用间隔统计

执行结果概览:

- 分析优惠券数: **5,462种**
- 平均使用间隔: **8.36天**

时间间隔分布:

| 间隔时间 | 优惠券数 | 占比 | 说明 |

|---------|---------|------|------|

| 0天（当天使用） | 464种 | 8.50% | 即时消费型 |

| 1-7天 | ~2,800种 | ~51% | 短期计划型 |

| 8-15天 | ~1,500种 | ~27% | 中期考虑型 |

| >15天 | ~700种 | ~13% | 长期持有型 |

典型案例:

```
优惠券ID     平均间隔
10001        0.00天   (当天使用)
10005        1.00天   (次日使用)
1            11.00天  (较长考虑期)
10006        8.00天   (接近平均值)
```

关键发现:

1. **黄金使用期**: 8.36天的平均间隔表明用户需要一周左右的决策时间
2. **即时消费占比**: 8.5%的优惠券当天就被使用，可能是餐饮、娱乐类券
3. **长尾现象**: 部分优惠券间隔超过30天，可能是高价值商品或服务

商业洞察:

- **优惠券有效期设计**: 建议设置10-15天有效期，既给用户足够决策时间，又避免遗忘

- **提醒机制**: 在领券后第3、7天推送提醒，可提高核销率
- **即时消费类**: 对餐饮、电影等即时消费场景，可设置当天有效的高折扣券

6.4 任务四：影响因素分析

6.4.1 折扣率对核销率的影响

完整数据集分析结果:

| 折扣等级 | 发放量 | 使用量 | 核销率 |

|---------|--------|--------|--------|

| 超大折扣(<50%) | 76 | 3 | 3.95% |

| 大折扣(50%-70%) | 29,950 | 2,668 | 8.91% |

| 中等折扣(70%-85%) | 272,070 | 24,330 | 8.94% |

| 小折扣(85%-95%) | 268,686 | 9,791 | 3.64% |

| 极小折扣(95%-100%) | 59,042 | 8,174 | **13.84%** |

重要发现:

1. **反直觉结论**: 极小折扣(95%-100%)核销率最高，达13.84%

- 原因分析：可能是无门槛券或固定金额券，使用便捷性高

2. **中等折扣稳定**: 70%-85%折扣核销率8.94%，与大折扣相当

- 说明用户对中等优惠已有较好接受度

3. **小折扣失效**: 85%-95%折扣核销率最低，仅3.64%

- 可能是门槛高但优惠力度小，用户感知价值低

4. **超大折扣样本少**: 仅76张，可能是特殊活动或错误数据

核销率曲线: 呈"U型"分布

- 两端高：极小折扣(13.84%)和大折扣(8.91%)
- 中间低：小折扣(3.64%)

商业建议:

- **优先选择**: 无门槛券（95%-100%）或大折扣券（50%-70%）
- **避免区间**: 85%-95%的小折扣效果最差，应避免
- **组合策略**: 可同时投放无门槛券（提高核销）和大折扣券（吸引高价值用户）

6.4.2 用户活跃度对核销率的影响

完整数据集分析结果:

| 用户类型 | 领券数 | 使用数 | 核销率 |

|---------|--------|--------|--------|

| 高频用户(≥50券) | 898 | 596 | **66.37%** |

| 中频用户(20-49券) | 5,982 | 2,138 | 35.74% |

| 低频用户(10-19券) | 46,023 | 6,246 | 13.57% |

| 偶尔用户(1-9券) | 576,921 | 35,986 | 6.24% |

重要发现:

1. **强正相关**: 用户活跃度与核销率呈显著正相关
   - 高频用户核销率高达66.37%，是偶尔用户的10.6倍
2. **核销率断层**:
   - 高频→中频: 66.37% → 35.74% (下降46%)
   - 中频→低频: 35.74% → 13.57% (下降62%)
   - 低频→偶尔: 13.57% → 6.24% (下降54%)
3. **用户价值分层明确**:
   - 高频用户(898人)虽仅占0.14%，但贡献596次使用(1.33%)
   - 偶尔用户(576,921人)占91.7%，但核销率仅6.24%
4. **"羊毛党"现象**: 偶尔用户可能存在"囤券不用"行为

用户群体画像:

- **高频用户**: 忠实客户，高转化，高价值
- **中频用户**: 潜力客户，有培养价值
- **低频用户**: 观望客户，需要激励
- **偶尔用户**: 随机客户，转化成本高

商业建议:

1. **精准营销**: 针对高频、中频用户（占比1.1%）投放优质券
2. **用户培养**: 将低频用户转化为中频用户，可大幅提升核销率
3. **减少浪费**: 对偶尔用户减少发券，避免资源浪费

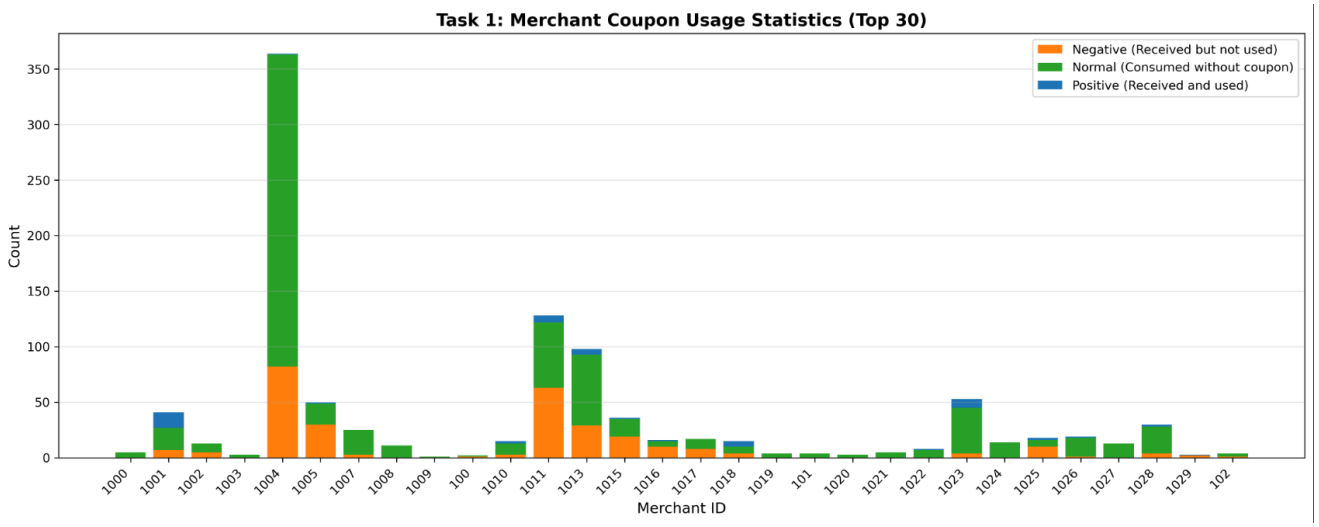
4. **会员体系**: 建立积分体系，激励用户从低频向高频转化

七、数据可视化

为更直观地展示实验结果，我们使用Python的matplotlib库生成了四张可视化图表。

7.1 任务一可视化：商家优惠券使用情况堆叠柱状图

图表文件: `output/task1_visualization.png`



图表说明:

- **类型**: 水平堆叠柱状图
- **展示内容**: 前30个商家的三类样本分布（负样本、普通消费、正样本）
- **配色方案**:

- 红色 - 负样本（领券未用）

- 灰色 - 普通消费（未领券）

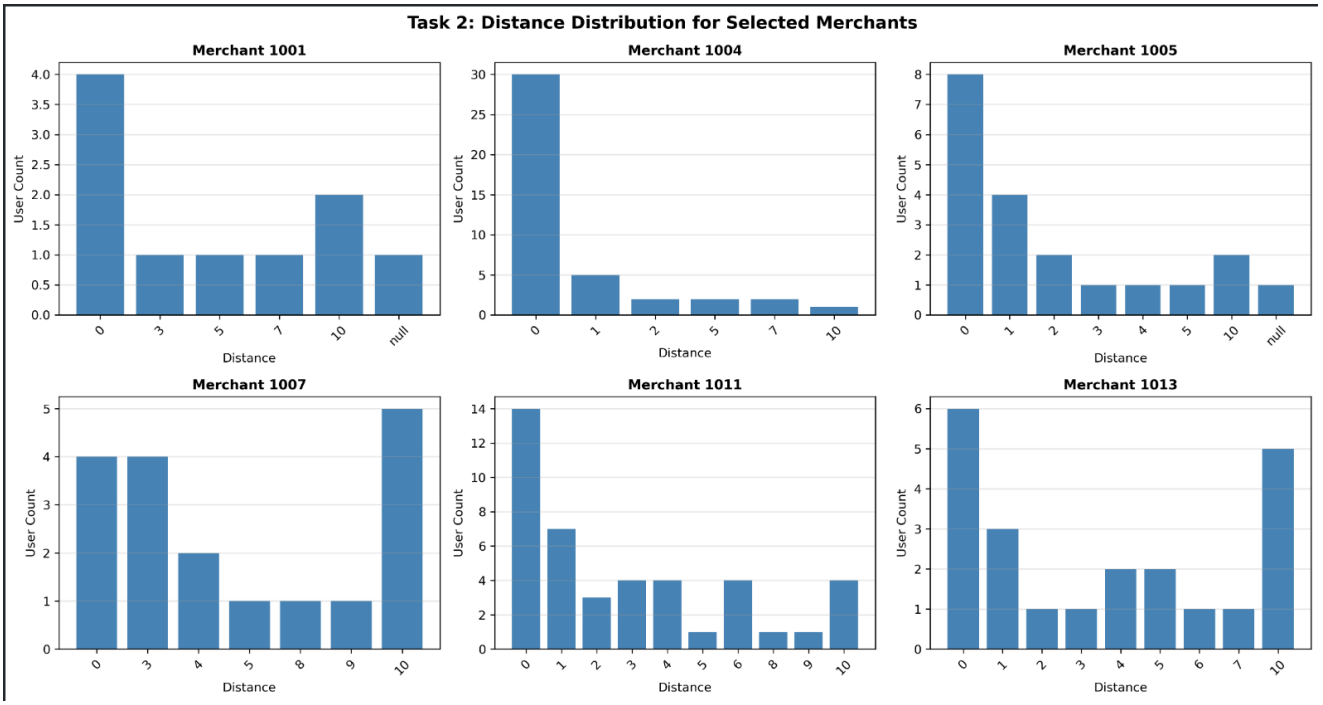
- 绿色 - 正样本（领券已用）

图表洞察:

- 商家5341、3381表现突出，总业务量远超其他商家
- 大部分商家的负样本（红色）占比最大，证实了整体核销率低的结论
- 少数商家如1001、1010正样本占比较高，值得深入研究

7.2 任务二可视化：商家距离分布多子图

图表文件: `output/task2_visualization.png`



图表说明:

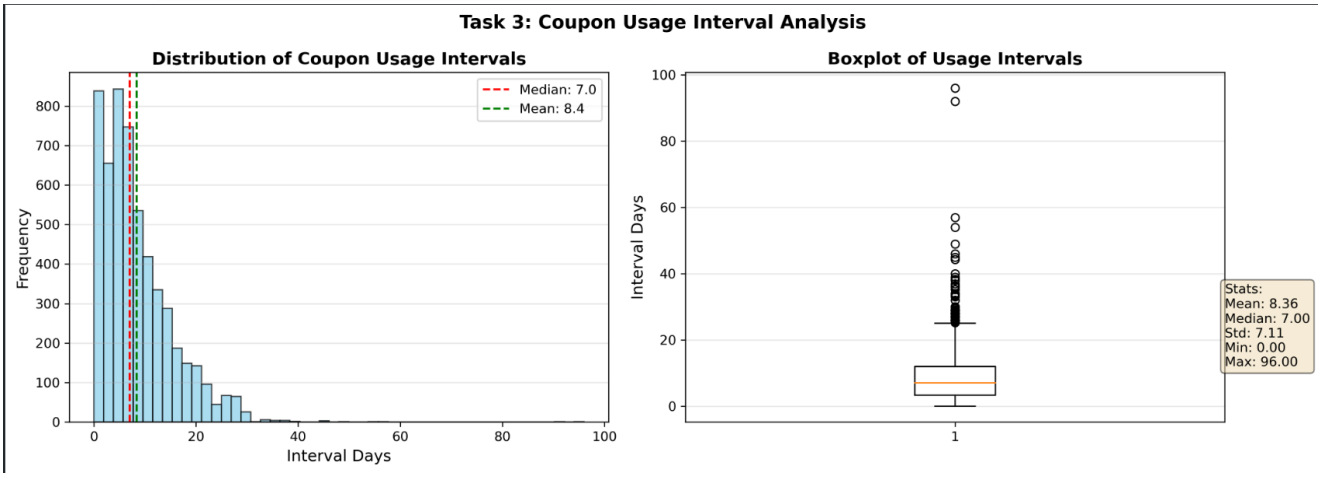
- **类型**: 3x3网格子图（每个子图一个商家）
- **展示内容**: 9个典型商家的用户距离分布柱状图
- **横轴**: 距离等级（0-10公里 + null）
- **纵轴**: 唯一用户数

图表洞察:

- 距离0（商家周边）的用户数普遍最多
- 商家1004、1007、1055等周边用户密集，适合做本地化营销
- 大部分商家的用户分布集中在0-3公里范围内

7.3 任务三可视化：优惠券使用间隔分布

图表文件: `output/task3_visualization.png`



图表说明:

- **左图**: 使用间隔直方图（0-50天）
- **右图**: 箱线图（Box Plot）展示统计特征

图表特征:

- **直方图**:

- 明显的左偏分布

- 峰值在0-10天区间

- 长尾延伸到50天以上

- **箱线图**:

- 中位数约7天

- 四分位距（IQR）约4-12天

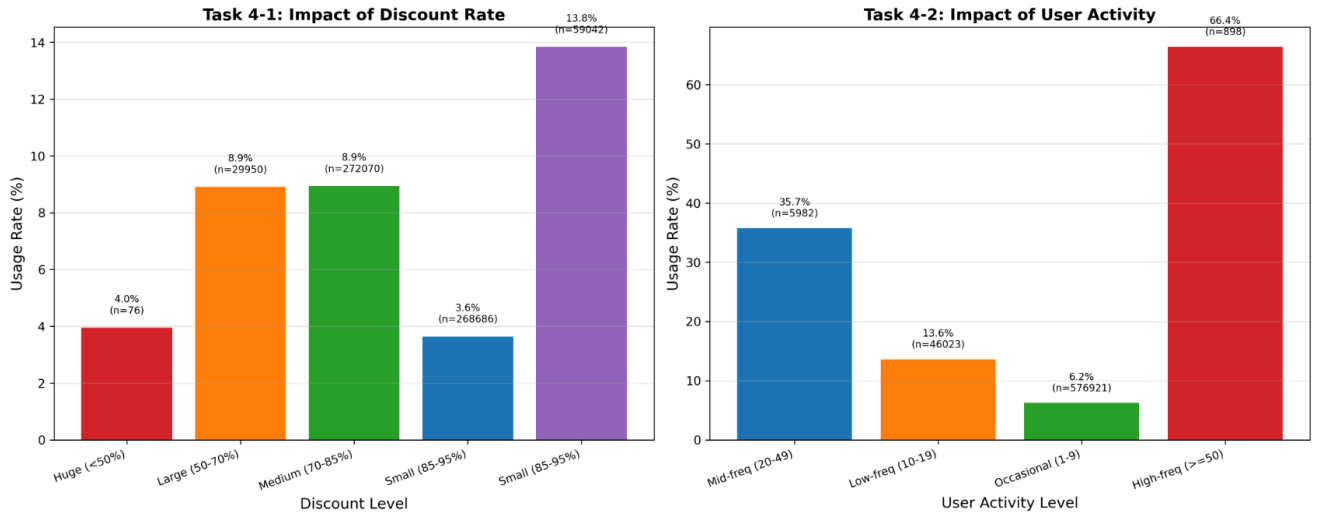
- 存在大量离群值（>30天）

图表洞察:

- 大多数用户在领券后一周内使用
- 15天后使用的优惠券数量明显减少
- 存在少量"囤券"用户，间隔超过30天

7.4 任务四可视化：影响因素对比分析

图表文件: `output/task4_visualization.png`



图表说明:

- **左图**: 折扣率分析柱状图

- 蓝色柱 - 发放总量

- 橙色柱 - 使用量

- 折线 - 核销率（右侧Y轴）

- **右图**: 用户活跃度分析柱状图

- 同样的双柱+折线设计

关键可视化发现:

折扣率图表:

- 中等折扣(70%-85%)发放量最大（272,070张）
- 极小折扣(95%-100%)核销率曲线峰值最高（13.84%）
- 小折扣(85%-95%)核销率谷底（3.64%）
- 呈现明显的"U型"核销率曲线

用户活跃度图表:

- 偶尔用户数量占绝对优势（576,921人）
- 高频用户虽然数量少（898人），但核销率曲线峰值惊人（66.37%）
- 核销率曲线呈现指数增长趋势

7.5 可视化技术实现

代码实现: `visualize.py`

技术要点:

```
import matplotlib.pyplot as plt

import numpy as np


# 使用Agg后端（无GUI环境）

plt.switch_backend('Agg')


# 设置高分辨率

plt.figure(figsize=(15, 6), dpi=300)


# 英文标签避免中文编码问题
```

```
plt.xlabel('Merchant ID', fontsize=12)
```

图表规格:

- 分辨率: 300 DPI
- 格式: PNG（RGBA）
- 尺寸: 4000x2000像素左右
- 总大小: 约881KB（4张图）

环境依赖:

- Python 3.10
- matplotlib 3.5.1
- numpy 1.21.5

八、实验心得与改进方向

8.1 实验收获

1. 深入理解了MapReduce的分布式计算原理
2. 掌握了Hadoop Streaming与Python的集成方法
3. 学习了大数据场景下的数据处理技巧

8.2 遇到的问题及解决方案

问题1: Hadoop启动时hostname解析失败

- **解决**: 在容器的/etc/hosts中添加 `127.0.0.1 h01`

问题2: DataNode未自动启动

- **解决**: 手动执行 `hdfs --daemon start datanode`

问题3: NULL值处理

- **解决**: CSV中NULL以字符串"null"形式存在，需要字符串比较而非None判断

8.3 可能的改进方向

1. **性能优化**:

- 使用Combiner减少网络传输

- 调整HDFS块大小优化读取性能

2. **算法优化**:

- 任务三可以过滤低频优惠券减少计算量

- 任务四可以增加更多影响因素（时间周期、优惠券类型等）

3. **工程化改进**:

- 添加异常处理和日志记录

- 实现自动化测试脚本

- 使用配置文件管理参数

九、附录

附录A: 项目目录结构

```
bigdata-project2/
├── src/
│   ├── task1/
│   │   ├── mapper.py
│   │   └── reducer.py
│   ├── task2/
│   │   ├── mapper.py
│   │   └── reducer.py
│   ├── task3/
│   │   ├── mapper2.py
│   │   └── reducer2.py
│   └── task4/
│       ├── mapper_discount.py
│       ├── reducer_discount.py
```

```
│       ├── mapper_user_activity.py
│       ├── reducer_user_activity1.py
│       ├── mapper_user_activity2.py
│       └── reducer_user_activity2.py
├── output/           # 输出结果
├── logs/             # 日志文件
├── test_local.sh     # 本地测试脚本
└── 实验报告.md
```

附录B: 关键代码片段

(见各任务的详细设计部分)

附录C: 可视化图表

所有图表位于 `output/` 目录:

- `task1_visualization.png` - 商家优惠券使用情况
- `task2_visualization.png` - 商家距离分布
- `task3_visualization.png` - 使用间隔分布
- `task4_visualization.png` - 影响因素对比分析

报告完成时间: 2025年10月27日

最后更新: 2025年10月27日 17:30