实验 3-3 利用 pandaBI 查看数据各个维度的分布

建议课时：40 分钟

一、 实验目的

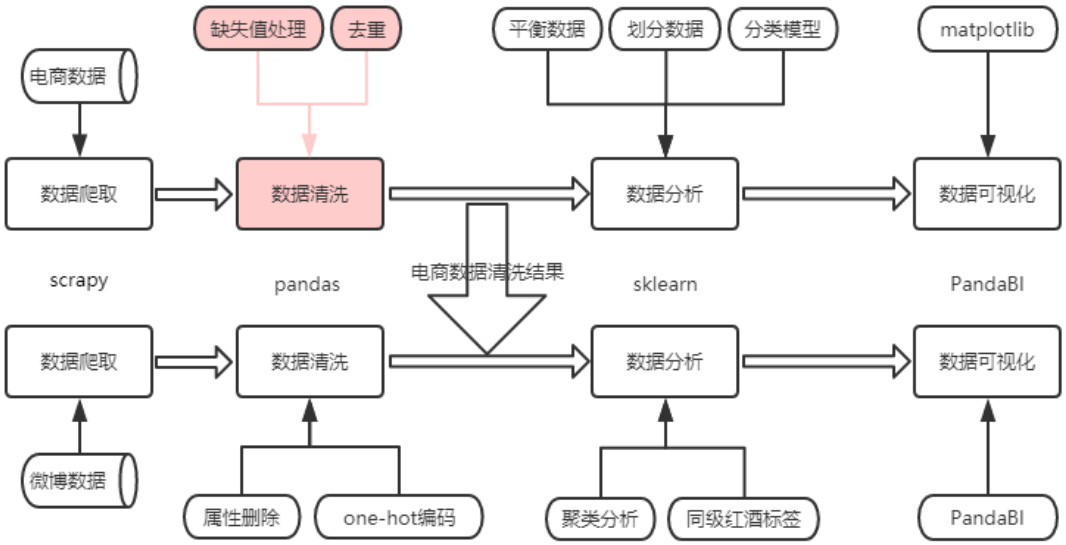
- 了解 PandaBI 的操作流程
- 了解数据清洗的基础方法

二、 实验环境

Python3 开发环境，第三方包有 pandas，工具有 PandaBI

三、 实验步骤

本节处理的内容有：



可利用 pandas 包中保存函数，将 3.2 节的最终处理结果进行保存为 pre-processed.csv，结合 pandaBI 可视化工具观察数据各个维度分布情况再做调整。本章首先简单介绍 pandaBI 的使用流程，再根据可视化结果对数据进行调整。

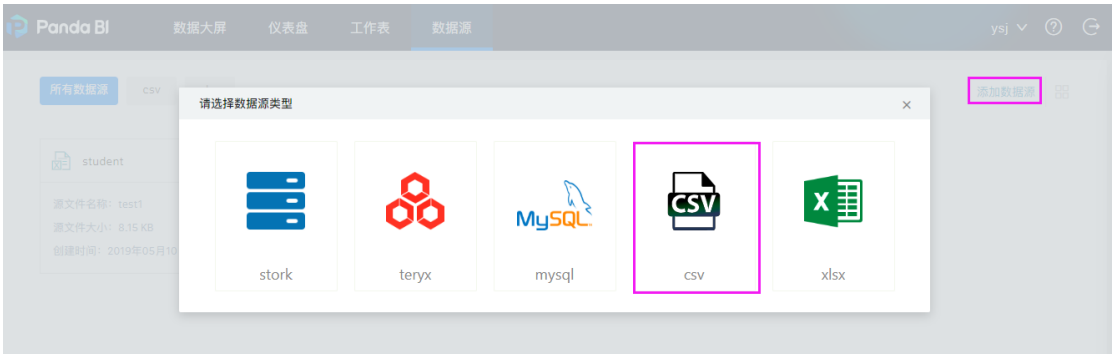
如上图所示：pandaBI 主要有以下四个模块：

- 数据大屏：用来做大屏展现（多个可视化分析结果的组合展现）
- 仪表盘：可视化结果展示（可用作页面嵌入）
- 工作表：进行多数据源的组合，也可以是单表数据
- 数据源：支持多数据源的导入

1. pandaBI 的使用流程

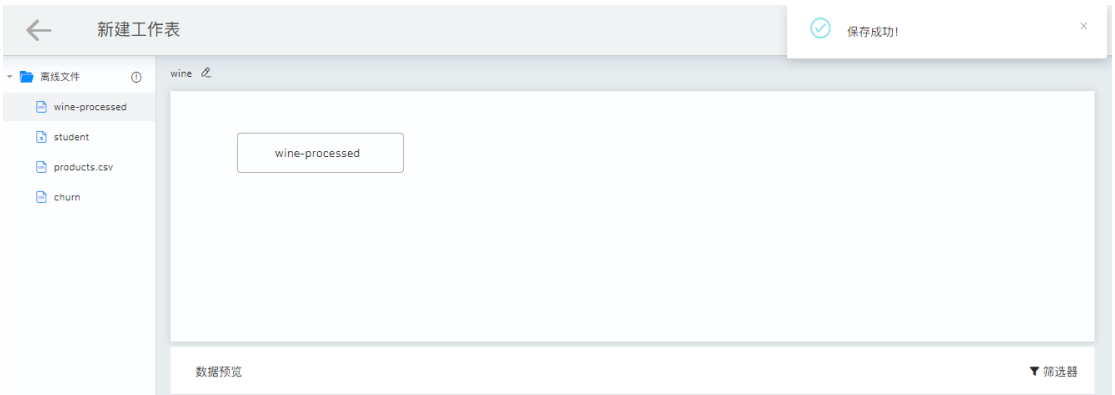
step1：导入数据

进入数据源模块 => 添加数据源 => 选择 csv => 填写数据源相关信息



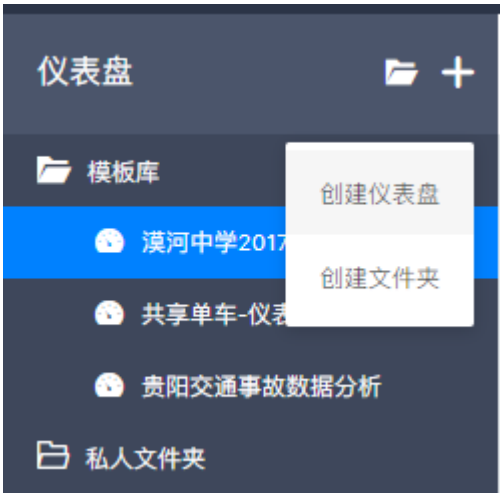
step2：构建数据表

进入工作表模块，我们只有一张表，不需要和其他表做连接，所以只要新建工作表，拖拽出刚才的数据源，然后保存即可。



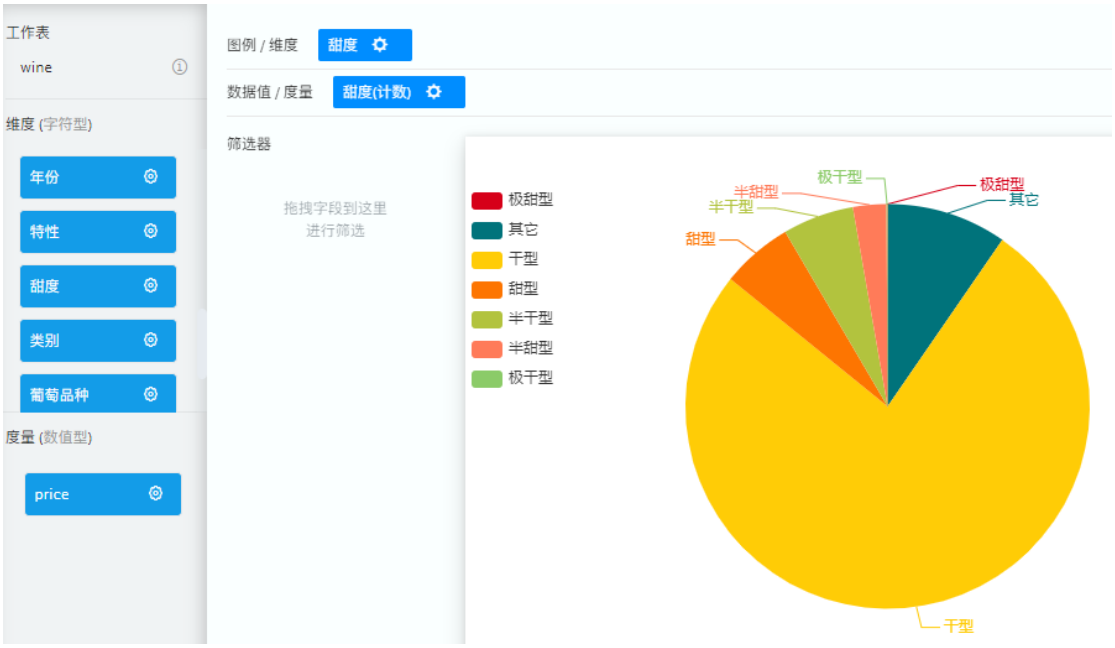
step3：可视化分析

进入仪表盘，即时可以开始进行对字段分析和探索了，数据大屏的展示和使用将在 3.3 节讲解。


仪表盘
模板库
创建仪表盘
创建文件夹
漠河中学2017
共享单车-仪表
贵阳交通事故数据分析
私人文件夹


新增图表
图表名称
图表1
仪表盘
wine-vision
工作表
wine
确定
取消

① 看甜度


工作表
wine
维度 (字符型)
年份
特性
甜度
类别
葡萄品种
度量 (数值型)
price
图例 / 维度
甜度
数据值 / 度量
甜度(计数)
筛选器
拖拽字段到这里
进行筛选
极甜型
其它
干型
甜型
半干型
半甜型
极干型

② 查看类别

同上可以查看其他字段，最终的判断结果如下：

- 存储方法：大部分数据是形容不能避光阴凉的不同说法，选择删除
- 原产地，甜度，颜色：含有部分取值为“其他”的数据，采用同一红酒品牌的众数替代策略予以调整
- 类别，葡萄品种：采用 onehot 编码的理论思想对数据进行转换
- 年份：数据中包含有很多“以实物为准”，“见瓶身”这种取值，但我们想知道的是酒的年龄，处理方式是利用文本处理的分词技术提取其中的年份，再与当前年份相减；部分数据的年份给的是一个范围值，此时取平均值；没有年份则用 9999 代替。
- 酒精度：同“年份”属性一样，有数据不规范，“实物为准”的取值问题，同样采用文本处理的分词技术提取酒精度的数值。
- 保质期：书写格式非常不统一，有些是按天为单位，有些是按年为单位，且有数字有中文的不同表达，该属性予以删除。

2. 处理特征

本节以”葡萄品种“，”酒精度“为例，讲解处理过程，其他属性的处理在前面小节的处理流程中均有涉及，学生可自行完成。

1. 处理 葡萄品种 的操作流程如下：

- 获取所有的葡萄品种类别
- 对每个葡萄品种新增一维来表示，假设共有 6 种葡萄品种，则需要 6 维来表示该属性值

代码如下：

```
# 处理葡萄品种
## 获取葡萄的所有品种类别
tmp = list(df.groupby("葡萄品种").count().index)
graps = []
for i in tmp:
    graps += i.split("|")
graps = list(set(graps))
print(graps)
del tmp

## 处理每个葡萄品种
for j in graps:
    df[j] = df['葡萄品种'].apply(lambda x : 1 if str(x).find(j) != -1 else 0)

# print(df[graps+['葡萄品种']].head)
df.drop("葡萄品种", axis=1, inplace=True)
```

2. 处理 酒精度 的操作流程如下：

- 利用 jieba 分词提取数值类型数据，用 float 转换
- 提取到两个数值，则代表是范围值，取平均数

代码如下：

```
# 处理酒精度
def deal_alcohol(x):
    numbers = []
    for w,p in pseg.cut(x):
        if p=='m':
            try:numbers.append(float(w))
            except:continue
    if len(numbers) > 1:
        return np.mean(numbers)
    elif len(numbers) == 1:
        return numbers[0]
    else:
        return 99999  # 酒精中无数字，以99999代替

df["alcohol"] = df["酒精度"].map(deal_alcohol)
df.drop("酒精度", axis=1, inplace=True)
```

综上，我们把数据清理完成，学生可后续再做些异常点处理的清洗工作，本案例暂不涉及。下一节处理标签，对价格区间进行定义和设置。