菜菜的机器学习sklearn第十期

sklearn中的朴素贝叶斯

小伙伴们晚上好~o(￣▽￣)ブ

我是菜菜，这里是我的sklearn课堂第十期，今晚的直播内容是朴素贝叶斯~

我的开发环境是**Jupyter lab**，所用的库和版本大家参考：

Python 3.7.1（你的版本至少要3.4以上

Scikit-learn 0.20.1 （你的版本至少要0.20

Numpy 1.15.4, **Pandas** 0.23.4, **Matplotlib** 3.0.2, **SciPy** 1.1.0

请扫码进群领取课件和代码源文件，**扫描二维码后回复"K"就可以进群哦~**

菜菜的机器学习sklearn第十期
sklearn中的朴素贝叶斯

1 概述

1.1 真正的概率分类器

在许多分类算法应用中，特征和标签之间的关系并非是决定性的。比如说，我们想预测一个人究竟是否会在泰坦尼克号海难中生存下来，那我们可以建一棵决策树来学习我们的训练集。在训练中，其中一个人的特征为：30岁，男，普通舱，他最后在泰坦尼克号海难中去世了。当我们测试的时候，我们发现有另一个人的特征也为：30岁，男，普通舱。基于在训练集中的学习，我们的决策树必然会给这个人打上标签：去世。然而这个人的真实情况一定是去世了吗？并非如此。

也许这个人是心脏病患者，得到了上救生艇的优先权。又有可能，这个人就是挤上了救生艇，活了下来。对分类算法来说，基于训练的经验，这个人"很有可能"是没有活下来，但算法永远也无法确定"这个人一定没有活下来"。即便这个人最后真的没有活下来，算法也无法确定基于训练数据给出的判断，是否真的解释了这个人没有存活下来的真实情况。这就是说，**算法得出的结论，永远不是100%确定的，更多的是判断出了一种"样本的标签更可能是某类的可能性"，而非一种"确定"**。我们通过某些规定，比如说，在决策树的叶子节点上占比较多的标签，就是叶子节点上所有样本的标签，来强行让算法为我们返回一个固定结果。但许多时候，我们也希望能够理解算法判断出的可能性本身。

每种算法使用不同的指标来衡量这种可能性。比如说，决策树使用的就是叶子节点上占比较多的标签所占的比例（接口predict_proba调用），逻辑回归使用的是sigmoid函数压缩后的似然（接口predict_proba调用），而SVM使用的是样本点到决策边界的距离（接口decision_function调用）。但这些指标的本质，其实都是一种"类概率"的表示，我们可以通过归一化或sigmoid函数将这些指标压缩到0~1之间，让他们表示我们的模型对预测的结果究竟有多大的把握（置信度）。但无论如何，我们都希望使用真正的概率来衡量可能性，因此就有了真正的概率算法：朴素贝叶斯。

朴素贝叶斯是一种直接衡量标签和特征之间的概率关系的有监督算法，它既可以做回归也可以分类，不过多是用于分类之中。朴素贝叶斯的算法根源就是基于概率论和数理统计的贝叶斯理论，因此它是根正苗红的概率模型。接下来，我们就来认识一下这个简单快速的概率算法。

1.2 冬眠的瓢虫：朴素贝叶斯是如何工作的

朴素贝叶斯被认为是最简单的分类算法之一。首先，我们需要了解一些概率论的基本理论。假设有两个随机变量X和Y，他们分别可以取值为x和y。有这两个随机变量，我们可以定义两种概率：

关键概念：联合概率与条件概率
联合概率：“X取值为x”和“Y取值为y”两个事件同时发生的概率，表示为$P(X=x,Y=y)$ **条件概率**：在”X取值为x“的前提下，”Y取值为y“的概率，表示为$P(Y=y\|X=x)$

举个例子，我们让X为"气温"，Y为“七星瓢虫冬眠”，则X和Y可能的取值分为别x和y，其中x = {0,1}，0表示没有下降到0度以下，1表示下降到了0度以下。y = {0,1}，其中0表示否，1表示是。

两个事件分别发生的概率就为：

$P(X=1)$ = 50%，则是说明，气温下降到0度以下的可能性为50%，则$P(X=0)=1-P(X=1)$ = 50%。

$P(Y=1)$ = 70%，则是说明，七星瓢虫会冬眠的可能性为70%，则$P(Y=0)=1-P(Y=1)$ = 30%。

则这两个事件的联合概率为$P(X=1，Y=1)$，这个概率代表了气温下降到0度以下和七星瓢虫去冬眠这两件事情**同时，独立**发生的概率。

而两个事件之间的条件概率为$P(Y=1|X=1)$，这个概率代表了，当气温下降到0度以下这个条件被满足之后，七星瓢虫会去冬眠的概率。也就是说，气温下降到0度以下，一定程度上影响了七星瓢虫去冬眠这个事件。

在概率论中，我们可以证明，两个事件的联合概率等于这两个事件任意条件概率 * 这个条件事件本身的概率。

$$P(X=1,Y=1) = P(Y=1|X=1)*P(X=1) = P(X=1|Y=1)*P(Y=1)$$

简单一些，则可以将上面的式子写成：

$$P(X,Y) = P(Y|X)*P(X) = P(X|Y)*P(Y)$$

由上面的式子，我们可以得到贝叶斯理论等式：

$$P(Y|X) = \frac{P(X|Y)*P(Y)}{P(X)}$$

而这个式子，就是我们一切贝叶斯算法的根源理论。我们可以把我们的特征X当成是我们的条件事件，而我们要求解的标签Y当成是我们被满足条件后会被影响的结果，而两者之间的概率关系就是$P(Y|X)$，这个概率在机器学习中，被我们称之为是标签的后验概率（posterior probability），即是说我们先知道了条件，再去求解结果。而标签Y在没有任何条件限制下取值为某个值的概率，被我们写作$P(Y)$，与后验概率相反，这是完全没有任何条件限制的，标签的先验概率（prior probability）。而我们的$P(X|Y)$被称为“类的条件概率”，表示当Y的取值固定的时候，X为某个值的概率。那现在，有趣的事情就出现了。

1.2.1 P(Y|X)

假设，我们依然让X是“气温”，这就是我们的特征，Y是“七星瓢虫冬眠”，就是我们的标签。现在，我们建模的目的是，预测七星瓢虫是否会冬眠。在许多教材和博客里，大家会非常自然地开始说，我们现在求的就是我们的$P(Y|X)$，然后根据贝叶斯理论等式开始做各种计算和分析。现在请问大家，我写作$P(Y|X)$的这个概率，代表了什么呢？更具体一点，这个表达，可以代表多少种概率呢？

P(Y|X)代表了多少种情况的概率?

$P(Y=1|X=1)$ 气温0度以下的条件下，七星瓢虫冬眠的概率
$P(Y=1|X=0)$ 气温0度以上的条件下，七星瓢虫冬眠的概率
$P(Y=0|X=1)$ 气温0度以下的条件下，七星瓢虫没有冬眠的概率
$P(Y=0|X=0)$ 气温0度以上的条件下，七星瓢虫没有冬眠的概率

数学中的第一个步骤，也就是最重要的事情，就是定义清晰。现在我们的Y有两种取值，而X也有两种取值，就让概率$P(Y|X)$的定义变得很模糊，排列组合之后竟然有4种可能。在机器学习当中，一个特征X下取值可能远远不止两种，标签也可能是多分类的，还会有多个特征，排列组合以下，到底求解的$P(Y|X)$是什么东西，是一个太让人感到混淆的点。同理，$P(Y)$随着标签中分类的个数，可以有不同的取值。$P(X|Y)$也是一样。在这里，我们来为大家澄清：

机器学习中的简写$P(Y)$，通常表示标签取到少数类的概率，少数类往往使用正样本表示，也就是$P(Y=1)$，本质就是所有样本中标签为1的样本所占的比例。如果没有样本不均衡问题，则必须在求解的时候明确，你的Y的取值是什么。

而$P(Y|X)$是对于任意一个样本而言，如果这个样本的特征X的取值为1，则表示求解$P(Y=1|X=1)$。如果这个样本下的特征X取值为0，则表示求解$P(Y=1|X=0)$。也就是说，$P(Y|X)$是具体到每一个样本上的，究竟求什么概率，由样本本身的特征的取值决定。每个样本的$P(Y|X)$如果大于阈值0.5，则认为样本是少数类（正样本，1），如果这个样本的$P(Y|X)$小于阈值0.5，则认为样本是多数类（负样本，0或者-1）。如果没有具体的样本，只是说明例子，则必须明确$P(Y|X)$中X的取值。

在机器学习当中，对每一个样本，我们不可能只有一个特征X，而是会存在着包含n个特征的特征向量$\boldsymbol{X}$。因此机器学习中的后验概率，被写作$P(Y|\boldsymbol{X})$，其中$\boldsymbol{X}$中包含样本在n各特征上的取值$X_1, X_2, X_3 .... X_n$。以此为基础，机器学习中，**对每一个样本**我们有：

$$P(Y=1|X_1, X_2 .... X_n) = \frac{P(X_1, X_2 .... X_n|Y=1) * P(Y=1)}{P(X_1, X_2 .... X_n)}$$

对于分子而言，$P(Y=1)$就是少数类占总样本量的比例，$P(X_1, X_2 .... X_n|Y=1)$则需要稍微复杂一点的过程来求解。假设我们只有两个特征，由联合概率公式，我们可以有如下证明：

$$\begin{aligned} P(X_1, X_2|Y=1) &= \frac{P(X_1, X_2, Y=1)}{P(Y=1)} \\ &= \frac{P(X_1, X_2, Y=1)}{P(X_2, Y=1)} * \frac{P(X_2, Y=1)}{P(Y=1)} \\ &= P(X_1|X_2, Y=1) * P(X_2|Y=1) \end{aligned}$$

假设X_1和X_2之是有条件独立的：

$$= P(X_1|Y=1) * P(X_2|Y=1)$$

是一种若推广到n个X上，则有：

$$P(\boldsymbol{X} = X_1, X_2, ... X_n|Y=1) = \prod_{i=1}^{n} P(X_i|Y=1)$$

这个式子证明，在Y=1的条件下，X的多种取值被同时取到的概率，就等于Y=1的条件下，X的多种取值被分别取到的概率相乘。其中，假设X_1与X_2是有条件独立则可以让公式$P(X_1|X_2, Y=1) = P(X_1|Y=1)$，这是在假设$X_2$是一个对$X_1$在某个条件下的取值完全无影响的变量。

比如说，温度(X_1)与观察到的瓢虫的数目(X_2)之间的关系。有人可能会说，温度在零下的时候，观察到的瓢虫数目往往很少。这种关系的存在可以通过一个中间因素：瓢虫会冬眠(Y)来解释。冬天瓢虫都冬眠了，自然观察不到很多瓢虫出来活动。也就是说，如果瓢虫冬眠的属性是固定的($Y=1$)，那么观察到的温度和瓢虫出没数目之间关系就会消失，因此无论是否还存在着"观察到的瓢虫的数目"这样的因素，我们都可以判断这只瓢虫到底会不会冬眠。这种情况下，我们就说，温度与观察到的瓢虫的数目，是条件独立的。

假设特征之间是有条件独立的，可以解决众多问题，也简化了很多计算过程，这是朴素贝叶斯被称为"朴素"的理由。因此，贝叶斯在特征之间有较多相关性的数据集上表现不佳，而现实中的数据多多少少都会有一些相关性，所以贝叶斯的分类效力在分类算法中不算特别强大。但无论如何，有了这个式子，我们就可以求解出我们的分子了。

对于贝叶斯理论等式的分母而言，我们可以使用全概率公式来求解$P(\boldsymbol{X})$：

$$P(\boldsymbol{X}) = \sum_{i=1}^{m} P(Y_i) * P(\boldsymbol{X}|Y_i)$$

其中m代表标签的种类，也就是说，对于二分类而言我们有：

$$P(\boldsymbol{X}) = P(Y=1) * P(\boldsymbol{X}|Y=1) + P(Y=0) * P(\boldsymbol{X}|Y=0)$$

基于这个方程，我们可以来求解一个非常简单的例子下的后验概率。

索引	温度（X_1）	瓢虫的年龄（X_2）	瓢虫冬眠
0	零下	10天	是
1	零下	20天	是
2	零上	10天	否
3	零下	一个月	是
4	零下	20天	否
5	零上	两个月	否
6	零下	一个月	否
7	零下	两个月	是
8	零上	一个月	否
9	零上	10天	否
10	零下	20天	否

现在，我们希望预测零下的时候，年龄为20天的瓢虫，是否会冬眠。

$$P(Y=1|X_1,X_2)=\frac{P(X_1,X_2|Y=1)*P(Y=1)}{P(X_1,X_2)}$$

$$P(\text{冬眠}|\text{零下},20\text{天})=\frac{P(\text{零下},20\text{天}|\text{冬眠})*P(\text{冬眠})}{P(\text{零下},20\text{天})}$$

$$=\frac{P(\text{零下}|\text{冬眠})*P(20\text{天}|\text{冬眠})*P(\text{冬眠})}{P(\text{冬眠})*P(\text{零下},20\text{天}|\text{冬眠})+P(\text{不冬眠})*P(\text{零下},20\text{天}|\text{不冬眠})}$$

对于分子我们可以求得：

$$P(\text{冬眠})=\frac{4}{11},\quad P(\text{零下}|\text{冬眠})=\frac{4}{4}=1,\quad P(20\text{天}|\text{冬眠})=\frac{1}{4}$$

对于分母我们可以求得：

$$P(\text{零下},20\text{天}|\text{冬眠})=\frac{1}{4},\ P(\text{不冬眠})=\frac{7}{11},\ P(\text{零下},20\text{天}|\text{不冬眠})=\frac{2}{7}$$

所以我们的，温度为零下的时候， 生活了20天的瓢虫，会冬眠的概率为：

$$\frac{4/11*1*1/4}{4/11\ *1/4\ +\ 7/11\ *2/7}=\frac{1/11}{3/11}=0.33$$

设定阈值为0.5，假设大于0.5的就被认为是会冬眠，小于0.5的就被认为是不会冬眠。根据我们的计算，我们认为一个在零下条件下，年龄为20天的瓢虫，是不会冬眠的。这就完成了一次预测。但是这样，有趣的地方又来了。刚才的预测过程是没有问题的。但我们总是好奇，这个过程如何对应sklearn当中的fit和transform呢？这个决策过程中，我们的训练集和我的测试集分别在哪里？以及，算法建模建模，我的模型在哪里呢？

1.2.2 与众不同的朴素贝叶斯

在过去的许多个星期内，我们学习的分类算法总是有一个特点：这些算法先从训练集中学习，获取某种信息来建立模型，然后用模型去对测试集进行预测。比如逻辑回归，我们要先从训练集中获取让损失函数最小的参数，然后用参数建立模型，再对测试集进行预测。在比如支持向量机，我们要先从训练集中获取让边际最大的决策边界，然后用决策边界对测试集进行预测。相同的流程在决策树，随机森林中也出现，我们在fit的时候必然已经构造好了能够让对测试集进行判断的模型。而朴素贝叶斯，似乎没有这个过程。

我给了大家一张有标签的表，然后提出说，我要预测零下的时候，年龄为20天的瓢虫，会冬眠的概率，然后我们就顺理成章地算了出来。没有利用训练集求解某个模型的过程，也没有训练完毕了我们来做测试的过程，而是直接对有标签的数据提出要求，就可以得到预测结果了。

这说明，**朴素贝叶斯是一个不建模的算法。**以往我们学的不建模算法，比如KMeans，比如PCA，都是无监督学习，而朴素贝叶斯是第一个有监督的，不建模的分类算法。在我们刚才举的例子中，有标签的表格就是我们的训练集，而我提出的要求"零下的时候，年龄为20天的瓢虫"就是没有标签的测试集。我们认为，训练集和测试集都来自于同一个不可获得的大样本下，并且这个大样本下的各种属性所表现出来的规律应当是一致的，因此训练集上计算出来的各种概率，可以直接放到测试集上来使用。即便不建模，也可以完成分类。

但实际中，贝叶斯的决策过程并没有我们给出的例子这么简单。

$$P(Y=1|X_1,X_2....X_n)=\frac{P(Y=1)*\prod_{i=1}^{n}P(X_i|Y=1)}{P(X_1,X_2....X_n)}$$

对于这个式子来说，从训练集中求解$P(Y=1)$很容易，但$P(X_1,X_2....X_n)$和$P(X_1,X_2....X_n|Y=1)$这一部分就没有这么容易了。在我们的例子中，我们通过全概率公式来求解分母，两个特征就求解了四项概率。随着特征数目的逐渐变多，分母上的计算两会成指数级增长，而分子中的$P(X_i|Y=1)$也越来越难计算。

不过幸运的是，对于同一个样本来说，在二分类状况下我们可以有：

$$P(Y=1|X_1,X_2....X_n)=\frac{P(Y=1)*\prod_{i=1}^{n}P(X_i|Y=1)}{P(X_1,X_2....X_n)}$$

$$P(Y=0|X_1,X_2....X_n)=\frac{P(Y=0)*\prod_{i=1}^{n}P(X_i|Y=0)}{P(X_1,X_2....X_n)}$$

并且：

$$P(Y=1|X_1,X_2....X_n)+P(Y=0|X_1,X_2....X_n)=1$$

在分类的时候，我们选择$P(Y=1|X_1,X_2,...X_n)$和$P(Y=0|X_1,X_2,...X_n)$中较大的一个所对应的Y的取值，作为这个样本的分类。在比较两个类别的时候，两个概率计算的分母是一致的，因此我们可以不用计算分母，只考虑分子的大小。当我们分别计算出分子的大小之后，就可以通过让两个分子相加，来获得分母的值，以此来避免计算$P(X_1,X_2,...X_n)$。这个过程，被我们称为"最大后验估计"（MAP）。在最大后验估计中，我们只需要求解分子，主要是求解$P(X_i|Y)$，就能够获得相应的概率。

虽然朴素贝叶斯使用了过于简化的假设，这个分类器在许多实际情况中都运行良好，著名的是文档分类和垃圾邮件过滤。而且由于贝叶斯是从概率角度进行估计，它所需要的样本量比较少。当然，如果样本量少于特征数目，贝叶斯的效果就会被削弱。

与SVM和随机森林相比，朴素贝叶斯运行速度更快，因为求解$P(X_i|Y)$本质是在每个特征上单独对概率进行计算，然后再求乘积，所以每个特征上的计算可以是独立并且并行的，因此贝叶斯的计算速度比较快。不过相对的，贝叶斯的运行效果不是那么好，所以贝叶斯的接口调用的predict_proba其实也不是总指向真正的分类结果。

1.3 sklearn中的朴素贝叶斯

如我们刚才所说，我们在贝叶斯中，需要重点求解贝叶斯概率公式中的分子。

在现实中，要求解分子也会有各种各样的问题。我们可能面临特征非常多，而每个特征下的分类也非常多的情况，像我们刚才那样去手动数出概率，需要极多的计算资源。并且，可能会出现，测试集中出现的某种概率组合，是训练集中从未出现的状况，这种时候就会出现某一个概率为0的情况，分子就会为0，这种情况下的概率预测就是无效的。还有，现实中的大多数标签还是连续型变量，要处理连续型变量的概率，就不是单纯的数样本个数的占比的问题了。求解连续型变量的概率，需要引入各种概率论中的数字分布，使用各种分布下的概率密度曲线来估计一个概率。其中涉及的数学过程是极其复杂的，要求大家必须要熟悉概率论和微积分。

其中，我们最常用的几个分布分别是：高斯分布，伯努利分布和多项式分布。sklearn中，基于这些分布以及这些分布上的概率估计的改进，为我们提供了四个朴素贝叶斯的分类器。

类	含义
naive_bayes.BernoulliNB	伯努利分布下的朴素贝叶斯
naive_bayes.GaussianNB	高斯分布下的朴素贝叶斯
naive_bayes.MultinomialNB	多项式分布下的朴素贝叶斯
naive_bayes.ComplementNB	补充朴素贝叶斯
linear_model.BayesianRidge	贝叶斯岭回归，在参数估计过程中使用贝叶斯回归技术来包括正则化参数

2 不同分布下的贝叶斯

2.1 高斯朴素贝叶斯GaussianNB

class `sklearn.naive_bayes.GaussianNB` (*priors=None*, *var_smoothing=1e-09*)

高斯朴素贝叶斯，通过假设$P(X_i|Y)$是服从高斯分布（也就是正态分布），来估计每个特征下每个类别上的条件概率。对于每个特征下的取值，高斯朴素贝叶斯有如下公式：

$$P(X_i|Y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp(-\frac{(X_i-\mu_y)^2}{2\sigma_y^2})$$

高斯朴素贝叶斯以最大化$P(X_i|Y)$为目标，来求解该式中的参数σ_y和μ_y。求解出参数后，带入一个X_i的值，就能够得到一个$P(X_i|Y)$的概率取值。这个类包含两个参数：

参数	含义
prior	可输入任何类数组结构，形状为（n_classes，） 表示类的先验概率。如果指定，则不根据数据调整先验，如果不指定，则自行根据数据计算先验概率$P(Y)$。
var_smoothing	浮点数，可不填（默认值= 1e-9） 在估计方差时，为了追求估计的稳定性，将所有特征的方差中最大的方差以某个比例添加到估计的方差中。这个比例，由var_smoothing参数控制。

但在实例化的时候，我们不需要对高斯朴素贝叶斯类输入任何的参数，可以说是一个非常轻量级的类。所以很遗憾的是，贝叶斯也没有太多的参数可以调整，调用的接口也全部都是我们曾经见过的。

无论如何，先来进行一次预测试试看吧：

1. **导入需要的库和数据**

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.naive_bayes import GaussianNB
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split

digits = load_digits()
X, y = digits.data, digits.target

Xtrain,Xtest,Ytrain,Ytest = train_test_split(X,y,test_size=0.3,random_state=420)
```

2. **建模，探索建模结果**

```
gnb = GaussianNB().fit(Xtrain,Ytrain)

#查看分数
acc_score = gnb.score(Xtest,Ytest)

#查看预测结果
Y_pred = gnb.predict(Xtest)

#查看预测的概率结果
prob = gnb.predict_proba(Xtest)

prob.shape
```

3. **使用混淆矩阵来查看贝叶斯的分类结果**

```
from sklearn.metrics import confusion_matrix as CM
CM(Ytest,Y_pred)
```

混淆矩阵和精确性可以帮助我们了解贝叶斯的分类结果。然而，我们选择贝叶斯进行分类，大多数时候都不是为了单单追求效果，而是希望看到预测的相关概率。这种概率给出预测的可信度，所以对于概率类模型，我们希望能够由其他的模型评估指标来帮助我们判断，模型在“概率预测”这项工作上，完成得如何。接下来，我们就来看看概率模型独有的评估指标。

2.2 概率类模型的评估指标

2.2.1 布里尔分数

概率预测的准确程度被称为“校准程度”，是衡量算法预测出的概率和真实概率的差异的一种方式。在二分类中，最常用的指标叫做布里尔分数，它被计算为是概率预测相对于测试样本的均方误差，表示为：

$$Brier\ Score = \frac{1}{N}\sum_{i=1}^{n}(p_i - o_i)^2$$

其中N是样本数量，p_i为朴素贝叶斯预测出的概率，o_i是样本所对应的真实结果，只能取到0或者1，如果事件发生则为1，如果不发生则为0。这个指标衡量了我们的概率距离真实标签结果的差异，其实看起来非常像是均方误差。布里尔分数的范围是从0到1，分数越高则贝叶斯的预测结果越差劲。由于它的本质也是在衡量一种损失，所以在sklearn当中，布里尔得分被命名为brier_score_loss。我们可以从模块metrics中导入这个分数来衡量我们的模型评估结果：

```
from sklearn.metrics import brier_score_loss
brier_score_loss(Ytest, prob[:,0], pos_label=0)
#我们的pos_label与prob中的索引一致，就可以查看这个类别下的布里尔分数是多少
```

基于布里尔分数，我们可以构筑概率校准曲线，来查看我们的模型的效果。当然，对于校准效果不理想的模型，我们还可以使用一些方法来帮助他们改善概率预测结果。

【完整版】2.2.2 使用布里尔分数绘制概率校准曲线

【完整版】2.2.3 修正概率校准效果不好的模型

【完整版】2.3 伯努利贝叶斯BernoulliNB

class `sklearn.naive_bayes.BernoulliNB`(*alpha=1.0*, *binarize=0.0*, *fit_prior=True*, *class_prior=None*)

【完整版】2.4 多项式朴素贝叶斯MultinomialNB

class `sklearn.naive_bayes.MultinomialNB`(*alpha=1.0*, *fit_prior=True*, *class_prior=None*)

【完整版】2.4.1 多项式朴素贝叶斯的参数与属性

【完整版】2.4.2 朴素贝叶斯的样本不平衡问题

【完整版】2.5 补充朴素贝叶斯ComplementNB

class `sklearn.naive_bayes.ComplementNB`(*alpha=1.0*, *fit_prior=True*, *class_prior=None*, *norm=False*)

【完整版】3 朴素贝叶斯vs其他模型

【完整版】3.1 朴素贝叶斯擅长的数据集

【完整版】3.2 朴素贝叶斯与SVM在的学习曲线

【完整版】4 贝叶斯岭回归

【完整版】5 案例：多项式朴素贝叶斯的文本分类

【完整版】6 朴素贝叶斯的参数，接口与属性列表