目录

8 离群值处理

离群值(outlier) 有时也被称为异常值，通常出现在定量变量之中. 目前尚缺乏关于离群值的严格定义，Hawkins (1980) 给出了离群值的一个直观定义:

定义 8.1 离群值是一个观测值，它偏离了其他观测值以至于怀疑它产生于不同机制.

按照这一定义，那些与绝大多数数据值明显不协调的数值即为离群值[1]. 离群值属于噪声数据，对数据分析可能产生一定的影响，包括:

(1) 离群值可能影响数据预处理效果. 例如，在数据规范化(标准化或最小值-最大值规范化) 中，离群值可能导致大部分数据集中在一个小范围之内.

(2) 离群值可能影响分析的准确性. 根据受离群值影响的程度，统计学方法可以分为两类: 不易受离群值影响的方法属于稳健的(robust) 方法，否则为非稳健的方法. 也就是说，离群值对于非稳健的方法具有一定的影响. 例如，普通最小二乘回归属于非稳健的方法，离群值可能对其分析结果产生较大的影响.

因此, 有必要在数据预处理阶段识别离群值, 并对离群值进行一定的处理.

8.1 离群值处理概述

一般，离群值处理中的关键是识别出离群值. 实际业务中，离群值可能由欺诈所致，识别离群值也成为欺诈检测算法(fraud detection) 的分析目标[2]. 数据预处理中常用的离群值识别方法主要有3类:

(1) 基于分布的(distribution-based) 方法. 这一类方法利用小概率原理，即当落入一个概率较小的取值范围时，数据值为离群值的可能性较大，从而被识别为离群值. 典型的方法有3sigma原则、箱线图和椭圆包络.

(2) 第二类是基于密度(density-based) 或距离的(distance-based) 方法. 这一类方法基于数据点之间的聚集情况，将远离大部分数据点聚集区域的数值识别为离群值. 典型的方法有局部离群点因子和聚类方法.

(3) 第三类是基于模型的(model-based) 方法. 这一类方法的特点是需要借助于模型识别离群值，典型的方法有孤立森林和单类支持向量机.

[1]那些偏离既定模型很远的数值也可以看作离群值，但因为数据预处理中往往不涉及某种既定模型，所以此处不讨论这类离群值.

[2]有许多反欺诈算法可用于识别离群值，但超出了本书范围. 本书仅介绍通用的离群值识别方法，并非专门针对反欺诈问题.

本章后续各小节将分别介绍几种常用的离群值识别方法[3].

对于识别出的离群值，需要分析导致离群值的原因，并结合后续数据分析要求进行处理，例如:

(1) 保留离群值. 若在后续的数据分析中可以采用稳健的方法，那么离群值的影响较为有限，可以考虑保留离群值，从而尽量保留数据信息.

(2) 替换离群值. 可采用一定的方法替换离群值，例如，通过数据离散化方法对定量变量分箱(此时，定量变量转换为区间取值的定序变量)，以一个区间替换离群值; 或者以缺失值替换离群值, 随后可结合缺失值处理方法进一步处理.

(3) 删除离群值. 若离群值对后续数据分析可能存在较大影响，可以考虑删除离群值，也即删除含有离群值的样本观测.

8.2 一元变量的离群值识别

本节介绍两种一元变量的离群值识别方法——3sigma原则和箱线图, 它们都是基于分布的方法.

8.2.1 3sigma原则

由概率论的知识, 若随机变量服从均值为μ、标准差为σ的正态分布, 则变量取值落入$\mu \pm 3\sigma$以外的概率约为0.3%, 是一个小概率. 根据小概率原理, 我们认为在一次试验中, 小概率事件不会发生, 因此对变量取值落入$\mu \pm 3\sigma$以外的事件产生怀疑, 认为这些取值可能"非正常", 从而将它们判别为离群值.

注记 8.1 *3sigma*原则针对服从正态分布的变量. 若变量不服从正态分布, 需要仔细考察变量取值落入$\mu \pm 3\sigma$以外的概率是否为小概率, 只有在小概率情况下, *3sigma*原则才较为有效.

注记 8.2 应用*3sigma*原则时, 需要μ和σ两个参数值, 当它们未知时一般用样本均值和样本标准差予以估计. 但是, 样本均值和样本标准差会受到离群值的影响, 导致估计不准确, 从而影响离群值识别的效果.

例 8.1 从二元正态分布$N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & -0.5 \\ -0.5 & 1 \end{pmatrix}\right)$中产生4个随机数*(*保留到1位小数*)*：$(-1.3, 1.7)$, $(0.3, 2.0)$, $(-2.1, 1.1)$, $(-0.9, 0.7)$. 在此基础上

[3]采用聚类分析方法, 将样本量较少类别中的样本观测识别为离群值, 这是离群值识别的一种直观方法. 它所涉及的技术主要是聚类分析, 这里不作介绍, 感兴趣的读者请查阅多元统计分析相关书籍.

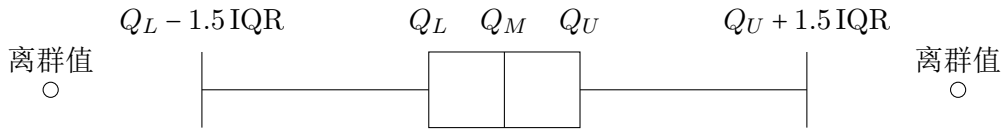


图 8.1: 箱线图

加入一个数据点$(10, 10)$, 直观上看, 它离其他*4*个数据点较远, 可被视为离群值. 基于这*5*个样本观测, 对*2*个维度分别计算样本均值和样本标准差, 得到

$$\hat{\mu}_1 = 1.2,\ \hat{\sigma}_1 \approx 4.99,\ \hat{\mu}_2 = 3.1,\ \hat{\sigma}_2 \approx 3.89.$$

在第*1*个维度上, $\hat{\mu}_1 \pm 3\hat{\sigma}_1 = [-13.77, 16.17]$; 在第*2*个维度上, $\hat{\mu}_2 \pm 3\hat{\sigma}_2 = [-8.57, 14.77]$. 根据*3sigma*原则, 数据点*(10,10)*在两个维度上都未被识别为离群值.

从这个例子可以看出, μ和σ的估计容易受到离群值的影响, 从而可能导致应用3sigma原则时无法灵敏地识别出离群值. 此外, 忽略多维变量中各维度的关联性、独立地在各个维度上识别离群值, 可能也会影响离群值识别的效果. 如何解决这些问题呢？请读者积极思考, 我们将在8.3节椭圆包络方法中介绍一种解决方案.

8.2.2 箱线图

箱线图(Box Plot) 方法由Tukey (1977) 提出. 箱线图由1个箱体和2根线条构成, 因此得名“箱线图”.

箱线图有多种形式, 常用的一种形式是基于4个统计量, 分别是下四分位数Q_L、中位数Q_M、上四分位数Q_U, 以及四分位距IQR (Inter Quartile Range), 其中, Q_L、Q_M和Q_U分别是数据的25%、50%和75%分位数, IQR= $Q_U - Q_L$用以刻画中间50%数据的离散程度. 在箱线图中, 我们将Q_L和Q_U连接构成一个箱体(box), 并在中位数的位置划一竖线予以标识; 从箱体两端出发, 各延申1.5倍[4]IQR, 作出线条(whisker). 落入线条与箱体范围内的数值, 即$[Q_L - 1.5\,\mathrm{IQR},\ Q_U + 1.5\,\mathrm{IQR}]$的数值, 被认为是正常值; 而未落入$[Q_L - 1.5\,\mathrm{IQR},\ Q_U + 1.5\,\mathrm{IQR}]$的数值被看作离群值, 如图8.1所示. 当变量服从正态分布时, 其取值未落入$[Q_L - 1.5\,\mathrm{IQR},\ Q_U + 1.5\,\mathrm{IQR}]$的概率约为0.7%, 是一个小概率. 此时, 箱线图方法符合小概率原理.

[4]1.5倍并非由严格的推导获得, 只是经验值, 参见文献[2].

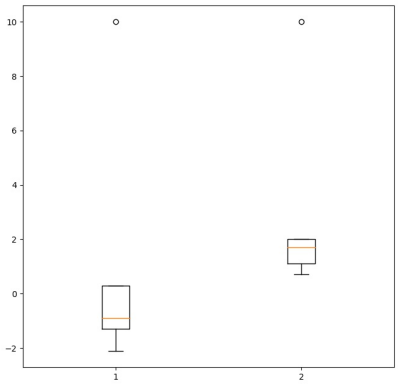


图 8.2: 5个数据点形成的箱线图

注记 8.3 需要注意*:* 当变量不服从正态分布时, 变量取值未落入$[Q_L - 1.5\,IQR,\ Q_U + 1.5\,IQR]$的概率不一定是小概率, 此时所识别的离群值不一定准确. 对称型的厚尾分布或非对称型分布都可能出现这种问题. 例如, 当变量服从均值为*0*、标准差为*1*的对数正态分布时, 取值大于$Q_U + 1.5\,IQR$的概率超过*7%*. 针对这一问题, 已经有一些解决方案*(*如文献*[3])*, 这里不作展开.

由于箱线图中所使用的统计量较为稳健, 因此箱线图不太容易受到离群值的影响. 下面的例子体现了这一点.

例 8.2 基于例*8.1*中的*5*个样本观测, 分别得到两个维度的分位数值[5]*:*

$$Q_{L1} = -1.3, \quad Q_{M1} = -0.9, \quad Q_{U1} = 0.3,$$
$$Q_{L2} = 1.1, \quad Q_{M2} = 1.7, \quad Q_{U2} = 2.0.$$

由此, 得到箱线图*(*如图*8.2*所示*)*. 数据值$(10, 10)$在两个维度上都未落入$[Q_L - 1.5\,IQR,\ Q_U + 1.5\,IQR]$, 因而被识别为离群值. 这体现了箱线图利用稳健统计量识别离群值的优势.

8.3 椭圆包络

椭圆包络(Elliptic Envelope) 类似于3sigma原则, 根据变量的分布找出小概

[5]分位数有多种计算方法. 这里的分位数按照Python numpy库中percentile函数获得, 方法是将最小值和最大值分别作为0%和100%位置, 然后在剩余的数值中(假定有k个数值) , 以$(k+1)$乘以指定的百分位作为分位数的位置, 通过线性插补获得分位数.

率区域, 将落入小概率区域的数值识别为离群值. 这里的关键问题是如何找出小概率区域.

回顾3sigma原则中的小概率区域, 它是$\mu \pm 3\sigma$范围以外的区域. 当$\sigma > 0$时, 这一小概率区域可表示为

$$\left\{x : \frac{(x-\mu)^2}{\sigma^2} > 3\right\}.$$

类似地, 对于多元随机变量$\boldsymbol{X}$, 可以通过$(\boldsymbol{X}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{X}-\boldsymbol{\mu})$构造小概率区域, 其中$\boldsymbol{\mu}$和$\boldsymbol{\Sigma}$分别为$\boldsymbol{X}$的均值(为列向量) 和协方差矩阵(假定为可逆矩阵). 设置临界值$\gamma(>0)$, 可以构造小概率区域:

$$\left\{\boldsymbol{x} : (\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) > \gamma\right\}.$$

可以看到, 离群值区域与正常值区域的分界线由满足$\left\{\boldsymbol{x} : (\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) = \gamma\right\}$数值构成, 它们恰好构成一个椭球面. 因此, 该方法被称为椭圆包络.

8.3.1 参数的稳健估计: MCD方法

在椭圆包络方法中, 需要两个参数值, 即均值$\boldsymbol{\mu}$和协方差矩阵$\boldsymbol{\Sigma}$. 实际应用中, 它们往往未知, 需要作出估计. 通常以样本均值估计$\boldsymbol{\mu}$, 以样本的协方差矩阵估计$\boldsymbol{\Sigma}$. 但是, 样本均值和样本协方差矩阵都会受到数据中离群值的影响, 最直接的做法是先剔除离群值, 然后再作估计. 这就产生了矛盾: 我们为了识别离群值, 所以需要估计$\boldsymbol{\mu}$和$\boldsymbol{\Sigma}$, 但为了获得好的参数估计值需要先剔除离群值, 仿若一个"鸡生蛋, 蛋生鸡"的死循环.

如何解决这个矛盾呢? 突破点只能是在不作离群值识别的前提下, 获得不易受离群值影响的参数估计, 称为稳健估计(robust estimate). 下面, 我们着重讲解椭圆包络方法中获得参数稳健估计的方法, 即MCD (Minimum Covariance Determinant) 方法.

由于离群值的存在会增大数据的波动, 因此数据的波动越小, 存在离群值的可能性就越小. 由此, 可以通过数据的波动来反映离群值存在的可能性. 多元数据的波动可以通过样本协方差矩阵的行列式值予以反映, 样本协方差矩阵的行列式值越小, 表明数据的波动越小, 此时存在离群值的可能性就越小.

基于上述原理, MCD方法的主要思路是: 寻找使得样本协方差矩阵的行列式值最小的h个样本观测, 以它们作为不存在离群值的"纯净"总体代表, 从而获得参数的稳健估计. Rousseeuw and & Driessen (1999) 认为, h的取值应落入区间$[(n+p+1)/2,\ n]$, 其中n是样本量, p是变量的维度.

表 8.1: 各样本观测组合下协方差矩阵的行列式值

样本观测序号	(1,2,3,4)	(1,2,3,5)	(1,2,4,5)	(1,3,4,5)	(2,3,4,5)
协方差矩阵的行列式值	0.14	3.25	5.06	7.12	5.68

那么, 如何获得这h个样本观测呢？若样本量不太大, 可以穷举搜索所有由h个样本观测组合的数据集, 此时, 共有C_n^h个数据集. 通过比较各数据集中样本协方差矩阵的行列式值, 可以获得使行列式值达到最小的h个样本观测.

例 8.3 基于例*8.1*中的*5*个样本观测, 取$h = 4$, 获得$C_5^4 = 5$种样本观测组成的数据集, 各数据集样本协方差矩阵的行列式值如表8.1所示.

本例中, 非离群值数据的真实分布为二元正态分布$N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & -0.5 \\ -0.5 & 1 \end{pmatrix}\right)$, 其协方差矩阵的行列式值为*0.25*. 可以看到, 若数据集中含有序号为*5*的样本观测, 则协方差矩阵的行列式值都远大于*0.25*, 反映了离群值对协方差矩阵的估计有较大影响. 序号为*1*至*4*的样本观测所构成的数据集, 其协方差矩阵的行列值最小, 符合*MCD*的要求. 此时, 参数估计值为[6]

$$\hat{\boldsymbol{\mu}} = \begin{pmatrix} -1 \\ 1.38 \end{pmatrix}, \ \hat{\boldsymbol{\Sigma}} = \frac{1}{n}(\boldsymbol{X} - \bar{\boldsymbol{X}})^T(\boldsymbol{X} - \bar{\boldsymbol{X}}) \approx \begin{pmatrix} 0.75 & 0.24 \\ 0.24 & 0.26 \end{pmatrix}.$$

注记 8.4 由于*MCD*方法围绕单一中心刻画*"*纯净*"*总体, 因此*MCD*方法仅适用于两边对称的单峰分布, 对于多峰分布并不适用.

8.3.2 FAST-MCD算法

当样本量较大时, 通过穷举搜索寻找MCD方法的最优解效率不高. Rousseeuw & Driessen (1999) 提出以迭代的方式实现MCD方法, 以获得其近似解. 首先进行**初始化处理**:

(1) 从全体样本观测$\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n$中随机取$p + 1$个样本观测, 得到协方差矩阵的估计$\hat{\boldsymbol{\Sigma}}$. 若$|\hat{\boldsymbol{\Sigma}}| = 0$, 再随机抽取1个样本观测加入, 循环直到$|\hat{\boldsymbol{\Sigma}}| > 0$. 记由这些样本观测所获得的均值的估计为$\hat{\boldsymbol{\mu}}_{\text{old}}$, 协方差矩阵的估计为$\hat{\boldsymbol{\Sigma}}_{\text{old}}$.

(2) 计算所有样本观测$\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n$的马氏距离(Mahalanobis Distance, MD)的平方值:

$$\text{MD}^2(\boldsymbol{x}_i; \hat{\boldsymbol{\mu}}_{\text{old}}, \hat{\boldsymbol{\Sigma}}_{\text{old}}) = (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_{\text{old}})^T \hat{\boldsymbol{\Sigma}}_{\text{old}}^{-1} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_{\text{old}}), \ i = 1, \cdots, n.$$

[6]scikit-learn中covariance.EllipticEnvelope函数采用样本离差阵除以样本量n作为协方差矩阵的估计, 而未使用无偏估计(即除以$n - 1$). 这里按照EllipticEnvelope函数计算, 便于理解代码结果.

(3) 按照MD^2值从小到大的顺序对样本观测排序, 取排在前h位的样本观测组成集合$H = \{\boldsymbol{x}_{(1)}, \cdots, \boldsymbol{x}_{(h)}\}$. 采用$H$中全体数据, 获得参数估计的更新值$\hat{\boldsymbol{\mu}}$和$\hat{\boldsymbol{\Sigma}}$.

随后, 进行若干次迭代直至满足迭代停止规则. 其中的一次迭代(称为**C-step**) 步骤为:

(1) 基于$\hat{\boldsymbol{\mu}}$和$\hat{\boldsymbol{\Sigma}}$, 计算所有样本观测的$\mathrm{MD}^2$值, 即$\mathrm{MD}^2(\boldsymbol{x}_i; \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$ $(i = 1, \cdots, n)$.

(2) 按照MD^2值从小到大的顺序对所有样本观测排序, 取排在前h位的样本观测组成的集合作为H的更新, 并相应更新参数估计值$\hat{\boldsymbol{\mu}}$和$\hat{\boldsymbol{\Sigma}}$.

迭代收敛停止规则是: $|\hat{\boldsymbol{\Sigma}}| = 0$或两次迭代产生的$|\hat{\boldsymbol{\Sigma}}|$差异很小.

但是, 上述迭代方法所产生的参数估计值会受到初始入样数据点的影响, 即使最终迭代收敛, 也不能保证是(近似) 最优解. 可以设想, 若初始入样的样本观测中包含离群值, 那么这些离群值可能一直存在于各次迭代更新后的集合H中, 从而影响最终的参数估计值. 为解决这一问题, Rousseeuw & Driessen (1999) 提出FAST-MCD算法, 通过"广撒网"的方式涵盖更多的初始入样数据点, 并兼顾搜索和迭代效率, 以提高算法有效性. 当$h < n$且$p \geq 2$时, FAST-MCD算法根据样本量大小区分了两种情况.

第一种情况是当n较小时(如, $n \leq 600$) [7] , FAST-MCD算法步骤为:

(1) 初始化处理加2次C-step, 这一操作重复500次, 产生500组参数估计值$\hat{\boldsymbol{\mu}}$和$\hat{\boldsymbol{\Sigma}}$;

(2) 从(1) 中产生的500组参数估计值中挑选10个协方差矩阵行列式值最小者, 分别运行C-step直至收敛;

(3) 从(2) 中获得的10组参数估计值中挑选协方差矩阵行列式值最小者, 作为最终的参数估计值.

第二种情况是当n较大时(如, $n > 600$) , FAST-MCD算法步骤为:

(1) 通过随机划分或抽样, 构建若干互不重叠的样本子集, 一般每个子集样本量约为300, 最多为5个子集;

(2) 对每一样本子集: 作初始处理加2次C-step, 这一操作重复100次, 产生100组参数估计值$\hat{\boldsymbol{\mu}}$和$\hat{\boldsymbol{\Sigma}}$, 从中挑选10个协方差矩阵行列式值最小的参数估计值;

[7] Rousseeuw & Driessen (1999)中以600作为样本量量级的判断标准, 而scikit-learn的EllipticEnvelope函数是以500作为标准, 即当$n \leq 500$时认为n较小, 否则认为n较大.

(3) 汇总上述样本子集数据, 对(2) 中产生的所有参数估计值, 分别作2次C-step, 这一操作重复50次. 从中挑选10个协方差矩阵行列式值最小的参数估计值;

(4) 基于全体n个样本观测, 对(2) 中产生的10组参数估计值, 分别作若干次C-step (最好是直到收敛) . 从中挑选协方差矩阵行列式值最小者, 作为最终的参数估计值.

8.3.3 判别临界值γ的确定

获得稳健估计量$\hat{\boldsymbol{\mu}}$和$\hat{\boldsymbol{\Sigma}}$后, 可以构建椭圆包络方法的离群值区域:

$$\left\{\boldsymbol{x} : \mathrm{MD}^2(\boldsymbol{x}; \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}) = (\boldsymbol{x} - \hat{\boldsymbol{\mu}})^T \hat{\boldsymbol{\Sigma}}^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}) > \gamma\right\},$$

其中, γ决定了离群值区域的大小, 取值越大, 则离群值区域越小. 由于$\mathrm{MD}^2(\boldsymbol{x}; \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$的分布一般难以确定, 常用的方法是基于$\mathrm{MD}^2(\boldsymbol{x}; \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$的经验分布确定$\gamma$与离群值区域概率之间的关系, 从而选取$\gamma$值. 一种简化且直观的处理方式是取所有样本观测$\mathrm{MD}^2$值的$1-\alpha$分位数作为$\gamma$, 其中$\alpha$近似对应于离群值在全体样本数据中所占的比例, 它往往取决于我们主观的预判.

例8.3 (续) 基于MCD方法所得的参数估计值, 计算5个数据点的MD^2值, 分别为1.09, 2.50, 1.64, 2.76, 306.86. 可以看出, 第5个数据点$(10, 10)$对应的MD^2值远大于其他数据点. 若取离群值所占比例为0.1, 判别临界值为5个MD^2值的90%分位数185.22, 形成的椭圆边界如图8.3所示. 可以看到, 数据值$(10, 10)$落在了椭圆之外, 被判为离群值. 其余4个数据值落入椭圆之内, 未被判为离群值.

8.4 局部离群点因子

局部离群点因子(Local Outlier Factor, LOF) 是一种基于密度识别局部离群点的算法, 由Breunig et al. (2000) 提出. 其主要思想是: 通过数据点与其k ($k \geq 1$)个邻居的距离来定义局部密度, 将数据点的局部密度与其k个邻居的局部密度作比较, 获得LOF值; 若数据点的局部密度相对较小, 则LOF值较大, 该数据点被判为离群点.

8.4.1 距离、邻居与密度

记数据集为D, 数据点p与o的距离记为$d(p, o)$, 其中$d(\cdot, \cdot)$可取为常用的距离, 如欧氏距离(Euclidean Distance). 为计算LOF值, Breunig et al. (2000) 提出了特定的距离、邻居和密度等概念.

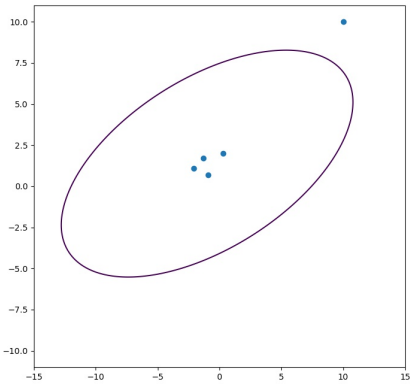


图 8.3: 5个数据点形成的椭圆包络

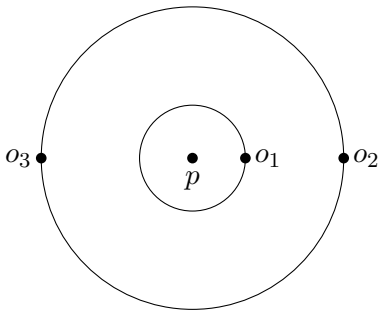


图 8.4: 数据点示意图

定义 8.2 对象p的k-距离*(k-distance of an object p):* 若数据点p与另一数据点$o \in D$之间的距离$d(p,o)$满足以下两个条件*:*

(i) 至少有k个数据点$o'(\in D\backslash\{p\})$与p的距离小于等于$d(p,o)$,

(ii) 最多有$k-1$个数据点$o'(\in D\backslash\{p\})$与p的距离小于$d(p,o)$,

则称距离$d(p,o)$为对象p的k-距离, 记为*k-distance(p).*

需要注意的是, 虽然k-distance(p)名称上是"距离", 但它仅涉及一个对象, 与通常的距离是不同的. k-distance(p)的计算方法是: 先计算数据点p与数据集D中其他所有数据点的距离, 然后按照距离从小到大排序, 排在第k位的距离值即为k-distance(p). 例如, 图8.4中, 以数据点p为对象, 1-distance(p)= $d(p,o_1)$, 2-distance(p)= $d(p,o_2)$ = $d(p,o_3)$.

定义 8.3 对象p的k-距离邻居*(k-distance neighborhood of an object p)*: 与数据点p的距离小于等于k-distance(p)的所有数据点, 称为对象p的k-距离邻居. 将对象p的k-距离邻居所构成的集合记为:

$$N_k(p) = \{q \in D \backslash \{p\} | d(p,q) \leq k - distance(p)\}.$$

$N_k(p)$包含所有与对象p为k-距离邻居关系的数据点, 它至少含有k个数据点. 当有多个数据点与p的距离都等于k-distance(p)时, $N_k(p)$中的数据点数量会超过k个. 例如, 以图8.4中数据点p为对象, $N_1(p) = \{o_1\}$, $N_2(p) = \{o_1, o_2, o_3\}$.

定义 8.4 对象p关于对象o的可达距离*(reachability distance of an object p w.r.t. object o)*:

$$reach\text{-}dist_k(p,o) = \max\{k - distance(o), d(p,o)\}.$$

例如, 图8.4中, 以数据点p为对象, reach-dist$_2(p, o_1) = d(o_1, o_2)$, reach-dist$_2(o_1, p) = d(p, o_2) > d(o_1, o_2) =$ reach-dist$_2(p, o_1)$. 这个例子表明, reach-dist$_k(p, o)$可能不等于reach-dist$_k(o, p)$, 从这个角度来看, 可达距离不具有对称性, 因而并非真正意义上的"距离".

定义 8.5 对象p的局部可达密度*(local reachability density of an object p)*:

$$lrd_k(p) = 1/\left[\frac{\sum_{o \in N_k(p)} reach\text{-}dist_k(p,o)}{|N_k(p)|}\right],$$

其中, $|N_k(p)|$表示集合$N_k(p)$的元素个数.

直观上看, 对象p的局部可达密度不仅取决于数据点p周围的密度, 还取决于其邻居周围的密集程度. 例如, 当$k = 2$时, 图8.4中数据点p的2-距离邻居集合$N_2(p)$包含3个数据点, 将相应的3个可达距离平均并取倒数即可获得数据点p的局部可达密度, 即

$$\text{lrd}_2(p) = 1/\left[\frac{\sum_{o \in N_2(p)} \text{reach-dist}_2(p,o)}{|N_2(p)|}\right] = \frac{3}{d(p,o_1) + d(p,o_2) + d(o_1,o_3)}.$$

同理可以计算数据点o_3的局部可达密度, 由于$N_2(o_3) = \{p, o_1\}$, 可得

$$\mathrm{lrd}_2(o_3) = 1/\left[\frac{\sum_{o\in N_2(o_3)} \text{reach-dist}_2(o_3, o)}{|N_2(o_3)|}\right] = \frac{2}{d(p, o_3) + d(o_1, o_3)}.$$

由于$d(p, o_1) < d(p, o_2) = d(p, o_3)$, 则有$1/\mathrm{lrd}_2(p) < 1/\mathrm{lrd}_2(o_3)$, 即$\mathrm{lrd}_2(p) > \mathrm{lrd}_2(o_3)$. 可以看到, 相对于位于数据聚集区域的数据点, 远离聚集区域的数据点的局部可达密度更小.

8.4.2 LOF值

基于上述邻居和密度等概念，可以给出LOF值的定义。

定义 8.6 对象p的*(局部)*离群因子*((local) outlier factor of an object p):*

$$LOF_k(p) = \frac{\sum_{o\in N_k(p)} \dfrac{lrd_k(o)}{lrd_k(p)}}{|N_k(p)|}.$$

由定义可知, LOF值是p的邻居的局部可达密度与p的局部可达密度比值的平均值. 对象p的LOF值较大, 表示相对于其邻居而言, 对象p的局部可达密度较小, 即对象p周围聚集点较少, 那么它离群的程度较大. 因而, LOF值刻画了数据点离群的程度, 其值越大, 表明数据点越有可能为离群点.

但是, 基于LOF值进行离群值判别的临界值并不明确, 难以确定当数据点的LOF值大到哪种程度才能被判为离群点. 在Breunig et al. (2000) 的一个实例中, 取离群值判别的LOF临界值为1.5 [8]. 若我们预先了解离群值所占的比例, 可以按照该比例取所有LOF值相应分位数值为临界值.

例 8.4 采用例*8.1*中的*5*个数据点, 取$d(\cdot,\cdot)$为欧氏距离, 得到各数据点间的距离如表*8.4*所示.

取邻居数$k = 2$, 以第*1*个数据点*(*记为o_1*)* 为例, 其最近的*2*个邻居*(*第*3*个数据点o_3和第*4*个数据点o_4*)* 与它的距离分别为*1.0*和*1.1*, 因此, 2-*distance*(o_1)=*1.1*. 同理可得2-*distance*(o_3) = 2-*distance*(o_4)=*1.3*. 由此得到o_1关于其邻居o_3的局部可达距离*:*

$$\textit{reach-dist}_2(o_1, o_3) = \max\{k - \textit{distance}(o_3), d(o_1, o_3)\} = \max\{1.3, 1.0\} = 1.3.$$

[8] scikit-learn中默认以1.5作为临界值. 若设定了离群值的比例, 则按照该比例取所有LOF值的分位数为临界值

表 8.2: 5个数据点间的欧氏距离

序号	1	2	3	4	5
1	0	1.6	1.0	1.1	14.0
2	1.6	0	2.6	1.8	12.6
3	1.0	2.6	0	1.3	15.0
4	1.1	1.8	1.3	0	14.3
5	14.0	12.6	15.0	14.3	0

表 8.3: 各数据点LOF值及其计算相关值

序号	2-distance	N_2	reach-dist$_2$	lrd$_2$	LOF$_2$
1	1.1	$\{3,4\}$	1.3, 1.3	1/1.3	1.1
2	1.8	$\{1,4\}$	1.6, 1.8	1/1.7	1.4
3	1.3	$\{1,4\}$	1.1, 1.3	1/1.2	1.0
4	1.3	$\{1,3\}$	1.1, 1.3	1/1.2	1.0
5	14.0	$\{2,1\}$	12.6, 14.0	1/13.3	9.0

类似地, 可以得到o_1关于其第*2*个邻居o_4的局部可达距离$reach\text{-}dist_2(o_1,o_4)$为*1.3*. 从而,

$$lrd_2(o_1) = 1/\left[\frac{\sum_{o\in N_2(o_1)} reach\text{-}dist_2(o_1,o)}{|N_2(o_1)|}\right] = \frac{2}{1.3+1.3} = \frac{1}{1.3}.$$

类似可以获得$lrd_2(o_3)$和$lrd_2(o_4)$, 均为1/1.2. 因此,

$$LOF_2(o_1) = \frac{\sum_{o\in N_2(o_1)} \frac{lrd_2(o)}{lrd_2(o_1)}}{|N_2(o_1)|} = \left(\frac{1.3}{1.2}+\frac{1.3}{1.2}\right)/2 \approx 1.1.$$

类似地, 可以计算得到其他数据点的LOF值, 列于表*8.3*中. 可以看到, *(10,10)*对应的LOF值最大, 大于*1.5*, 可被识别为离群值.

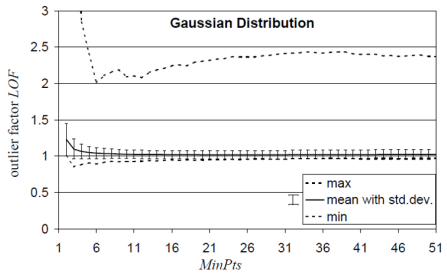


图 8.5: 局部离群点因子方法中邻居数量k的影响

8.4.3 邻居数量k的设置

Breunig et al. (2000) 指出, 邻居数量k [9]是一个较为关键的参数值. 基于随机模拟的数据, Breunig et al. (2000) 呈现了k与LOF_k的关系, 如图8.5所示, 图中横坐标是邻居数量k, 纵坐标是LOF_k值. 图中有三条线, 实线表示在各邻居数量k下所有数据点LOF_k的平均值, 由平均值向上下两边延申的区间长度是所有数据点LOF_k的标准差; 上下两条虚线分别表示所有数据点LOF_k的最大值和最小值. 可以看到, 邻居数量k取值不同, LOF值的变化较大, 因此k对于离群值的识别有较大的影响. Breunig et al. (2000) 建议, k最好大于10, 这样LOF值波动较小, 较为稳定; 同时, k应小于数据簇中的样本量, 即数据聚集区域中数据点个数, 以保证所找到的k个邻居确实距离相近.

8.5 孤立森林

孤立森林(Isolation Forest) [10]由Liu, Ting & Zhou (2008) 提出, 其主要思想是: 由于离群值应是“少而不同”, 即分布稀疏且离密度高的数据点群体较远, 因此在随机构建的“二叉树”中, 离群值离根节点较近, 即路径长度较小, 从而可根据路径长度值识别离群点.

一些实证结果显示, 相对于局部离群点因子等方法, 孤立森林方法识别离群值的效果更好且计算效率更高.

[9]上文提到, 对象p的k-距离邻居$N_k(p)$至少含有k个元素, 即对象p的邻居数量可能超过k, 所以k不一定是确切的“邻居数量”. 在scikit-learn的LocalOutlierFactor函数中, 邻居数量等于k (不会大于k), 原因在于在LocalOutlierFactor函数中仅取$N_k(p)$中的k个元素.

[10]根据周志华(2019), 将isolation翻译为“孤离”更贴合方法的原理, 但此处仍然遵从对该方法习惯性的称呼“孤立森林”.

8.5.1 孤立树

孤立森林由多棵随机构建的“二叉树”构成, 这些树被称为孤立树(Isolation Tree, 简记为iTree). 与CART等决策树类似, iTree形如一棵倒挂的树, 顶端为根节点, 其中包含了所有的样本观测; 自根节点开始, 需要通过一定的检测条件决定下行的分枝, 直至末端的叶节点(不产生分枝的节点). 除根节点和叶节点外的节点, 都属于中间节点. 记T为iTree中的一个节点, T要么为叶节点, 要么为根据1个检测条件分为2个分枝(T_l, T_r)的根节点或中间节点. 检测条件由随机选取的单一自变量X和随机选取的临界值x_0决定, 根据$X \le x_0$是否成立, 将节点T中的样本观测分入左枝T_l或右枝T_r.

当达到一定规模后, iTree应停止生长. 在Liu, Ting & Zhou (2008) 中, 只要满足下列条件之一, iTree即会停止生长:

- (1) 树已达到要求的最大深度(或高度);
- (2) 所有叶节点都只含有1个观测;
- (3) 叶节点中所有样本观测的取值相同.

注记 8.5 *iTree*属于无监督算法*(unsupervised algorithm)*, 即分枝并不依据某一优化指标, 而是通过随机选择分枝变量并随机选择分枝临界值来构造. 由此可见, *iTree*与*CART*等监督算法*(supervised algorithm)* 具有很大差异.

之所以可以根据iTree识别离群值, 是因为离群值与其他数值间隔更大, 从而间隔中包含了更多的备选分枝临界值, 因而离群值更有可能被分割进入一个只包含少数样本观测的节点中, 从而更快地落入叶节点. 下面的例子是一个直观的说明.

例 8.5 采用例*8.1*中的*5*个数据点, 设置*iTree*的最大深度为*2*. 建立*iTree*时, 需要在每一节点随机选取一个变量作为分枝变量, 然后在该变量的取值域中随机选取一个数值作为分枝临界值. 为便于计算, 我们可以先将每一维度上的取值排序.

按照第*1*个维度X_1的取值从小到大排序*: -2.1, -1.3, -0.9, 0.3, 10;*

按照第*2*个维度X_2的取值从小到大排序*: 0.7, 1.1, 1.7, 2.0, 10.*

当X_1被选为分枝变量时*(*概率为*0.5)* , 需在$(-2.1, 10)$中选择一个分枝临界值*;* 当X_2被选为分枝变量时*(*概率为*0.5)* , 需在$(0.7, 10)$中选择一个分枝临界值.

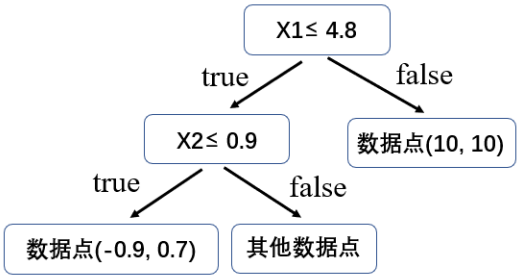


图 8.6: 一棵可能的iTree

对于前*2*个数据点*(-1.3, 1.7)*和*(0.3, 2.0)*, 无论哪一个变量作为分枝变量、无论分枝临界值落入哪个位置, 它们都不可能在第一次分枝中被单独分到一个叶节点, 因此需要*2*次分割才能进入叶节点.

第*3*个数据点*(-2.1, 1.1)*在X_1被选为分枝变量且分枝临界值位于$(-2.1, -1.3)$时, 可在第*1*次分割中就进入叶节点, 发生的概率为*:*

$$0.5 \times \frac{-1.3 - (-2.1)}{10 - (-2.1)} \approx 0.0331.$$

第*4*个数据点*(-0.9, 0.7)*在X_2被选为分枝变量且分枝临界值位于$(0.7, 1.1)$时, 可在第*1*次分割中就进入叶节点, 发生的概率为*:*

$$0.5 \times \frac{1.1 - 0.7}{10 - 0.7} \approx 0.0215.$$

第*5*个数据点*(10,10)*在X_1被选为分枝变量且分枝临界值位于$(0.3, 10)$或X_2被选为分枝变量且分枝临界值位于$(2.0, 10)$时, 可在第*1*次分割中就进入叶节点, 发生的概率为*:*

$$0.5 \times \frac{10 - 0.3}{10 - (-2.1)} + 0.5 \times \frac{10 - 2.0}{10 - 0.7} \approx 0.8309.$$

可以看到, 数据点*(10,10)*在两个维度上的值都离其他值更远, 分枝临界值落入以*10*为上限的区间中可能性更大, 从而*(10,10)*有可能更快地进入叶节点. 图*8.6*中展示了一棵可能的*iTree*, 经过*1*次分割, *(10,10)*就进入了叶节点.

8.5.2 路径长度

获得iTree后, 需要借助一定的量化指标对数据值的离群程度作刻画并排序. 下面介绍第一种量化指标——路径长度.

定义 8.7 路径长度*(path length)*是从$iTree$的根节点到达数据点$\boldsymbol{x}$所在的叶节点所经过的边的数量, 记为$h(\boldsymbol{x})$.

例如, 图8.6中, 数据点$(10, 10)$的路径长度为1, 其余数据点的路径长度都为2. 由于当节点中只含有1个样本观测时不能再继续分枝, 所以$h(\boldsymbol{x})$的取值是有限的. 设样本量为n, 容易知道, $0 < h(\boldsymbol{x}) \leq n - 1$. 直观上看, $h(\boldsymbol{x})$越小, 则$\boldsymbol{x}$为离群值的可能性越大. 但是, 由于iTree是完全随机构建的, 因此$h(\boldsymbol{x})$具有随机性, 难以直接用于离群值的识别. 如何解决这个问题呢？统计学中消除随机性最常用的办法就是对随机变量取期望值. 这里, 我们可以使用期望值$E(h(\boldsymbol{x}))$作为判别离群值的指标. $E(h(\boldsymbol{x}))$越小, 数据值离群的可能性就越大.

例 8.5 (续) 基于$h(x)$的分布, 可以计算各数据点的平均路径长度$E(h(\boldsymbol{x}))$. 对于前2个数据点(-1.3, 1.7)和(0.3, 2.0), 无论哪一个变量作为分枝变量、无论分枝临界值落入哪个位置, 它们都不可能在第一次分枝中被单独分到一个叶节点, 因而路径长度只可能为2, 从而期望路径长度也为2.

第3个数据点(-2.1, 1.1)在X_1被选为分枝变量且分枝临界值位于$(-2.1, -1.3)$时路径长度为1, 其他情况的路径长度为2, 从而期望路径长度为:

$$1 \times 0.5 \times \frac{-1.3 - (-2.1)}{10 - (-2.1)} + 2 \times \left(1 - 0.5 \times \frac{-1.3 - (-2.1)}{10 - (-2.1)}\right) \approx 1.9669.$$

第4个数据点(-0.9, 0.7)在X_2被选为分枝变量且分枝临界值位于$(0.7, 1.1)$时路径长度为1, 其他情况的路径长度为2, 从而期望路径长度为:

$$1 \times 0.5 \times \frac{1.1 - 0.7}{10 - 0.7} + 2 \times \left(1 - 0.5 \times \frac{1.1 - 0.7}{10 - 0.7}\right) \approx 1.9785.$$

第5个数据点(10,10)在X_1被选为分枝变量且分枝临界值位于$(0.3, 10)$或X_2被选为分枝变量且分枝临界值位于$(2.0, 10)$时, 路径长度为1, 发生的概率为0.8309. 从而, 数据点(10,10)会以概率$1 - 0.8309 = 0.1691$取到路径长度2. 因此, 数据点(10,10)的期望路径长度为:

$$1 \times 0.8309 + 2 \times 0.1691 = 1.1691.$$

由这个例子可以看出, 期望路径长度$E(h(\boldsymbol{x}))$的计算是比较复杂的, 尤其是当数据量比较大时, 期望路径长度的计算难以实现. 那么, 如何快速获得期望路径长度呢？一个直接的想法是以样本均值估计期望值. 因此, Liu, Ting & Zhou (2008) 产生多棵iTrees构成森林, 以这些iTrees的$h(\boldsymbol{x})$的均值作为$E(h(\boldsymbol{x}))$的估计值[11]. 这就是孤立森林名称的由来.

[11]请注意, 由此获得的$E(h(\boldsymbol{x}))$估计值具有随机性, 其值与iTrees的数量, 以及构建每一棵iTree时随机抽取的分枝变量和分枝临界值有关.

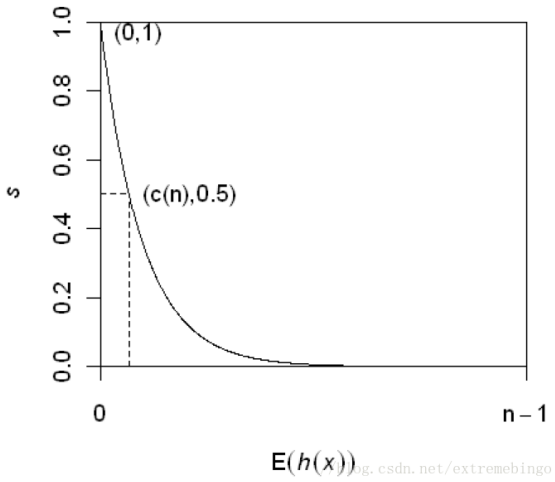


图 8.7: 异常分与$E(h(\boldsymbol{x}))$的关系

8.5.3 异常分

注意到, 路径长度$h(\boldsymbol{x})$的取值范围与样本量n有关, 因此$E(h(\boldsymbol{x}))$受到样本量n的影响, 使得我们难以找到普适性的判断临界值. Liu, Ting & Zhou (2008) 采用给定n时$E(h(\boldsymbol{x}))$的均值(记为$c(n)$) 予以调整, 从而得到标准化的度量指标——异常分(anomaly score):

$$s(\boldsymbol{x}, n) = 2^{-\frac{E(h(\boldsymbol{x}))}{c(n)}}.$$

根据二叉查找树(binary search tree) 中的理论, $c(n)$为含有n个样本观测的不成功搜索的平均路径长度, 其表达式为

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n},$$

其中, $H(i)$为调和数, 其估计值为$\ln(i) + 0.5772156649$.

图8.7展示了异常分与$E(h(\boldsymbol{x}))$的关系. 可以看出, 异常分与$E(h(\boldsymbol{x}))$成反比, 当$E(h(\boldsymbol{x})) \to 0$时, $s \to 1$; 当$E(h(\boldsymbol{x})) \to c(n)$时, $s \to 0.5$; 当$E(h(\boldsymbol{x})) \to n-1$时, $s \to 0$. 异常分越大的数值, 为离群值的可能性就越大. 因此, 孤立森林判断离群值的准则为:

- 若数据值的异常分s接近于1, 则该值被判为离群值;
- 若数据值的异常分s小于0.5, 则该值不是离群值;
- 若所有数据值的异常分s约等于0.5, 则所有值都不是离群值.

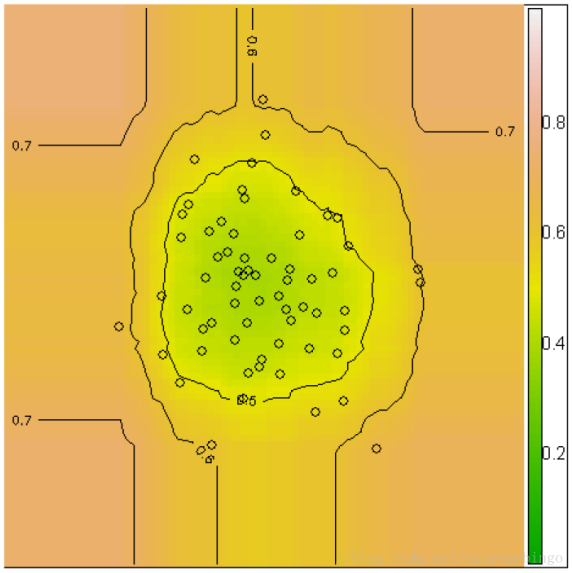


图 8.8: 等高线图

8.5.4 判别临界值γ的确定

可以看到, 上述判断离群值的准则比较粗略. 实际使用中, 需要设置临界值$\gamma(>0)$, 当异常分$s \geq \gamma$时, 将数据值判为离群值. 临界值γ是孤立森林中的一个重要参数, 但尚缺乏普适性的规则确定γ. 基于64个高斯分布随机数的模拟实验, Liu, Ting & Zhou (2008) 制作了等高线图(如图8.8所示). 图中最中心的圆圈对应γ取值为0.5, 向外一层的圆圈和线条对应γ取值为0.6, 三个角落的线条对应γ取值为0.7. 可以看到, 当γ取值为0.6时, 落在等高线以外的样本观测约为3个, 数量较为合理. 因此, Liu, Ting & Zhou (2008) 选用0.6作为判别临界值γ. 实践中, 我们可以先设置离群点数量比例, 然后确定满足该比例的判别临界值γ [12].

例 8.5 (续)由于

$$c(5) = 2\times H(5-1) - \frac{2\times(5-1)}{5} = 2\times(\ln(4)+0.5772156649) - \frac{2\times(5-1)}{5} \approx 2.3270,$$

从而5个数据点的异常分分别为0.5512, 0.5512, 0.5566, 0.5547, 0.7059. 可以看到, 数据点(10, 10)的期望路径长度相对最小、异常分最高. 若取判别临界值γ为0.6, 那么只有(10,10)的异常分高于0.6, 此时(10,10)被判为离群值.

[12] 由scikit-learn中的IsolationForest函数实现孤立森林, γ的默认值为0.5, 可以通过设置离群点数量比例确定γ.

注记 8.6 *Liu, Ting & Zhou (2008)* 发现, 当数据量较小时更容易识别离群值, 因而提出基于子采样*(sub-sampling)* 的数据集构建每一棵*iTree,* 以提高识别准确度和计算效率.

8.6 单类支持向量机

单类支持向量机(One-Class Support Vector Machine) 由Schölkopf et al. (2001) 提出, 可用于识别离群值, 也可用于发现一些新兴个体. 该方法的主要思想是: 以一个超平面分割出两个区域, 其中一个区域"小"且包含尽量多的样本观测, 而落入另一个区域中的少量数据值即为离群值.

8.6.1 基于线性超平面的离群值识别

单类支持向量机基于线性超平面$\{\boldsymbol{x} : \boldsymbol{w}^T\boldsymbol{x} - \rho = 0\}$, 采用的决策函数(decision function) 为$f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^T\boldsymbol{x} - \rho)$, 其中sign$(\cdot)$为取值为$\pm 1$的符号函数. 当$\boldsymbol{w}^T\boldsymbol{x} - \rho < 0$ (即$f(\boldsymbol{x}) = -1$)时, 数据值$\boldsymbol{x}$被判为离群值. 可以看到, ρ越大, 则离群值的区域越大, 可能识别出更多的离群值.

超平面中包含的参数有$\boldsymbol{w}$和ρ, 它们是以下优化问题的解:

$$\begin{aligned} \min_{\boldsymbol{w}\in\mathbb{R}^p, \boldsymbol{\xi}\in\mathbb{R}^n, \rho\in\mathbb{R}} \quad & \frac{\|\boldsymbol{w}\|^2}{2} + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho, \\ \text{s.t.} \quad & \boldsymbol{w}^T\boldsymbol{x}_i \geq \rho - \xi_i, \quad \xi_i \geq 0 \quad (i = 1, \cdots, n). \end{aligned} \tag{8.1}$$

注意, 在上述优化问题中, ν为设定值, 而$\boldsymbol{w}, \rho, \boldsymbol{\xi}$都是需要优化的参数. 理解这一个目标函数以及各个参数至关重要, 下面作具体解释.

由于$\boldsymbol{w}$是超平面中$\boldsymbol{x}$对应的系数, 所以目标函数中的第1项, 即$\|\boldsymbol{w}\|^2/2$, 刻画的是模型的复杂度. 我们希望这一项值要小, 也就是希望模型的复杂度尽量小, 使模型尽量简单.

在目标函数的第2项中, $\xi_i\,(i = 1, \cdots, n)$是支持向量机的松弛变量(slack variable), 显示了数据点与超平面之间的关系. 若数据点$\boldsymbol{x}_i$满足$\boldsymbol{w}^T\boldsymbol{x}_i \geq \rho$, 即落入超平面中或位于超平面上方, 则数据点天然地满足了第一个约束条件, 因而不需要额外地放松条件, 可取$\xi_i = 0$. 若数据点$\boldsymbol{x}_i$满足$\boldsymbol{w}^T\boldsymbol{x}_i < \rho$, 即落入超平面下方, 也即落入离群值区域之中, 则必须要求$\xi_i > 0$才能满足第一个约束条件, 这就是所谓的"松弛"的含义. 并且, 超平面下方的数据点离超平面越远, 所需要的松弛强度就越大, ξ_i的值就会越大. 因此, 松弛变量从一定程度上刻画了偏离正常数据区域的程度, 这种偏离可以看作是一种损失. 从而$\sum_{i=1}^{n}\xi_i$为所有损失之和. 可以看到, 该损失之和受到离群值数量的影响, 为消除这一影响, 我们以每一个离群值所带来的平均损失予以刻画. 后面的分析将会显示, 设定值ν表示离群值所占比例的上界, 那么νn就是离群值数量

的上界, 在一定程度上代表离群值数量. 所以用损失之和除以νn即可得到平均损失. 我们希望平均损失要小.

前面提到, 单类支持向量机的离群值识别是通过比较$\boldsymbol{w}^T\boldsymbol{x}$与$\rho$而得的. 因此, 可以将$\rho$看作一个判别临界值, ρ的取值控制着正常数据区域的范围. 因为希望正常数据区域尽量小, 因而ρ越大越好. 取$-\rho$, 则是希望这一项尽量小, 以与其他项的最小化保持一致.

注记 8.7 在优化目标中引入超平面的截距项ρ是单类支持向量机的一个特点, 它使得超平面可作平移调节以满足特定要求[13] *(*后面将看到, 设定ν的值实质是对支持向量和离群值的数量提出要求*).*

汇总上述3项, 最终希望目标函数尽量小. 至此, 我们已经将离群值识别问题转化一个数学上的优化问题, 只要求解出优化问题的解, 就可以识别离群值了.

8.6.2 参数$\boldsymbol{w}$的求解

单类支持向量机的优化问题(8.1) 是一个带约束的非凸优化问题, 其求解是比较困难的. 与通常的支持向量机方法类似, 求解(8.1) 较为高效的方法是求解其对偶问题(dual problem)[14]. 下面, 简要介绍这一求解过程.

通过引入拉格朗日乘子$\alpha_i \geq 0,\ \beta_i \geq 0\ (i = 1, \cdots, n)$, 可得到拉格朗日函数:

$$\begin{aligned} L(\boldsymbol{w}, \boldsymbol{\xi}, \rho, \boldsymbol{\alpha}, \boldsymbol{\beta}) \quad &= \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho \\ &\quad - \sum_{i=1}^{n}\alpha_i(\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i) - \sum_{i=1}^{n}\beta_i\xi_i. \end{aligned}$$

这样, 原始带约束的优化问题等价于:

$$\begin{aligned} \min_{\boldsymbol{w}\in\mathbb{R}^p, \boldsymbol{\xi}\in\mathbb{R}^n, \rho\in\mathbb{R}} \max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & L(\boldsymbol{w}, \boldsymbol{\xi}, \rho, \boldsymbol{\alpha}, \boldsymbol{\beta}), \\ \text{s.t.} \quad & \alpha_i \geq 0,\ \beta_i \geq 0 \quad (i = 1, \cdots, n). \end{aligned} \tag{8.2}$$

理解这一等价形式的关键是, 其中拉格朗日乘子$\boldsymbol{\alpha}, \boldsymbol{\beta}$对应的目标是取max, 一旦原优化问题中的约束条件不满足, 拉格朗日乘子可以取到正无穷, 从而目标函数不可能达到最小值. 这种处理强制优化的解一定满足原优化问题中的约束条件.

[13]监督学习中的支持向量机方法ν-SVC与ν-SVR也采用了类似机制, 通过调节超平面边界以控制支持向量和错误预测的数量.

[14]可查看周志华著《机器学习》附录中关于优化的内容.

交换(8.2)式中min和max, 就得到原问题的对偶问题:

$$\begin{aligned} \max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \min_{\boldsymbol{w}\in\mathbb{R}^p,\boldsymbol{\xi}\in\mathbb{R}^n,\rho\in\mathbb{R}} \quad & L(\boldsymbol{w},\boldsymbol{\xi},\rho,\boldsymbol{\alpha},\boldsymbol{\beta}), \\ \text{s.t.} \quad & \alpha_i \geq 0,\ \beta_i \geq 0 \ \ (i=1,\cdots,n). \end{aligned} \tag{8.3}$$

之所以认为对偶问题更容易求解, 是因为对偶问题中的min部分是关于$(\boldsymbol{w},\boldsymbol{\xi},\rho)$的无约束优化, 可通过令偏导为0, 表达出$(\boldsymbol{w},\boldsymbol{\xi},\rho)$, 即:

$$\frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{n}\alpha_i\boldsymbol{x}_i = 0 \quad \Rightarrow \quad \boldsymbol{w} = \sum_{i=1}^{n}\alpha_i\boldsymbol{x}_i, \tag{8.4}$$

$$\frac{\partial L}{\partial \xi_i} = \frac{1}{\nu n} - \alpha_i - \beta_i = 0 \quad \Rightarrow \quad \alpha_i + \beta_i = \frac{1}{\nu n},\ (i=1,\cdots,n) \tag{8.5}$$

$$\frac{\partial L}{\partial \rho} = -1 + \sum_{i=1}^{n}\alpha_i = 0 \quad \Rightarrow \quad \sum_{i=1}^{n}\alpha_i = 1. \tag{8.6}$$

(8.4)式清晰地显示了超平面中系数$\boldsymbol{w}$的解, 即为数据取值的加权平均, 其中的权重由拉格朗日乘子$\boldsymbol{\alpha}$决定. 后面将看到, 一部分的α_i取值等于0, 因此$\boldsymbol{w}$仅由$\alpha_i > 0$对应的数据点所决定, 这显示出单类支持向量机的解具有稀疏性. (8.6)式恰好对应$\boldsymbol{w}$的求解式中所有权重之和为1.

将上述表达式代入8.3式, 对偶问题变为:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\sum_{1\leq i,j\leq n}\alpha_i\alpha_j\boldsymbol{x}_i^T\boldsymbol{x}_j \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq \frac{1}{\nu n}\ (i=1,\cdots,n),\ \sum_{i=1}^{n}\alpha_i = 1, \end{aligned} \tag{8.7}$$

其中的约束$\alpha_i \leq \frac{1}{\nu n}$来源于(8.5)式, 即$\alpha_i = \frac{1}{\nu n} - \beta_i \leq \frac{1}{\nu n}$; 约束$\sum_{i=1}^{n}\alpha_i = 1$来源于(8.6)式.

这又成为一个带约束的非凸优化问题. 可以通过随机梯度下降(Stochastic Gradient Descent, SGD) 等方法求解. 也有一些高效的求解方法, 感兴趣的读者可参考Schölkopf et al. (2001), 这里不再展开.

现在, 我们假定已经得到了对偶问题(8.7)式的解. 此时, 还需要考虑: 对偶问题的解等于原问题的解吗? 答案是, 只有在某些情况下二者具有相同解. 研究发现, 当满足KKT (Karush-Kuhn-Tucker) 条件时, 对偶问题与原问题具有相同解. 这里, KKT条件主要包含3个部分:

(1) 原问题中的约束成立, 即$\boldsymbol{w}^T\boldsymbol{x}_i \geq \rho - \xi_i,\ \ \xi_i \geq 0\ (i=1,\cdots,n)$;

(2) 对偶问题中的约束成立, 即$0 \leq \alpha_i \leq \frac{1}{\nu n},\ \sum_{i=1}^{n}\alpha_i = 1,\ \beta_i \geq 0,\ \alpha_i + \beta_i = \frac{1}{\nu n}\ (i=1,\cdots,n)$;

(3) 拉格朗日函数中的每一乘积项应等于0, 即$\alpha_i(\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i) = 0$, $\beta_i\xi_i = 0$, $(i = 1, \cdots, n)$, 且两个相乘项不同时为0.

上述KKT条件中的第3项要求对任意数据点$\boldsymbol{x}_i(i = 1, \cdots, n)$, 满足$\alpha_i(\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i) = 0$, 且$\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i$与$\alpha_i$不能同时为0. 根据KKT中的第1项, $\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i \geq 0$. 若$\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i > 0$, 则最优解在约束区域内部, 约束天然成立, 不需要拉格朗日乘子作用, 可设置拉格朗日乘子α_i为0; 若$\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i = 0$, 则最优解在约束区域边界, 需要拉格朗日乘子满足使得拉格朗日函数在最优解处的偏导数为0的条件, 此时拉格朗日乘子α_i不能为0. 因此, $\alpha_i(\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i)$必然为0, 但两个相乘项不能同时为0. 类似地, 应要求对任意数据点$\boldsymbol{x}_i(i = 1, \cdots, n)$, 满足$\beta_i\xi_i = 0$, 且$\xi_i$与$\beta_i$不能同时为0.

一旦由对偶问题获得了α_i $(i = 1, \cdots, n)$, 便可根据(8.4)式获得系数$\boldsymbol{w}$的估计值:

$$\hat{\boldsymbol{w}} = \sum_{i=1}^{n} \alpha_i \boldsymbol{x}_i. \tag{8.8}$$

8.6.3 支持向量与离群值

由(8.4) 和(8.9) 式可以看出, 单类支持向量机模型的参数估计值仅依赖于$\alpha > 0$的样本观测, 而与$\alpha = 0$的样本观测无关. 所以, 我们说$\alpha > 0$的样本观测支撑着超平面, 称它们为支持向量(support vector). 这里, 落在超平面中和超平面下方的样本观测对应的α_i都大于0, 它们构成了支持向量.

由KKT条件, 我们将数据点分为3类:

(1) 第1类数据点满足$\alpha_i = 0$. 根据KKT条件(3), α_i与$\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i$不能同时为0, 则此时$\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i > 0$. 进一步, 由KKT条件(2) 可得$\beta_i = 1/(\nu n)$; 由于$\beta_i > 0$, 则由KKT条件(3) 知, 必定有$\xi_i = 0$. 因此, $\boldsymbol{w}^T\boldsymbol{x}_i - \rho > 0$, 这表明$\boldsymbol{x}_i$落在超平面上方. 此时, $\boldsymbol{x}_i$既非支持向量, 亦非离群值.

(2) 第2类数据点满足$0 < \alpha_i < 1/(\nu n)$. 根据KKT条件(3) 可知$\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i = 0$. 与上面的推理类似, 可知$\beta_i > 0$, $\xi_i = 0$. 因此, $\boldsymbol{w}^T\boldsymbol{x}_i - \rho = 0$, 这表明$\boldsymbol{x}_i$正好落在超平面中. 此时, $\boldsymbol{x}_i$为支持向量, 但并非离群值.

(3) 第3类数据点满足$\alpha_i = 1/(\nu n)$. 根据KKT条件(3) 可知$\boldsymbol{w}^T\boldsymbol{x}_i - \rho + \xi_i = 0$. 根据KKT条件(2)可推知$\beta_i = 0$, 进一步由KKT条件(3) 得到$\xi_i > 0$. 此时, $\boldsymbol{w}^T\boldsymbol{x}_i - \rho = -\xi_i < 0$, 这表明$\boldsymbol{x}_i$落在超平面下方. 此时, $\boldsymbol{x}_i$为支持向量, 且被判为离群值.

由此, 可以得到以下结论:

(1) 支持向量位于超平面中或超平面下方; 支持向量包含了离群值.

(2) 离群值位于超平面下方, 对应的$\alpha = 1/(\nu n)$; 所有$\alpha = 1/(\nu n)$对应的样本观测值都属于离群值. 这表明, 可以根据α判别离群值.

至此, 我们可以解释单类支持向量机中重要参数ν的含义. 由于所有的α都小于等于$\frac{1}{\nu n}$, 为满足$\sum_{i=1}^{n} \alpha_i = 1$, 则支持向量至少为$\nu n$个. 另一方面, 离群值对应的$\alpha = \frac{1}{\nu n}$, 由于$\sum_{i=1}^{n} \alpha_i = 1$, 则离群值至多为$\nu n$个. 因而, ν是支持向量所占比例的下界、离群值所占比例的上界.

注记 8.8 ν取值范围为$(0, 1)$, 它在很大程度上决定了支持向量和离群值的数量, 是需要考虑的一个关键设置参数.

8.6.4 参数ρ的求解

下面, 讨论如何获得超平面中的参数ρ. 求解ρ的思路是从超平面中找出1个样本观测(记为$\boldsymbol{x}_m$), 建立关于ρ的方程, 从而求解ρ. 具体而言, 由于超平面中的样本观测$\boldsymbol{x}_m$满足$\boldsymbol{w}^T\boldsymbol{x}_m - \rho = 0$, 因此可以得到$\rho$的解:

$$\rho = \boldsymbol{w}^T\boldsymbol{x}_m = \sum_{i=1}^{n} \alpha_i \boldsymbol{x}_i^T \boldsymbol{x}_m. \tag{8.9}$$

代入$\boldsymbol{w}$的估计值$\hat{\boldsymbol{w}}$ ((8.8)式), 得到ρ的估计值为:

$$\hat{\rho}_m = \hat{\boldsymbol{w}}^T\boldsymbol{x}_m = \sum_{i=1}^{n} \alpha_i \boldsymbol{x}_i^T \boldsymbol{x}_m. \tag{8.10}$$

如何找出超平面中的样本观测$\boldsymbol{x}_m$呢? 根据前面的分析, 可以根据α的值判断样本观测与超平面的相对位置: $\alpha = 0$对应的样本观测落在超平面上方; $0 < \alpha < 1/(\nu n)$对应的样本观测正好落入超平面中; $\alpha = 1/(\nu n)$对应的样本观测落在超平面下方. 因此, 求解对偶问题(8.7)式获得$\boldsymbol{\alpha}$的解之后, 我们只需要找一个满足$0 < \alpha < 1/(\nu n)$的样本观测, 即可作为$\boldsymbol{x}_m$.

若满足$0 < \alpha < 1/(\nu n)$的样本观测有多个, 那么可以获得多个ρ的估计值, 这些估计值可能并不相同. 此时, 如何估计ρ呢? 一种方法是取所有ρ估计值的最小值[15], 即

$$\hat{\rho} = \min_{m \in \{i:\ 0<\alpha_i<1/(\nu n)\}} \{\hat{\rho}_m.\} \tag{8.11}$$

这可以保证α与离群值的对应关系, 即只有$\alpha = 1/(\nu n)$对应的样本观测被识别为离群值. 但是, 这个方法可能导致一部分$0 < \alpha < 1/(\nu n)$对应的样本观测落入超平面上方, 而不再位于超平面之中.

[15]监督算法中的支持向量机通常取所有ρ的估计值的平均值, 以获得稳健的估计量. 但在单类支持向量机中, 这样的估计量会导致集合H中的部分样本观测被识别为离群值.

记$\hat{\rho}$对应的样本观测为$\boldsymbol{x}_k$, 则所得的决策函数为:

$$\hat{f}(\boldsymbol{x}) = \text{sign}\left(\hat{\boldsymbol{w}}^T\boldsymbol{x} - \hat{\rho}\right) = \text{sign}\left(\sum_{i=1}^{n}\alpha_i\boldsymbol{x}_i^T\boldsymbol{x} - \sum_{i=1}^{n}\alpha_i\boldsymbol{x}_i^T\boldsymbol{x}_k\right). \tag{8.12}$$

8.6.5 基于非线性超平面的离群值识别

上述超平面$\{\boldsymbol{x} : \boldsymbol{w}^T\boldsymbol{x} - \rho = 0\}$是线性的, 即由$\boldsymbol{x}$各维度的线性组合生成. 采用线性超平面, 意味着我们假定可以由线性超平面分割正常数据区域与离群值区域. 这是一种非常强的假定, 而现实数据不一定符合该假定. 一种直观的解决办法是先对原始数据作变换, 使得变换后的数据$\phi(\boldsymbol{x})$可以在线性超平面的框架下处理.

但是, $\phi(\cdot)$的选择范围很广, 为实际操作带来了困难. 注意到决策函数(8.12)式完全由数据的内积($\boldsymbol{x}_i^T\boldsymbol{x}$和$\boldsymbol{x}_i^T\boldsymbol{x}_k$) 决定, 可以采用核方法(kernel trick), 即由核函数代替超平面中的内积, 相当于先将$\boldsymbol{x}$变换为$\phi(\boldsymbol{x})$后再构建线性超平面.

记核函数κ为:

$$\kappa(\boldsymbol{a}, \boldsymbol{b}) = \phi(\boldsymbol{a})^T\phi(\boldsymbol{b}),$$

其中, $\phi(\cdot)$ 为隐式(implicit) 非线性数据映射, 一般难以确定. 核函数直接计算特征空间的内积, 而非产生具体的特征空间向量, 从而避免选取$\phi(\cdot)$. 例如, 常用的核函数是高斯径向基函数(Gaussian Radial Basis Function, RBF):

$$\kappa(\boldsymbol{a}, \boldsymbol{b}) = \exp\left\{-\gamma\|\boldsymbol{a} - \boldsymbol{b}\|^2\right\}.$$

高斯径向基函数的本质是以数据点与其邻域中的数据点的相似度作为数据变换. 其中的参数γ调节了尺度(可将$1/\gamma$看作窗宽), γ越大, 更倾向于与小邻域的样本观测作相似度运算, 导致RBF形状越为尖峭, 超平面越不光滑, 且非离群值区域通常越小.

除RBF外, 常用的核函数有多项式核函数(Polynomial kernel) 和Sigmoid核函数(Sigmoid kernel) 等. 应该选择哪一种核函数, 以及核函数的参数取值如何, 现实中一般都需要作为超参数调参后确定. 另一方面, 核函数的运算比较复杂, 当数据维度较高时会存在运行时间长、占用内存多等问题. 已经有学者提出了相关的解决方案[16], 这里不再展开.

[16]例如, 可寻找满足

$$\phi(\boldsymbol{a})^T\phi(\boldsymbol{b}) \approx \kappa(\boldsymbol{a}, \boldsymbol{b})$$

的非线性映射$\phi(\cdot)$, 从而避免使用核函数. scikit-learn中6.7 Kernel Approximation介绍了相关方法.

8.7 案例分析

下面, 以R语言rpart包中car.test.frame数据集(简称为数据集car, 存储为car.csv文件) 为例, 识别其中的离群值. 首先读入数据集, 将变量Mileage作为因变量y, 其余变量作为自变量. 随后划分训练集和测试集.

```
alldata = pd.read_csv('car.csv')
y = alldata['Mileage']
X = alldata[['Country', 'Reliability', 'Price', 'Type', '
    Weight', 'Disp.', 'HP']]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=1/3, random_state=1)
```

下面, 仅针对训练集中的自变量识别离群值.

1. 箱线图

预处理: 因为Reliability含有缺失值, 因此作图的统计量取值为nan, 无法作出箱线图; 由于变量尺度不同, 部分变量的箱线图显示不清晰. 下面, 以中位数插补缺失值, 并对变量标准化. 预处理后的数据集为X_trainpreprocess.

```
numeric_features = np.where(X_train.dtypes != object)[0]
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())])
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features)])
X_trainpreprocess = pd.DataFrame(preprocessor.fit_transform(
    X_train), columns=X_train.columns[numeric_features])
```

制作箱线图, 所得结果如图8.9所示.

```
boxX=plt.boxplot(X_trainpreprocess, labels=X_train.columns[
    numeric_features])
plt.show()
```

2. 椭圆包络

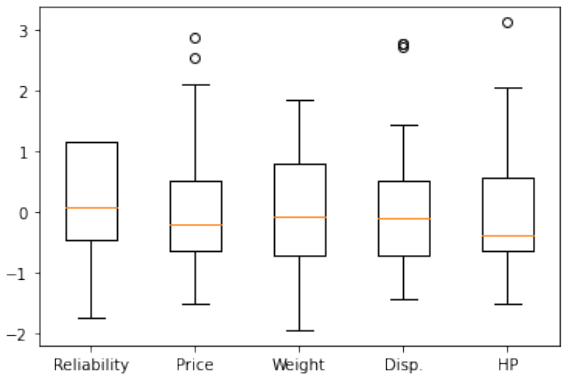


Figure 8.9: car数据集的箱线图

由于EllipticEnvelope函数不允许缺失值, 因此预处理中需处理缺失值, 本例中以中位数插补缺失值(以下代码中仅展示numeric_transformer的设置, 其前后的语句与箱线图中的代码一致, 下同). 随后, 实施椭圆包络方法, 设置离群值比例为0.1.

```
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median'))])
......
cov = EllipticEnvelope(contamination=0.1, random_state=1)
cov.fit(X_trainpreprocess)
```

3. 局部离群点因子

由于LocalOutlierFactor函数不允许缺失值, 因此预处理中需处理缺失值, 本例中以中位数插补缺失值; 由于LOF方法涉及欧氏距离运算, 因此对变量作标准化, 以保证距离运算中各变量尺度一致. 随后, 实施局部离群点因子方法, 设置离群值比例为0.1, 近邻数量为20.

```
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())])
......
lof = LocalOutlierFactor(n_neighbors=20, contamination=0.1)
lofscores = lof.fit_predict(X_trainpreprocess)
```

4. 孤立森林

Table 8.4: 离群值识别代码运行结果

方法	识别出的离群值对应样本编号	
	训练集	测试集
箱线图	10, 12, 27, 33, 40	3, 19
椭圆包络	10, 27, 33, 40	19
局部离群点因子	10, 12, 33, 40	19
孤立森林	27, 33, 36, 40	3, 13, 16, 19
单类支持向量机	17, 28, 30, 36	3, 11

由于IsolationForest函数不允许缺失值, 因此预处理中需处理缺失值, 本例中以中位数插补缺失值. 随后, 实施孤立森林方法, 设置离群值比例为0.1, 孤立树数量20.

```
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median'))])
......
iforest = IsolationForest(n_estimators=20, contamination=0.1,
    random_state=0)
iforest.fit_predict(X_trainpreprocess)
```

5. 单类支持向量机

由于OneClassSVM函数不允许缺失值, 因此预处理中需处理缺失值, 本例中以中位数插补缺失值; 为提高参数迭代优化性能, 对变量作0-1规范化处理. 随后, 实施单类支持向量机方法, 设置离群值所占比例的上界$\nu = 0.1$, 采用线性核函数.

```
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', MinMaxScaler())])
......
clf = OneClassSVM(kernel='linear', nu=0.1)
clf.fit_predict(X_trainpreprocess)
```

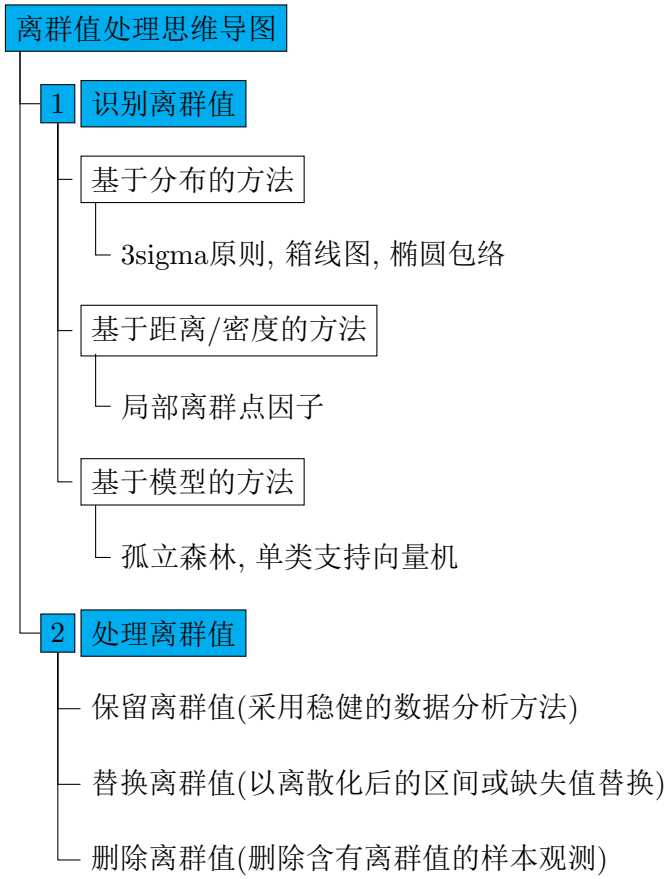
以上代码运行后, 分别识别到训练集和测试集中的离群值如表8.4所示.

可以看到, 几种离群值识别方法所得结果有所不同. 具体实践中需要考虑

方法选择或方法的综合使用.

本章小结

(1) 思维导图



(2) Python函数

表8.5列出了相关离群值识别方法的Python函数.

习题

1. 基于例8.1中的5个样本观测, 采用MCD方法估计$\boldsymbol{\mu}$和Σ. 设置$h = 4$, 设初始抽取到序号为$1, 2, 5$的样本观测, 得到

$$\hat{\boldsymbol{\mu}}_{\text{old}} \approx \begin{pmatrix} 3.00 \\ 4.57 \end{pmatrix}, \ \hat{\boldsymbol{\Sigma}}_{\text{old}} = \frac{1}{n}(\boldsymbol{X} - \bar{\boldsymbol{X}})^T(\boldsymbol{X} - \bar{\boldsymbol{X}}) \approx \begin{pmatrix} 4.99 & 4.37 \\ 4.37 & 3.84 \end{pmatrix}.$$

各样本观测的的MD^2值分别为$2.00, 2.00, 2.38, 5.92, 2.00$. 请计算:

表 8.5: 离群值识别方法的Python实现

方法	函数名称与所属模块	重要参数	应用提示
箱线图	boxplot (matplotlib库pyplot)	-	(1) 函数可允许缺失值, 但不可对含有缺失值的变量绘制箱线图; (2) 为同时展示多个尺度不一变量的箱线图, 可先对变量作标准化.
椭圆包络	EllipticEnvelope (scikit-learn库covariance)	(1) contamination: 离群值所占的比例, 默认值为0.1; (2) support_fraction: 用于参数估计的样本观测所占比例, 默认值为$(n+p+1)/(2n)$; (3) random_state: 随机数种子，用于MCD中随机抽取或排列样本观测.	函数不允许缺失值,使用前应先处理缺失值.
局部离群点因子	LocalOutlierFactor (scikit-learn库neighbors)	(1) contamination: 离群值所占的比例, 默认值为0.1或'auto'(根据版本); (2) n_neighbors: 邻居数量	(1) 函数不允许缺失值, 使用前应先处理缺失值; (2) 由于LOF中涉及距离计算, 应对定量变量做标准化处理, 否则会数据量纲会影响距离.
孤立森林	IsolationForest (scikit-learn库ensemble)	(1) contamination: 离群值所占的比例, 默认值为'auto'; (2) n_estimators: 孤立树的数量; (3) random_state: 随机数种子, 用于构建孤立树时随机抽取分枝变量和分枝临界值.	函数不允许缺失值, 应先处理缺失值.
单类支持向量机	OneClassSVM (scikit-learn库svm)	(1) nu: 离群值所占比例的上界、支持向量所占比例的下界, 默认值为0.5; (2) kernel: 核函数，默认值为'rbf' .	(1) 函数不允许缺失值，应先处理缺失值; (2) 参数的优化求解要求对定量变量作标准化或最小值-最大值规范化.

(1) 集合H;

(2) 基于集合H, 更新参数估计值. 在此基础上, 得到各样本观测的的MD^2值分别为$0.89, 3.00, 1.12, 8.08, 2.99$. 基于这一结果, 集合$H$有变化吗? 如果迭代继续进行, 参数的估计值会有变化吗? [17]

2. 设有5个样本观测, 取值分别为1,2,3,5,6, 取$k=2$, 采用Manhattan距离, 请回答:

(1) 各样本观测的LOF值为多少? (请制作类似于表8.3的表格展示过程)

(2) 请采用scikit-learn中的LocalOutlierFactor函数计算LOF值. 对比(1)中的计算结果, 你有何发现?

提示：两个向量$\boldsymbol{a}=(a_1,\cdots,a_p), \boldsymbol{b}=(b_1,\cdots,b_p)$的Manhattan距离是：$\frac{\sum_{i=1}^{p}|a_i-b_i|}{p}$.

3. 采用例8.5中的数据, 请回答:

(1) 基于以下条件，建立1棵深度为2的孤立树:

- 第1次抽取的变量为X_2, 随后在$[0.7, 10]$中随机抽取的值为1;
- 第2次抽取的变量为X_1, 随后在$[-2.1, 10]$中随机抽取的值为7.

[17]本例表明, 由MCD方法所得的参数估计值会受到初始抽样的影响; 若初始入样的样本观测中包含离群值, 则由MCD方法所得的参数估计值可能受到离群值的影响.

(2) 在这棵孤立树中, 数据点$(10, 10)$的路径长度是多少?

(3) 合并这棵孤立树与图8.6所示的孤立树, 形成孤立森林. 数据点$(10, 10)$的平均路径长度是多少? 异常分是多少?

4. 基于例8.1中的5个样本观测, 采用scikit-learn中的OneClassSVM函数, 设置nu=0.2, kernel='linear', 得到各样本观测对应的α值分别为$0, 0, 0, 0.98, 0.02$. 请回答:

(1) 哪些样本观测是支持向量? 支持向量所占比例是多少?

(2) 哪些样本观测是离群值? 离群值所占比例是多少?

References

[1] Hawkings D.M. (1980) Identification of Outliers. London: Chapman and Hall.

[2] Tukey J.W. (1977) Exploratory Data Analysis. Boston: Addison-Wesley.

[3] Vandervieren E., Hubert M. (2004) An adjusted boxplot for skewed distributions. Computational Statistics & Data Analysis, 52(12):5186-5201.

[4] Rousseeuw J.P., Katrien van D. (1999) A fast algorithm for the minimum covariance determinant estimator. Technometrics, 41(3): 212-223.

[5] Breunig M.M., Kriegel H.P., Ng R.T., Sander J. (2000) LOF: Identifying density-based local outliers. Proc. ACM SIGMOD 2000 Int. Conf. On Management of Data, Dalles, TX.

[6] Schölkopf B., Platt J.C., Shawe-Taylor J., Smola A.J., Williamson R.C. (2001). Estimating the support of a high-dimensional distribution. Neural Computation.

[7] 吴翌琳, 房祥忠(2016) 大数据探索性分析. 北京:中国人民大学出版社.

[8] 周志华(2016) 机器学习. 北京:清华大学出版社.

[9] Liu F.T., Ting K.M., Zhou Z.H. (2008) Isolation Forest. ?