内容简介

本节通过数据数量和质量两方面，结合数据字段说明，完成对数据字段的检视工作，最终提出为了解决业务问题需要对现有数据进行的改造方案。

1. 主要内容

- 通过生成统计文档“summary.xlsx”的方式保存该案例 49 个字段的描述性统计；
- 在了解直方图、泊松分布、对数正态分布的基本概念后，对本案例中所有数值型字段的分布形状进行判断；
- 根据数据情况和业务问题，提出一系列数据改造方案。

2. 学习目标

学完本节，能解决以下问题：

- Pandas 中 Excel 数据表的导入与导出？
- 泊松分布和对数正态分布的特性是什么？
- 本案例的数据改造方案可以分为几大块？分别是什么？

数据质量

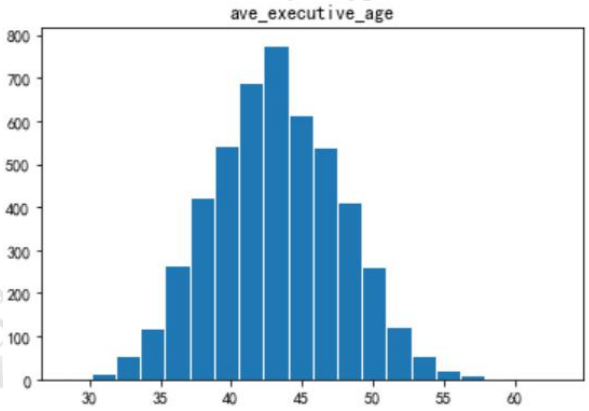
初级课程中我们通过数据的集中趋势、离散程度以及分布形状对数据的质量进行描述，通过对应的计算值和图表反映出数据的分布情况，这不仅是数据字段的检视常用的方法，同时对于数据探索以及数据结果的解释具有着重要的作用。

1. 直方图

在分布形状中我们了解了偏态系数、峰态系数以及正态分布这三个概念，而如果我们希望直观地查看字段的数据分布，往往需要借助直方图。

直方图：是一种频数分布图，它反映处在某一观测值范围内的观测数，可以直观地反映数据的总体分布。

如图所示：



在直方图中：

每个直方条下部的中点坐标是该观测值范围的中点；

直方条的宽度代表该观测值范围；

直方条的高度代表该观测值范围内的频数或百分比。

2. 数据的分布类型

根据变量的类型，数据的分布主要可以分为两类：

离散型统计分布和连续型统计分布

一般来说，定类、定序变量用数值表示后都属于离散型变量，定距、定比变量属于连续型变量。但是，一些特殊的定比变量，如人数，因为其只能取值为自然数，在数值的表示上依然属于离散型变量。

简单起见，区分离散和连续的方法就是：

取值只能为整数或自然数的就是离散型变量，不限于整数或自然数的就是连续型变量。

其中离散型统计分布包含：伯努利分布、二项分布、多项式分布、泊松分布等多种类型，而连续型统计分布包含：正态分布、对数正态分布、指数分布等多种类型，本次我们简单介绍**泊松分布**和**对数正态分布**。

（1）泊松分布

泊松分布（poisson distribution）：是用来描述在一指定时间范围内或在指定的面积或体积之内某一事件出现的次数的分布。

比如医院平均每 1 个小时接生 3 个婴儿，某呼叫中心平均 10 分钟接到 1 个电话等等。尽管我们知道了单位时间内事件发生次数的平均值，但是在给定的时长内，事件真正会发生多少次是未知的，但服从一定的概率分布。对于这种有固定发生频率的事件，其发生的概率就服从泊松分布。

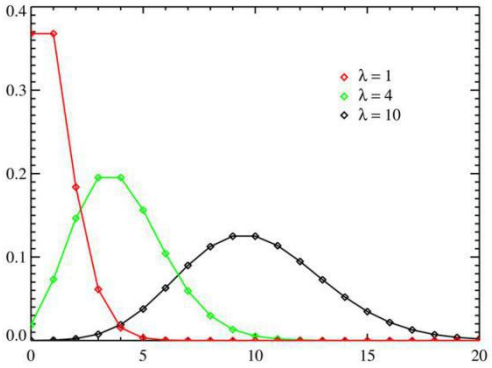
$P(X)=\frac{\lambda^x e^{-\lambda}}{x!}$，$x=0,1,2,\cdots$ 式中：

λ 为给定的时间间隔内的事件平均数；

X 表示事件实际发生的次数；

P(X)表示发生该次数的概率。

给定不同的平均值 λ，其泊松分布的曲线不同。如图所示：



泊松分布实际上不是一条曲线，而是多个可被一条曲线相连接的点，同时泊松分布的极限分布是正态分布。

（2）对数正态分布

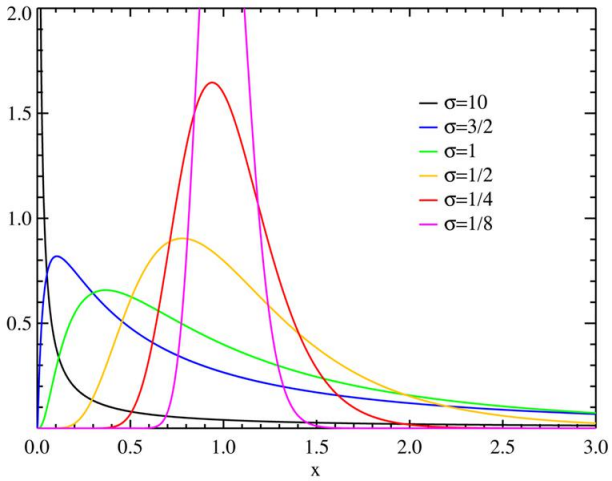
对数正态分布（log-normal distribution）:是指一个随机变量的对数服从正态分布，则该随机变量服从对数正态分布。

在很多自然现象中，变量分布不太可能像正态分布那样完全对称。例如人们完成马拉松所用的时间，很少有人能在 2.5 小时内跑完，多数业余选手能够在 4-5 小时内完成，还存在着一些选手完成马拉松需要花费很长的时间 7 小时甚至更长。生活还有很多情况是符合对数正态分布的，比如个人的收入情况、企业利润等。

之前介绍过正态分布，如果一个随机变量 X 服从一个数学期望为μ、方差为σ^2的正态分布，记为：$X \sim N(\mu,\sigma^2)$

μ：均值（反映集中程度）；σ^2：方差（反映离散趋势）

而若一个随机变量 X 的对数服从一个数学期望为μ、方差为σ^2的正态分布，记为：$\ln X \sim N(\mu,\sigma^2)$。不同条件下的对数正态分布如下图所示：



可以发现：对数正态分布属于右（正）偏态分布，向右小尾巴趋向足够长，同时对数正态分布也属于畸形的单峰连续数据分布。

对数：对数是对求幂的逆运算，如 $a^x = N$，那么 x 称为以 a 为底 N 的对数，记作：$x = \log_a N(a > 0 \& a \neq 1)$

当 $a = 10$ 时，记作：$x = \lg N$

当 $a = e$（e：自然常数，是一个无限不循环小数）时，记作：$x = \ln N$

提出数据改造方案

在检查数据数量、数据字段情况和数据质量之后，通常会发现一些数据不能直接应用于，需要进行一定程度的改造之后才可以用于数据分析，需要数据分析人员结合业务问题和数据情况提出具有操作性的数据改造方案。

本案例中现有数据需要进行的改造：

①提取模型输出组变量；②文本型字段转换为数值型；③缺失/空白和异常数据的清理；④逻辑回归模型对数据的要求；⑤数据分割；⑥判断训练集中是否存在非平衡数据并进行相应的处理

1. 明显可以放弃的字段（放弃字段均标灰）

在解决这些问题之前，我们首先将一些明显可以放弃的字段进行筛选：

在“数据获取”中了解到本案例共 49 个字段，明显可以放弃共 8 个字段：

部分字段放弃依据表

英文名称	特征名称	放弃依据
company_code	公司代码	公司标识，不参与数据建模
company_name	公司名称	
trading_days	交易日期	交易发生后字段，不适合用于预测分析
trading_price	交易平均价格	
total_transaction_amount	交易总金额	
financing_amount_wan	融资金额（万）	与本案例主业务问题无关联，用于后续案例
financing_cnt	融资次数	
outway	公司去向	

放弃后，剩余 41 个字段。

2. 模型输出组变量生成

本案例的主业务问题是“**根据公司特征分类公司是否会发生股票交易**”。

因此模型最终的输出组变量应该是一个二元变量（是/否），所有字段中没有直接可以对应的字段。

在放弃字段的时候我们发现有三个字段是发生交易后才会存在的字段：”trading_days”、”total_transaction_amount”、”trading_price”,这三个字段均有 3262 个非空取值，因此任取一个字段可以作为输出组变量的特征提取依据。

为统一案例，我们采用”trading_days”字段，通过：

空值 = 未发生交易　　非空值 = 发生交易

生成新字段”trading_happen”。

3. 文本型字段转换为数值型字段

逻辑回归模型不能直接处理文本型字段，sklearn 中提供了一种特征提取方法 onehot 函数，可以将文本型字段转换为数值型字段。

经数据检视了解到剩余 41 个字段中共 9 个文本型字段，**可在数据清理后进行转换**：

文本型字段转换为数值型字段依据表

英文名称	特征名称	取值类别	是否转换
transfer_mode	转让方式	2 种	是
layer	公司分层	2 种	
high_executive_edu	高管最高学历	4 种	
industry	第三级投资型行业	67 种	类别过多，不利于建模，放弃字段
province	省份	31 种	
accounting_firm	会计事务所	-	
local_accounting_firm	会计事务所	39 种	
broker	主办券商	-	
local_broker	主办券商	96 种	

注：onehot 特征提取方法举例说明：

现实变量

转让方式
协议
做市

onehot 方法 ⟹

虚拟变量

转让方式-协议	转让方式-做市
1	0
0	1

新引入的两个虚拟变量为二元变量——0=否，1=是；
协议(转让)：设置价格售卖或双方协商后进行转让，流通性比较差；
做市(交易)：通过做市商(批发零售商)进行的股票买卖行为，流通性比较强。

其中有 6 个字段由于类别过多，直接放弃，加上之前放弃字段，共放弃 **14** 个字段，剩余 **35** 个字段。

4. 数据清理

（1）缺失/空白分析

经过数据检视可以发现本案例中绝大部分字段都存在缺失情况。

处理方法：均值填充、众数填充和“0”值填充等。

①均值填充：**连续型变量。**剩余 35 个字段中包括以下 26 个连续型字段：

均值填充依据表

英文名称	特征名称	英文名称	特征名称
listing_days	挂牌时长	liability	负债总计
registered_capital	注册资本	net_asset_per_share	每股净资产
register_days	注册时长	earning_per_share	基本每股收益
customer_rt_1st	第一大客户占比	asset_liablity_rt	资产负债率
customer_rt_5all	五大客户占比	current_rt	流动比率
supplier_rt_1st	第一大供应商占比	gross_cash_flow	现金流量总额
supplier_rt_5all	五大供应商占比	asset_gr	总资产增长率
total_stock_equity	总股本	income_gr	营业收入增长率
business_income	营业收入	net_profit_gr	净利润增长率
gross_profit_rt	毛利率	ave_executive_age	高管平均年龄
net_profit_rt	净利润	ave_executive_edu	高管平均学历
roa	净资产收益率	holder_rt_1st	第一大股东占比
asset	资产总计	holder_rt_10all	前十大股东占比

②众数填充：**非连续型变量。**都包括以下共 8 个字段：

众数填充表

英文名称	特征名称
transfer_mode	转让方式
layer	公司分层
maker_num	做市商数量
employee_num	雇员数量
board_num	董事会人数
supervisor_num	监事会人数
executive_num	高管人数
highest_executive_edu	高管最高学历

③“0”值填充：**二元变量**：

英文名称	特征名称
contain_org_holder	存在机构股东

注：“contain_org_holder”字段取值为 0、1，该字段缺失默认其不存在机构股东

缺失值填充需完成：26 个均值填充+8 个众数填充+1 个“0”值填充=35 个字段

（2）异常值分析

异常值指的是数据字段的取值超出合理范围，需要根据业务问题进行人工修正。

根据“数据字段说明.xlsx”和“summary.xlsx”对以下共 8 个字段进行人为修改：

异常值人工修改依据表

英文名称	特征名称	合理范围
customer_rt_5all	五大客户占比	占比取值为[0,1]
supplier_rt_5all	五大供应商占比	
gross_profit_rt	毛利率	极差巨大&大多数据集中于[-1,1]区间内
net_profit_rt	净利润	
roa	净资产收益率	
asset_gr	总资产增长率	极差巨大&大多数据集中于[-1,5]区间内
income_gr	营业收入增长率	
net_profit_gr	净利润增长率	极差巨大&大多数据集中于[-10,10]区间内

5. 逻辑回归对数据的要求

初级课程中我们介绍，k-means 算法在建模前，需要通过归一化对输入组变量进行无量纲化处理（去除字段单位，方便不同量纲字段之间进行计算）。逻辑回归模型也同样需要这步操作，同时在理想情况下，逻辑回归模型希望所有的输入变量组都服从标准正态分布。

在“数据质量-2”中，我们对 38 个数值型字段进行分布类型查看，得到如下情况：

数值型字段分布情况统计表

分布类型	英文名称	特征名称	分布类型	英文名称	特征名称
近似泊松分布（4 个）	maker_num	做市商数量		liability	负债总计
	employee_num	雇员数量		gross_cash_flow	现金流量总额
	board_num	董事会人数	特殊分布	holder_rt_10all	前十大股东占比
	executive_num	高管人数	其他分布（9 个）	supervisor_num	监事会人数
近似正态分布（7 个）	customer_rt_5all	五大客户占比		gross_profit_rt	毛利率
	supplier_rt_5all	五大供应商占比		net_profit_rt	净利润
	net_asset_per_share	每股净资产		roa	净资产收益率
	earning_per_share	基本每股收益		current_rt	流动比率
	asset_liablity_rt	资产负债率		asset_gr	总资产增长率
	ave_executive_age	高管平均年龄		income_gr	营业收入增长率
	holder_rt_1st	第一大股东占比		net_profit_gr	净利润增长率
近似对数正态分布（10 个）	listing_days	挂牌时长		ave_executive_edu	高管平均学历
	registered_capital	注册资本	放弃字段（5 个）	financing_amount_wan	融资金额（万）
	register_days	注册时长		financing_cnt	融资次数
	customer_rt_1st	第一大客户占比		total_transaction_amount	交易总金额
	supplier_rt_1st	第一大供应商占比		trading_days	交易日期
	total_stock_equity	总股本		trading_price	交易平均价格
	business_income	营业收入	（文本型）	highest_executive_edu	高管最高学历
	asset	资产总计	"0"填充	contain_org_holder	存在机构股东

注：”highest_executive_edu”字段取值为 1,2,3,4，但仅数值表示类型，为文本型；而与之关联的字段”ave_executive_edu”是运算后取值在[1,3.5]的连续型变量；

小提示：通过整理，可以发现这 38 个字段中用于本次案例中数据建模的共 38-5=33 个字段，但是在“2.缺失/空白”中我们分析最终是需要 35 个字段，剩余两个字段去哪呢？其实在文本型数据中还有两个字段不要落下。

无量纲化处理的方法：除了初级课程中提到，导入 sklearn 中的 MinMaxScaler()函数，将数据缩放到一定范围的线性归一化方法外；还有一种除了可以实现去量纲和一定程度的缩放，更侧重于将数据处理为均值为 0，方差为 1 的标准正态分布的方法：**标准化**，这也正好满足了逻辑回归对输入组变量分布的希望。

在之前操作过程中，3 个文本型字段和 1 个“0”值填充字段均具有二元变量属性，不太需要标准化处理。

除此以外，对于剩余的 31 个字段，sklearn 提供了 StandardScaler()函数的标准化处理方法，公式为：

$$y_i = \frac{x_i - \bar{x}}{\sigma^2}$$

由于可以将这些字段转换为符合标准正态分布的分布情况。

sklearn 同时提供了 PowerTransformer()函数，通过设置其中的参数 standardize 可以实现标准化功能，同时处理后的特征的取值范围更加的一致，受异常值/极端值的影响会更小，更有利于模型达到更好的效果。但是 PowerTransformer()函数只适用于单峰连续分布。“数据质量”中我们了解对数正态分布是单峰连续分布的一种，

为了模型效果更佳，使用 PowerTransformer()将 10 个对数正态分布字段进行转变。

字段“holder_rt_10all”中取值大部分集中在 0.9-1.0 之间，选择通过线性归一化进行无量钢化处理，同时也不太需要标准化处理。

至此，按照操作可以完成逻辑回归相关字段的预处理工作。

6. 数据分割

在处理完 35 个输入组变量和 1 个输出组变量后 **数据预处理**的工作就已经完成了。由于逻辑回归算法和决策树算法相同，都为有监督学习，因此接下来需要对数据集进行分割，本次分割方法采用最简单的“直接下定义”，定义训练集和测试集的样本个数。

直接定义训练集和测试集的样本个数的方法：sklearn 包中提供了 train_test_split()函数

7. 训练集中是否存在非平衡数据？

平衡数据：在分类问题中，每个类别有大致相等的样本数，此时不会因为某一类中由于样本数过多，而模型对特征的选择有所偏向；反之当各类间的样本个数差异较大时，模型在对数据特征的选择就会有所偏向，例如某一类样本数远超过其他类别，模型只需要全部输出大类对应标签即可达到较高准确率，此时该数据集称为**非平衡数据**。

但是为了保证模型对于每一个类的分类准确性都相近，整体效果最优，我们需要判断训练集中是否存在非平衡数据。

本案例训练集分为两类：未发生交易/发生交易，具体情况如下表：

训练集中各类样本数

未发生交易	1356	小类
发生交易	2596	大类

本次案例的训练集为非平衡数据，处理非平衡数据主要有以下两种方法：重采样和亚采样，重采样是指从小类中再次进行随机抽取，直至样本数与大类样本数相同，在本案例中即为在未发生交易这类中再随机抽取 2596-1356=1240（个）样本；而反之则为亚采样：从大类中随机抽取出于小类样本数相同的样本数。

而无论重采样还是亚采样一般称之为**数据重构**。

本次案例中样本数量并非庞大，同时为了减少数据浪费，我们使用重采样的数据重构方法。

注:pandas 中提供了 balance_resample 重采样方法。

最后，将数据改造方案中的所有需要完成的操作，以字段为依据在 EXCEL 中进行整理，得到“数据字段处理记录.xlsx”,放置压缩包中，请下载并认真查看思考，回答练习中的问题。

数据检视总结

1、如何生成字段描述性统计文档？

（1）创建 ExcelWriter 对象

相当于在桌面创建了一个 EXCEL 表格，例如案例中的代码：

summary_writer = pd.ExcelWriter('summary.xlsx')

创建了一个名为“summary”的 xlsx 格式表格，并命名为 summary_writer

（2）将统计数据保存到数据表中

将数据集的描述性统计生成的数据存放在表格里一个 sheet 中，并对其命名，例如案例中的代码：

features.describe(include='all').to_excel(summary_writer, sheet_name='describe')

对数据集 features 进行描述性统计，并将其存放于 summary_writer 表中的一个 sheet 中，该 sheet 命名为“describe”

（3）执行

相当于该表格保存的步骤，例如案例中的代码：

summary_writer.save()

2、泊松分布和对数正态分布的特性是什么？

（1）泊松分布

泊松分布（poisson distribution）：是用来描述在一指定时间范围内或在指定的面积或体积之内某一事件出现的次数的分布，同时泊松分布的极限分布是正态分布。

它描述的是离散型数据的分布。一般情况下有固定发生频率的事件，其发生的概率就服从泊松分布。直方图上趋向于正态分布的图形，但泊松分布实际上不是一条曲线，

而是多个可被一条曲线相连接的点。

（2）对数正态分布

对数正态分布（log-normal distribution）:是指一个随机变量的对数服从正态分布，则该随机变量服从对数正态分布。

它描述的是连续型变量，从图形上来看，对数正态分布属于右（正）偏态分布，向右小尾巴趋向足够长，同时对数正态分布也属于畸形的单峰连续数据分布。

3、本案例的数据改造方案可以分为几大块？分别是什么？

参考答案（可以根据自己理解进行总结）:

本案例的数据改造方案可以分为 3 大块，分别为：①直接放弃无关字段②数据预处理③数据分割与数据重构

（1）直接放弃无关字段

根据对每个字段的审查放弃无关字段并给出理由，本案例主要放弃三部分字段，2 个编号标识字段，3 个交易发生后生成的字段和 3 个与本案例无关字段。

（2）数据预处理

在数据预处理部分又可以分为三部分内容：

- 常规的数据清理工作。包括对缺失值处理和异常值处理
- 根据模型和主业务问题的要求明确输出组变量。
- 根据模型的要求对输入组变量进行数据转换。

①数据清理

其中缺失值分析中，对案例的 35 个字段分别采用了：

处理方法	字段个数	依据
均值填充	26	连续型变量
众数填充	8	非连续性变量

“0”值填充	1	二元变量
总计	35	-

异常值分析中有 8 个字段要么取值超出合理范围，要么存在极值，需要人工修正。

②明确输出组变量

也可以称为提取新特征，是将现有字段中可以替代输出组变量的字段进行特征提取。最后生成符合本案例逻辑回归模型的二元变量。

③根据模型对输入组变量进行数据转换

逻辑回归模型对输入组变量有 3 点要求：数值型数据、无量纲、服从标准正态分布

首先对于文本型的字段需要转换为数值型字段：

有 9 个文本型字段，其中 6 个字段类别过多，不适宜转换，因此舍弃，其他 3 个字段转换后生成虚拟变量，均属于二元变量（无量纲），转换方法为 onehot()函数，针对逻辑回归模型，二元变量不太需要转换为标准正态分布。

数据标准化在实现去量纲和一定程度的缩放外，更侧重于将数据处理为均值为 0，方差为 1 的标准正态分布。

标准化有两种方法：

sklearn 同时提供了 PowerTransformer()函数，通过设置其中的参数 standardize 可以实现标准化功能，同时处理后的特征的取值范围更加的一致，受异常值/极端值的影响会更小，更有利于模型达到更好的效果。但是 PowerTransformer()函数只适用于单峰连续分布；

sklearn 提供了 StandardScaler()函数的标准化处理方法。

因此决定对于服从近似对数正态分布的 10 个字段使用 PowerTransformer()函数；

剩余 7 个服从近似正态分布和 4 个泊松分布字段使用 StandardScaler()函数。

有 1 个特殊分布字段“holder_rt_10all”中取值大部分集中在 0.9-1.0 之间，选择通过线性归一化进行无量纲化处理，同时也不太需要标准化处理。

最后还有一个“0”值填充的字段 contain_org_holder 原本就是二元变量，故不做考虑数据转换。

（3）数据分割与数据重构

数据分割：

本案例采用“直接下定义”法，使用 train_test_split()函数直接定义测试集样本数量。

数据重构：

非平衡数据将影响模型质量，为了保证模型对于每一个类的分类准确性都相近，整体效果最优，要进行数据重构。

数据重构方法有两种：

重采样：从小类中再次进行随机抽取，直至样本数与大类样本数相同；

亚采样：从大类中随机抽取出于小类样本数相同的样本数。

内容简介

本节主要依据数据改造方案完成数据预处理工作。

1. 主要内容

- 提取新特征；
- 分类特征的数据清理及数据转换等操作。

2. 学习目标

学完本节，能解决以下问题：

- 如何将通过 notnull()函数提取数据新特征并转换为 0/1 的浮点型数据？
- 如何使用 DataFrameMapper 将 sklearn 的数据转换方法用在 DataFrame 中？
- 案例中共使用了几种 sklearn 数据预处理方法？分别是什么？

数据预处理总结

1、如何将通过 notnull()函数提取数据新特征并转换为 0/1 的浮点型数据？

pandas 提供的 notnull()函数，可以将某字段中有取值就返回 True，没取值就返回 False；其次使用 astype()函数修改数据类型为浮点数（float）类型，例如案例中的代码：

features['trading_happen'] = features['trading_days'].notnull().astype(float)

先使用 notnull()函数检查字段 trading_days 中有无取值，有取值的返回 True，无取值的返回 False；

然后再使用 astype()函数将新特征转换为浮点型数据，即：True=1，False=0；

最后将得到的结果赋值给 features 表中新字段 trading_happen。

2、如何使用 DataFrameMapper 将 sklearn 的数据转换方法用在 DataFrame 中？

在数据预处理中，sklearn 中的数据转换方法并不能直接在 DataFrame 中进行使用，需要通过使用 DataFrameMapper 定义数据转换方法，然后使用 fit_transform 执行该定义好的转换方法。具体操作步骤如下：

（1）导入 DataFrameMapper

从 sklearn_pandas 包导入 DataFrameMapper，例如代码：

from sklearn_pandas import DataFrameMapper

（2）定义数据转换方法

在 DataFrameMapper()括号中定义好各字段的转换方法及输出格式，例如案例中的代码：

x_mapper = DataFrameMapper([([字段],[转换器]), ...], df_out=False)

将输出内容命名为 *x_mapper*。

其中，转换器即转换函数，如 PowerTransformer()、StandardScaler()等。

df_out 为数据格式：当 df_out 取值 True 时，输出为 DataFrame 格式；

取值为 False 时，输出为 numpy 数组形式。

（3）执行数据转换

在定义好了每一个特征的处理方法后，使用 fit_transform 执行数据转换操作，例如代码：

X = x_mapper.fit_transform(features.copy())

在 features 这个数据表的副本上执行数据转换操作，将得到的数据命名为 X。

注:这里只会对已经定义好转换方式的特征进行转换并输出,对于那些没有定义转换方式的特征,就被抛弃了，输出结果 X 中是没有这些特征的。

3、案例中共使用了几种 sklearn 数据预处理方法？分别是什么？

本案例共使用了 6 种 sklearn 数据预处理方法，其中 1 种数据填充方法：SimpleImputer()，5 种数据转换方法：OneHotEncoder()、PowerTransformer()、StandardScaler()、MinMaxScaler()、FunctionTransformer()。

（1）填补缺失值　SimpleImputer()

SimpleImputer()是 sklearn.impute 包下一种方法，因此导入代码：

from sklearn.impute import SimpleImputer

设置 strategy 参数来定义填补方法，mean-用均值填补（该参数默认值为 mean，因此可以不修改），most_frequent-用众数填补。“0”值填充设置 fill_value=0 即可，例如案例中代码：

(['listing_days'], [SimpleImputer(), PowerTransformer()])

对连续型变量 listing_days 填充均值；

(['maker_num'],[SimpleImputer(strategy='most_frequent'),StandardScaler()])

对离散型变量 maker_num 填充众数；

(['contain_org_holder'], SimpleImputer(fill_value=0))

对二元变量 contain_org_holder 填充“0”值。

（2）文本型转换为数值型　OneHotEncoder()

OneHotEncoder()是 sklearn.preprocessing 包下一种方法，因此导入代码：

from sklearn.preprocessing import OneHotEncoder

我们需要设置 OneHotEncoder()中的两个参数 categories 和 sparse，例如案例中的代码：

(['layer'],[SimpleImputer(strategy='most_frequent'),OneHotEncoder(categories='auto', sparse=False)])

对公司分层 layer 字段众数填充后，转换为数值型字段，categories 是指确定类别的方式，取值’auto’:自动确定，经自动确认后，公司分层字段取值有两类：基础层和创新层，会生成 2 个虚拟二元变量。Sparse=True 时，会返回一个稀疏矩阵；Sparse=False 时，会返回数组。这里将 layer 字段返回为数组。

（3）畸形的单峰连续数据分布→标准正态分布　PowerTransformer()

PowerTransformer()也是 sklearn.preprocessing 包下一种方法，导入代码相同，使用默认参数，例如案例中的代码：

(['listing_days'], [SimpleImputer(), PowerTransformer()])

将近似服从对数分布的字段 listing_days 均值填充后，转换为标准正态分布。

（4）将一般分布转换为标准正态分布　StandardScaler()

StandardScaler()也是 sklearn.preprocessing 包下一种方法，导入代码相同，使用默认参数，例如案例中的代码：

(['earning_per_share'], [SimpleImputer(), StandardScaler()])

将近似服从正态分布的字段 earning_per_share 均值填充后 ,转换为标准正态分布。

（5）线性归一化　MinMaxScaler()

（6）自定义函数　FunctionTransformer()

StandardScaler()也是 sklearn.preprocessing 包下一种方法，也需要进行导入。

我们需要将异常数据压缩至一个限定的区间范围 ,采用自定义一个函数来实现这个效果，例如案例中定义函数 truncate_wrapper 的代码：

```
# 将连续分布的特征压缩到[l,r]区间中
def truncate_wrapper(l, r):
    if l >= r:
        raise ValueError("Left value {} should be smaller than Right value {}".format(l, r))

    def truncate(v):
        v[v < l] = l
        v[v > r] = r
        return v

    return truncate
```

其中，l：区间左边界、r：区间的右边界；如果 l≥r，报错：左侧的值应该比右侧的值小；如果这两个值的设置没有问题，就调用这个函数，将小于最小值的数值修正为最小值，将大于最大值的数值修正为最大值，最后返回修正后的数据。

随后根据情况使用这个函数来处理数据，例如案例中的代码：

(['customer_rt_5all'],[SimpleImputer(),FunctionTransformer(truncate_wrapper(0, 1), validate=True), StandardScaler()])

customer_rt_5all 的合理范围为[0,1]，对该字段填充均值后，使用自定义函数

truncate_wrapper 设置区间[0,1]后，进行标准化，参数 validate=True 表示：在调用自定义函数前，要将输入数据转换为二维 numpy 数组或稀疏矩阵。

除此以外，在数据预处理中，可能会出现因数值过大溢出导致 PowerTransform 出现警告的问题，此时可以采用同字段数值同比缩小的方法进行处理。

数据分割与重构作业

在本节作业中您需要完成以下内容：

（1）使用 train_test_split()函数对数据集进行分割；

（2）通过数据重构将训练集重构为平衡训练集。

数据分割与重构总结

1、在 Python 中如何使用 train_test_split()函数对数据集进行分割？

train_test_split()函数是 sklearn 包中 model_selection 模块下的一个数据分割方法，首先需要导入该函数，代码：

```
from sklearn.model_selection import train_test_split
```

其次使用该函数将数据集拆分为训练集和测试集，例如案例中的代码：

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=1000,random_state=0)
```

其中 X，y 分别为数据预处理后的输入组变量和输出组变量，test_size 为测试集样本个数，这里我们设置的样本个数为 1000。通过该函数生成的训练集和测试集分别保存为 X_train,X_test,y_train,y_test，注意这四个名称顺序是固定的。

2、如何通过重采样的方法实现数据重构？

重采样或者亚采样都是为了处理数据集不平衡问题，通过查看案例中训练集取值为 1 的样本 2596 个，取值为 0 的样本 1356 个，属于数据不平衡的情况。

重采样是指从小类中再次进行随机抽取，直至样本数与大类样本数相同。具体操作分为三步：

①确定重采样的样本数量；

②随机采样；

③重构连接组成新的平衡的训练集。

例如案例中的代码：

```
# 样本数量
num_sml = (y_train==0).sum()
num_big = (y_train==1).sum()
# 确定需要重采样的数量
num_samples = num_big - num_sml

# 大类样本
X_big = X_train[y_train == 1]
y_big = y_train[y_train == 1]
#小类样本
X_sml = X_train[y_train == 0]
y_sml = y_train[y_train == 0]

# 随机采样
random_state = np.random.RandomState(seed=0)
indx = random_state.randint(low=0, high=num_sml, size=num_samples)
X_sampl = X_sml[indx]
y_sampl = y_sml[indx]

# 连接
X_train_bal = np.vstack((X_big, X_sml, X_sampl))
y_train_bal = np.hstack((y_big, y_sml, y_sampl))
```

这样新生成的 X_train_bal 和 y_train_bal 就组成了新的平衡训练集。