R 机器学习

第 09 讲　朴素贝叶斯

张敬信

2022 年 10 月 27 日

哈尔滨商业大学

一. 朴素贝叶斯原理

- 朴素贝叶斯属于概率模型，是专门用于解决分类问题的经典数据挖掘算法。其思想就是**贝叶斯定理**，通过已知类别的训练数据，计算样本的先验概率，再利用贝叶斯公式计算未知类别样本属于某个类别的后验概率，最终以最大后验概率所对应的类别作为样本的预测类别。
- **优点**：运算简单且高效；分类效率稳定；对缺失数据和异常数据不敏感。
- **缺点**：模型结果依赖于先验概率，故分类结果存在一定的错误率；要求自变量 X 具有相同的特征（如均为数值型或离散型或 0-1 型），要求自变量之间的独立性和连续变量近似服从正态分布；模型的前提假设在有些实际应用中很难满足。
- 朴素贝叶斯常用于垃圾邮件识别、手写字体识别、广告推荐系统、医疗诊断、欺诈识别、客户流失、投资决策、信用评级等。

1. 贝叶斯假设

- 假设在给定类别 C 下，各个输入特征 $X_i,\ i = 1,\ \cdots,\ n$ 是相互独立的：

$$P(X_1,\ X_2,\ \cdots,\ X_n|C) = P(X_1|C) \cdot P(X_2|C) \cdots P(X_n|C)$$

- 例如，一个水果是红色、圆的、直径约 10cm，则它可能是苹果。朴素贝叶斯分类器要求各个特征为"$\Pr\{$该水果是苹果$\}$"的贡献是相互独立的，不考虑颜色、圆形度、直径之间的相关性。

2. 贝叶斯定理

- 条件概率公式：

$$P(B|A) = \frac{P(AB)}{P(A)}$$

表示在已知事件 A 发生的条件下事件 B 发生的概率，其中 $P(AB)$ 表示事件 A 与事件 B 同时发生的概率。

- 根据条件概率公式，可得到乘法概率公式：

$$P(AB) = P(A)P(B|A) = P(B)P(A|B)$$

- 回顾全概率公式：

$$P(A)=\sum_{k=1}^{K}P(AB_k)=\sum_{k=1}^{K}P(B_k)P(A|B_k)$$

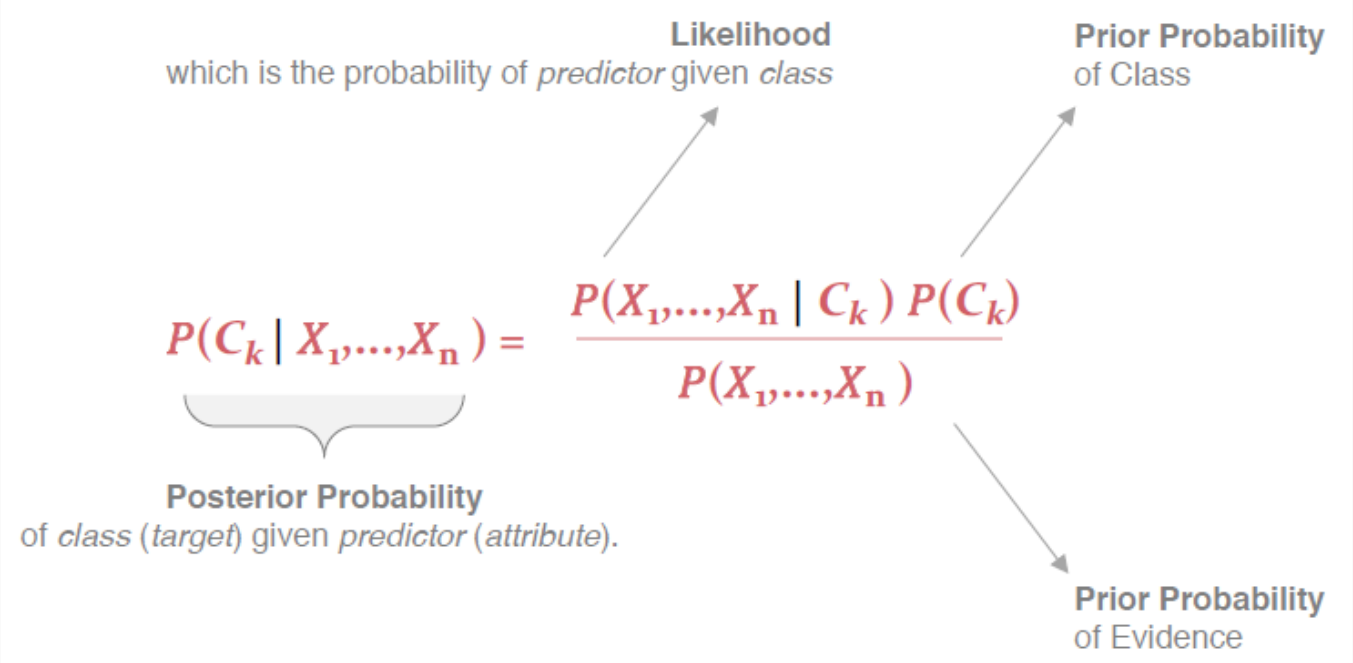
其中，事件 $B_1,\cdots,B_K$ 构成了一个完备的事件组，且每个 $P(B_i)$ 均大于 0。该公式表示，对任一事件 A 都可以表示成 K 个完备事件组每个事件发生概率与该条件下 A 发生的概率乘积的和。

- 由条件概率公式与全概率公式，就能得到贝叶斯公式，即在已知 X 的条件下，计算样本属于某个类别的概率：

$$P(C_k|X)=\frac{P(C_kX)}{P(X)}=\frac{P(C_k)P(X|C_k)}{\sum_{k=1}^{K}P(C_k)P(X|C_k)}$$

其中，C_k 表示样本所属的某类别。

贝叶斯思想



- **估计后验概率：**贝叶斯概率度量的是”信念度”，为了计算假设概率，贝叶斯法是先得到**先验概率**，再根据新的相关数据（证据）将其更新为后验概率。
  - 在观察数据之前，先验信念可用先验概率分布表示（代表我们对未知特征的所了解的知识）
  - 有了观察数据后，将其用于更新先验概率分布为后验概率分布，从而获得后验信念。

3. 朴素贝叶斯算法

- 利用贝叶斯公式可计算出样本 X 属于每个类别 C_k 的概率 $P(C_k|X)$，则其中概率值最大的 $P(C_k|X)$ 对应的类别，即为样本的预测类别：

$$\hat{y} = \underset{k\in\{1,\cdots,K\}}{\operatorname{argmax}} P(C_k|X) = \underset{k\in\{1,\cdots,K\}}{\operatorname{argmax}} \frac{P(C_k)P(X|C_k)}{P(X)}$$

- 注意，上式分母 $P(X) = \sum_{k=1}^{n} P(C_k)P(X|C_k)$ 对所有类别来说都是相同的，故不用考虑。
- 而由贝叶斯独立性假设，

$$\begin{aligned} P(X|C_k) &= P(X_1, X_2, \cdots, X_n|C_k) \\ &= P(X_1|C_k)P(X_2|C_k)\cdots P(X_n|C_k) \end{aligned}$$

- 从而，朴素贝叶斯模型可表示为：

$$\hat{y} = \underset{k \in \{1,\cdots,K\}}{\operatorname{argmax}} P(C_k) \prod_{i=1}^{n} P(X_i|C_k)$$

- 具体计算：$P(C_k)$ 通常是用训练样本各类别的占比（频率代替概率），

注：特征间的独立性越强，贝叶斯分类器的分类效果越好。

4. 朴素贝叶斯分类的示例

- 现有玩网球相关的天气统计数据：

Day	Weather	Temperature	Humidity	Wind	Play
1	晴	热	高	弱	No
2	晴	热	高	强	No
3	阴	热	高	弱	Yes
4	雨	适中	高	弱	Yes
5	雨	冷	正常	弱	Yes
6	雨	冷	正常	强	No
7	阴	冷	正常	强	Yes
8	晴	适中	高	弱	No
9	晴	冷	正常	弱	Yes
10	雨	适中	正常	弱	Yes
11	晴	适中	正常	强	Yes
12	阴	适中	高	强	Yes
13	阴	热	正常	弱	Yes
14	雨	适中	高	强	No

- 若新样本 X = (Weaher= 晴, Temperature= 冷, Humidity= 高, Wind= 强), 问是否去打网球?

- 首先要做一个 lookup 表:

Weather	Play=Yes	Play=No	Total
晴	2/9	3/5	5/14
阴	4/9	0/5	4/14
雨	3/9	2/5	5/14

Temperature	Play=Yes	Play=No	Total
热	2/9	2/5	4/14
适中	4/9	2/5	6/14
冷	3/9	1/5	4/14

Humidity	Play=Yes	Play=No	Total
高	3/9	4/5	7/14
正常	6/9	1/5	7/14

Wind	Play=Yes	Play=No	Total
强	3/9	3/5	6/14
弱	6/9	2/5	8/14

```r
library(tidyverse)

df = readxl::read_xlsx("datas/PlayTennis.xlsx",
                       range = "A1:F15")
df %>%
  count(Weather, Play) %>%
  pivot_wider(names_from = Play, values_from = n,
              values_fill = list(n = 0)) %>%
  mutate(Total = Yes + No,
         across(where(is.numeric), ~ .x / sum(.x)))
#> # A tibble: 3 x 4
#>   Weather    No   Yes Total
#>   <chr>   <dbl> <dbl> <dbl>
#> 1 晴        0.6 0.222 0.357
#> 2 阴        0   0.444 0.286
#> 3 雨        0.4 0.333 0.357
```

从这些 lookup 表中可以提取需要的先验概率：

- play=Yes 下 X 的似然概率：
  - P(Weather= 晴 |Play=Yes) = 2/9
  - P(Temperature= 冷 |Play=Yes) = 3/9
  - P(Humidity= 高 |Play=Yes) = 3/9
  - P(Wind= 强 |Play=Yes) = 3/9
- play=No 下 X 的似然概率：
  - P(Weather= 晴 |Play=No) = 3/5
  - P(Temperature= 冷 |Play=No) = 1/5
  - P(Humidity= 高 |Play=No) = 4/5
  - P(Wind= 强 |Play=No) = 3/5
- 结果类别的先验概率：
  - P(Play=Yes) = 9/14
  - P(Play=No) = 5/14

- 根据朴素贝叶斯算法计算后验概率：

$$X=\{\text{天气}=\text{晴},\ \text{温度}=\text{冷},\ \text{湿度}=\text{高},\ \text{风力}=\text{强}\}$$

$$P\{\text{Play}=\text{是}\mid X\}=\frac{P\{X\mid \text{Play}=\text{是}\}\cdot P\{\text{Play}=\text{是}\}}{P\{X\}}$$

$$=\frac{P\{\text{天气}=\text{晴}\mid \text{Play}=\text{是}\}\cdot P\{\text{温度}=\text{冷}\mid \text{Play}=\text{是}\}\cdot P\{\text{湿度}=\text{高}\mid \text{Play}=\text{是}\}\cdot P\{\text{风力}=\text{强}\mid \text{Play}=\text{是}\}\cdot P\{\text{Play}=\text{是}\}}{P\{\text{天气}=\text{晴}\}\cdot P\{\text{温度}=\text{冷}\}\cdot P\{\text{湿度}=\text{高}\}\cdot P\{\text{风力}=\text{强}\}}$$

$$=\frac{(2/9)\cdot(3/9)\cdot(3/9)\cdot(3/9)\cdot(9/14)}{(5/14)\cdot(4/14)\cdot(7/14)\cdot(6/14)}$$

$$=\frac{0.0053}{0.02186}=0.2424$$

$$P\{\text{Play}=\text{否}\mid X\}=\frac{P\{X\mid \text{Play}=\text{否}\}\cdot P\{\text{Play}=\text{否}\}}{P\{X\}}$$

$$=\frac{P\{\text{天气}=\text{晴}\mid \text{Play}=\text{否}\}\cdot P\{\text{温度}=\text{冷}\mid \text{Play}=\text{否}\}\cdot P\{\text{湿度}=\text{高}\mid \text{Play}=\text{否}\}\cdot P\{\text{风力}=\text{强}\mid \text{Play}=\text{否}\}\cdot P\{\text{Play}=\text{否}\}}{P\{\text{天气}=\text{晴}\}\cdot P\{\text{温度}=\text{冷}\}\cdot P\{\text{湿度}=\text{高}\}\cdot P\{\text{风力}=\text{强}\}}$$

$$=\frac{(3/5)\cdot(1/5)\cdot(4/5)\cdot(3/5)\cdot(5/14)}{(5/14)\cdot(4/14)\cdot(7/14)\cdot(6/14)}$$

$$=\frac{0.0206}{0.02186}=0.9421$$

- P{Play= 否 |X}>P{Play= 是 |X}, 故该天气条件下应该不去打网球。

5. 连续特征的处理

- 若特征是连续特征，需要先做处理。一种方法是对连续特征做离散化处理，变成离散特征；
- 另一种方法：假设连续特征具有正态分布，进而估计似然概率 $P(X_i|C=c_i)$:

$$\hat{P}(X_j|C=c_i)=\frac{1}{\sqrt{2\pi}\sigma_{ji}}\exp\left(-\frac{(X_j-\mu_{ji})^2}{2\sigma_{ji}^2}\right)$$

其中，μ_{ji}: 为 $C=c_i$ 类下特征 X_j 的均值；

σ_{ji}: 为 $C=c_i$ 类下特征 X_j 的标准差。

- 例如，Temperature 实际上是连续变量，比如取值为
  - Yes: $25.2, 19.3, 18.5, 21.7, 20.1, 24.3, 22.8, 23.1, 19.8$
  - No: $27.3, 30.1, 17.4, 29.5, 15.1$
- 先计算每一类的均值和标准差：
  - $\mu_{Yes} = 21.64, \quad \sigma_{Yes} = 2.35$
  - $\mu_{No} = 23.88, \quad \sigma_{No} = 7.09$
- 从而，样本 X 下若 `Temperature=`T_0，则将 $x = T_0$ 代入如下正态变换公式计算似然概率:

$$\hat{P}\{x|Yes\} = \frac{1}{2.35\sqrt{2\pi}} \exp\Big(-\frac{(x-21.64)^2}{2\times 2.35^2}\Big)$$

$$\hat{P}\{x|No\} = \frac{1}{7.09\sqrt{2\pi}} \exp\Big(-\frac{(x-23.88)^2}{2\times 7.09^2}\Big)$$

6. 修正 0 频率问题

- 若给定类别下，训练数据中某特征值没有出现，则基于频率的概率估计将是 0. 这是有问题的，因为它要与其它概率连乘，就都变成 0 了。
- 例如，收到一封 Email 包含一个在训练 Email 中从未出现过的单词，则 $P(X|y) = 0$, 对任意的 y;
- 因为一个事件还没有发生过，不代表它永远不会发生。朴素贝叶斯分类需要每个条件概率都非 0, 否则，预测概率将为 0.

- 解决办法：Laplace **修正**。
- 例如，数据集包含 1000 个样本，`income=low (0)`, `income=medium (990)`, `income=high (10)`. 用 Laplace 变换 (给每一类加 α, 称为 Laplace 平滑系数)：
  - $P(\text{income} = low) = \dfrac{\alpha}{1000 + 3\alpha}$
  - $P(\text{income} = medium) = \dfrac{990 + \alpha}{1000 + 3\alpha}$
  - $P(\text{income} = high) = \dfrac{10 + \alpha}{1000 + 3\alpha}$
- 修正的概率估计与原概率值很接近，又避免了出现条件概率为 0.

二. 朴素贝叶斯的案例

- 以来自 UCI 的 mushrooms 数据集为例，是关于毒蘑菇的数据，包含 8124 个观测，22 个变量。
- 因变量为 type，表示蘑菇是否有毒；其它为自变量，包括蘑菇的形状、标面光滑度、颜色、生长环境等。
- 本例将演示朴素贝叶斯分类的一般流程，顺便介绍一下**特征选择**。
- mlr3 中的朴素贝叶斯学习器用的 e1071 包中的 `naiveBayes()` 函数，可以接受连续型、离散型特征数据，其中参数 laplace 设置 Laplace 平滑系数 α.

1. 准备数据

```
library(mlr3verse)

dat = read_csv("datas/mushrooms.csv") %>%
  mutate(across(everything(), as.factor))

glimpse(dat)
#> Rows: 8,124
#> Columns: 22
#> $ type                  <fct> poisonous, edible, edible
#> $ cap_shape             <fct> convex, convex, bell, cor
#> $ cap_surface           <fct> smooth, smooth, smooth, s
#> $ cap_color             <fct> brown, yellow, white, whi
#> $ bruises               <fct> yes, yes, yes, yes, no, y
#> $ odor                  <fct> pungent, almond, anise, p
#> $ gill_attachment       <fct> free, free, free, free, f
```

```
summary(dat)  # 查看各变量各类别的水平及频数
#>         type          cap_shape     cap_surface     cap_colo
#>  edible   :4208   bell   : 452   fibrous:2320   brown  :22
#>  poisonous:3916   conical:   4   grooves:   4   gray   :18
#>                   convex :3656   scaly  :3244   red    :15
#>                   flat   :3152   smooth :2556   yellow :10
#>                   knobbed: 828                  white  :10
#>                   sunken :  32                  buff   : 1
#>                                                 (Other): 2
#>       odor      gill_attachment  gill_spacing   gill_size
#>  none   :3528   attached: 210   close  :6812   broad :5612
#>  foul   :2160   free    :7914   crowded:1312   narrow:2512
#>  fishy  : 576
#>  spicy  : 576
#>  almond : 400
#>  anise  : 400
#>  (Other): 484
```

2. 创建任务

```
task = as_task_classif(dat, target = "type")
task
#> <TaskClassif:dat> (8124 x 22)
#> * Target: type
#> * Properties: twoclass
#> * Features (21):
#>   - fct (21): bruises, cap_color, cap_shape, cap_surface,
#>     gill_attachment, gill_color, gill_size, gill_spacing,
#>     odor, population, ring_number, ring_type, spore_print_
#>     stalk_color_above_ring, stalk_color_below_ring, stalk_
#>     stalk_shape, stalk_surface_above_ring, stalk_surface_b
#>     veil_color
```

3. 特征选择

- 当数据集包含很多特征时，只提取最重要的部分特征来建模，称为**特征选择**。特征选择可以增强模型的解释性、加速学习过程、改进学习器性能。
- 常用的特征选择方法有两种：
  - **过滤法**：用 `mlr3filters` 包实现，基于某种衡量特征重要度的指标(如相关系数)，用外部算法计算变量的排名，只选用排名靠前的若干特征；
  - **包装法**：用 `mlr3fselect` 包实现，随机选择部分特征拟合模型并评估模型性能，通过交叉验证找到最佳的特征子集。

另外，有些学习器内部提供了选择有助于做预测的特征子集的方法，称为是**嵌入法**。

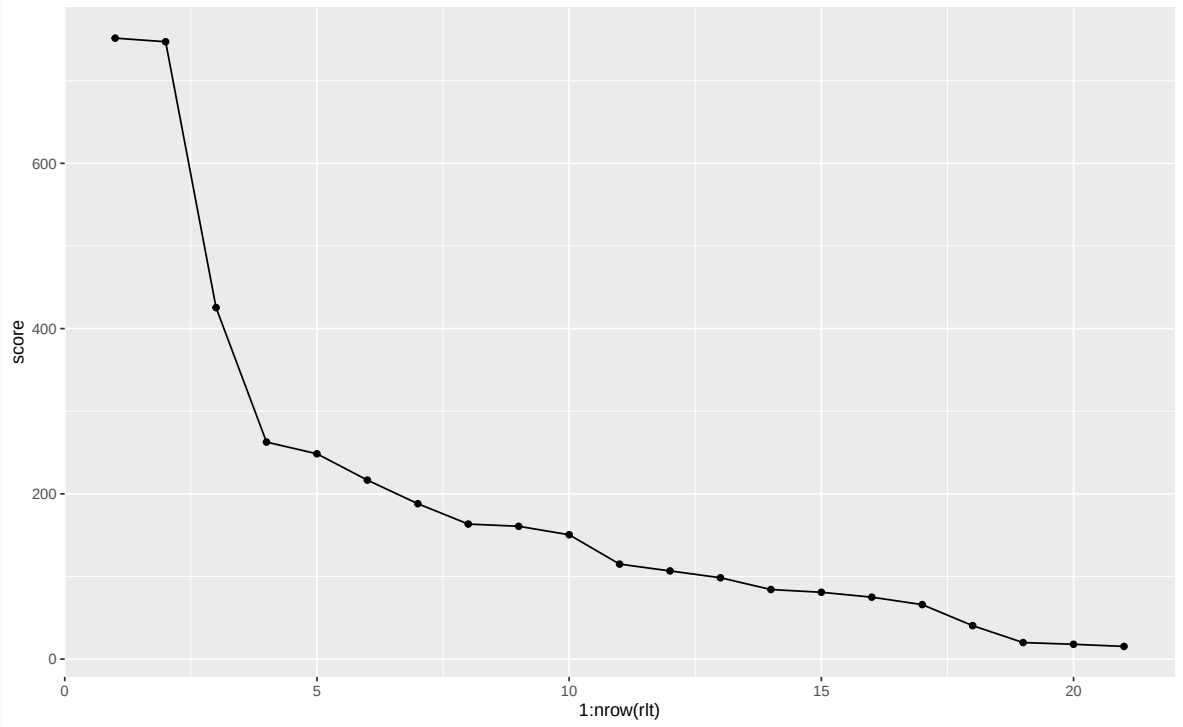
(1) 过滤法

- 采用基于随机森林学习器和特征重要度指标 `impurity` 的过滤法。

```
lrn = lrn("classif.ranger", importance = "impurity")
filter = flt("importance", learner = lrn)
filter$calculate(task)
```

```
rlt = as.data.table(filter)
rlt
#>                       feature score
#>  1:         spore_print_color 751.6
#>  2:                      odor 747.2
#>  3:                 gill_size 425.4
#>  4:                 ring_type 262.6
#>  5:                population 248.5
#>  6:                   bruises 216.6
#>  7:                stalk_root 188.1
#>  8:               gill_spacing 163.4
#>  9: stalk_surface_below_ring 160.7
#> 10: stalk_surface_above_ring 150.5
#> 11:                   habitat 115.0
#> 12:                 gill_color 106.7
#> 13:               stalk_shape  98.4
#> 14:               ring_number  84.2
```

```
rlt %>%
  ggplot(aes(x=1:nrow(rlt), y=score)) +
  geom_line() + geom_point()
```



- 根据随机森林不纯度指标，可选择前 3 个特征：
  `spore_print_color`, `odor`, `gill_size`.

```
library(praznik)
filter = flt("jmim")
filter
#> <FilterJMIM:jmim>
#> Task Types: classif, regr
#> Task Properties: -
#> Packages: mlr3filters, praznik
#> Feature types: integer, numeric, factor, ordered
```

```
filter$calculate(task)
as.data.table(filter)
#>                      feature score
#>  1:                      odor  1.00
#>  2:         spore_print_color  0.95
#>  3:                 gill_size  0.90
#>  4:                 ring_type  0.85
#>  5:                gill_color  0.80
#>  6: stalk_surface_below_ring  0.75
#>  7:                stalk_root  0.70
#>  8:                 cap_color  0.65
#>  9:   stalk_color_below_ring  0.60
#> 10: stalk_surface_above_ring  0.55
#> 11:   stalk_color_above_ring  0.50
#> 12:                population  0.45
#> 13:                   bruises  0.40
#> 14:                   habitat  0.35
```

- jmim 法选出三个特征：odor，spore_print_color，gill_size.
- 更多可选指标，参阅 mlr3book ch9.3. 有些指标，比如 correlation，还可以设置超参数。

(2) 包装法

```
library(mlr3fselect)
learner = lrn("classif.ranger", importance = "impurity")
instance = FSelectInstanceSingleCrit$new(
  task = task,
  learner = learner,
  resampling = rsmp("cv", folds = 3),
  measure = msr("classif.ce"),
  terminator = trm("none"),
  store_models = TRUE)
```

```
fselector = fs("rfe", recursive = TRUE)
fselector$optimize(instance)
#> INFO  [11:38:58.755] [bbotk] Starting to optimize 21 param
#> INFO  [11:38:58.791] [bbotk] Evaluating 1 configuration(s)
#> INFO  [11:38:58.995] [mlr3] Running benchmark with 3 resam
#> INFO  [11:38:59.036] [mlr3] Applying learner 'select.class
#> INFO  [11:38:59.665] [mlr3] Applying learner 'select.class
#> INFO  [11:39:00.274] [mlr3] Applying learner 'select.class
#> INFO  [11:39:00.906] [mlr3] Finished benchmark
#> INFO  [11:39:01.014] [bbotk] Result of batch 1:
#> INFO  [11:39:01.016] [bbotk]  bruises cap_color cap_shape
#> INFO  [11:39:01.016] [bbotk]    TRUE      TRUE      TRUE
#> INFO  [11:39:01.016] [bbotk]  gill_spacing habitat odor po
#> INFO  [11:39:01.016] [bbotk]         TRUE    TRUE TRUE
#> INFO  [11:39:01.016] [bbotk]  stalk_color_above_ring stalk
#> INFO  [11:39:01.016] [bbotk]                    TRUE
#> INFO  [11:39:01.016] [bbotk]  stalk_surface_above_ring sta
```

```
as.data.table(instance$archive)[1:3,1:22]
#>    bruises cap_color cap_shape cap_surface gill_attachment
#> 1:    TRUE      TRUE      TRUE         TRUE            TRUE
#> 2:    TRUE     FALSE     FALSE        FALSE           FALSE
#> 3:      NA        NA        NA           NA              NA
#>    gill_spacing habitat odor population ring_number ring_t
#> 1:         TRUE    TRUE TRUE       TRUE        TRUE      T
#> 2:         TRUE   FALSE TRUE       TRUE       FALSE      T
#> 3:           NA      NA   NA         NA          NA
#>    stalk_color_above_ring stalk_color_below_ring stalk_roo
#> 1:                   TRUE                   TRUE       TRU
#> 2:                  FALSE                  FALSE       TRU
#> 3:                     NA                     NA         N
#>    stalk_surface_above_ring stalk_surface_below_ring veil_
#> 1:                     TRUE                     TRUE
#> 2:                     TRUE                     TRUE
#> 3:                       NA                       NA
```

- 前面是用**递归特征消除法**筛选特征，结果是保留全部特征，预测错误率为 0. 第 2 个结果，保留了 10 个特征，预测错误率也为 0. 也可以选用第 3 个结果，保留了 5 个特征，预测错误率为 0.628%.
- 用 instance$result_feature_set 访问筛选出来的特征名字。除了递归特征消除，更多方法查阅 mlr3fselect 包文档。

修改任务的特征

```
task$feature_names    # 查看当前特征
#>  [1] "bruises"                   "cap_color"
#>  [3] "cap_shape"                 "cap_surface"
#>  [5] "gill_attachment"           "gill_color"
#>  [7] "gill_size"                 "gill_spacing"
#>  [9] "habitat"                   "odor"
#> [11] "population"                "ring_number"
#> [13] "ring_type"                 "spore_print_color"
#> [15] "stalk_color_above_ring"    "stalk_color_below_ring"
#> [17] "stalk_root"                "stalk_shape"
#> [19] "stalk_surface_above_ring" "stalk_surface_below_ring"
#> [21] "veil_color"
```

```r
# 选择最重要的 3 个特征
task$select(c("odor","spore_print_color", "gill_size"))
task
#> <TaskClassif:dat> (8124 x 4)
#> * Target: type
#> * Properties: twoclass
#> * Features (3):
#>   - fct (3): gill_size, odor, spore_print_color
```

4. 划分训练集测试集

- 做留出 (holdout) 重抽样，70% 作为训练集，其余 30% 作为测试集
- 为了保持训练集、测试集的因变量数据具有相似的分布，采用分层抽样方法
- 用 partition( ) 函数对任务做划分，默认按因变量分层，取出训练集索引和测试集索引

```
set.seed(123)
split = partition(task, ratio = 0.7)
# 默认 stratify = TRUE
```

5. 选择学习器

```
NB = lrn("classif.naive_bayes")   # 需要 e1071 包
NB$param_set
#> <ParamSet>
#>           id    class lower upper nlevels default value
#> 1:       eps ParamDbl  -Inf   Inf     Inf       0
#> 2:   laplace ParamDbl     0   Inf     Inf       0
#> 3: threshold ParamDbl  -Inf   Inf     Inf   0.001
```

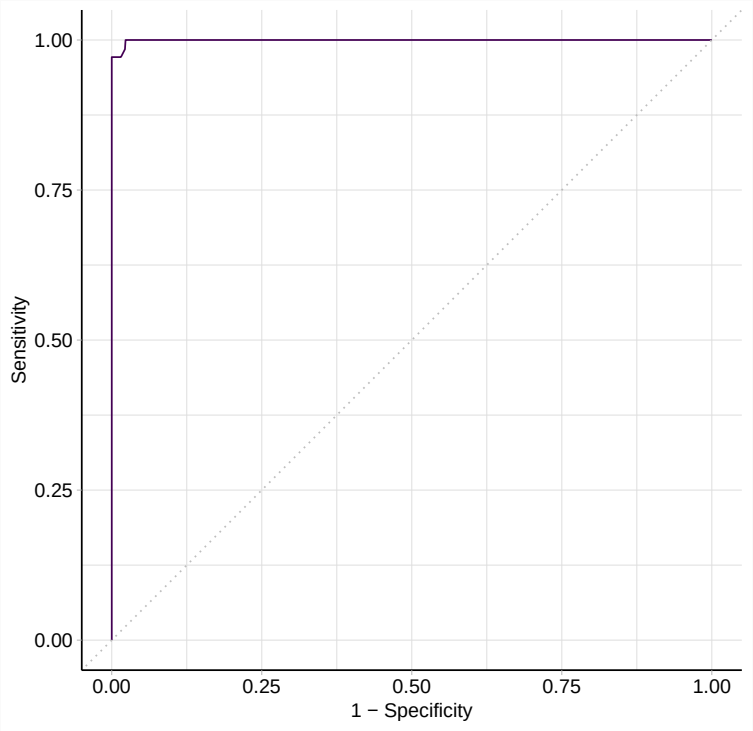
6. 模型训练与预测

```
NB$train(task, row_ids = split$train)
prediction = NB$predict(task, row_ids = split$test)
prediction
#> <PredictionClassif> for 2437 observations:
#>     row_ids     truth  response
#>           9 poisonous poisonous
#>          19 poisonous poisonous
#>          44 poisonous poisonous
#> ---
#>        8113    edible    edible
#>        8116    edible    edible
#>        8121    edible    edible
```

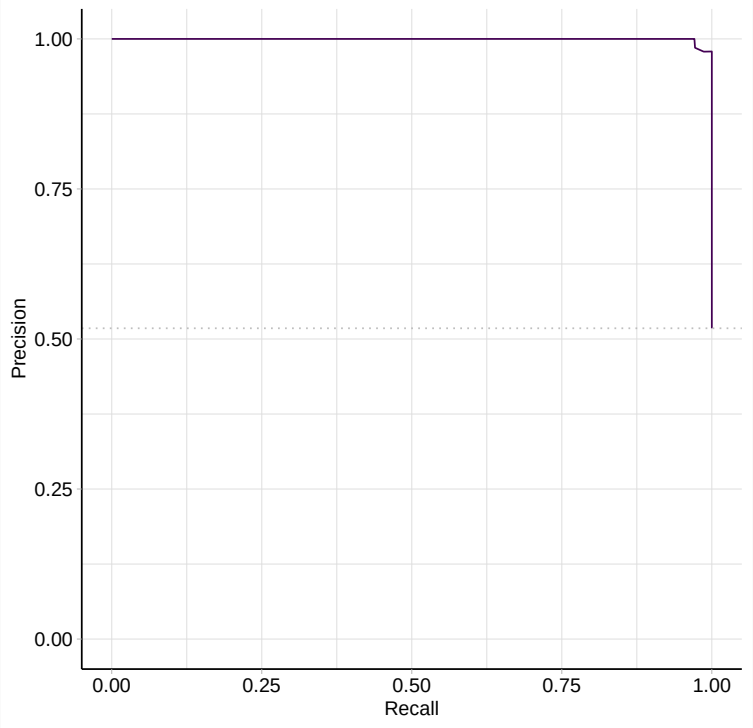
```
prediction$confusion                          # 混淆矩阵
#>             truth
#> response     edible poisonous
#>   edible       1226         18
#>   poisonous      36       1157
prediction$score(msr("classif.acc"))   # 准确率
#> classif.acc
#>       0.978
```

```r
NB$predict_type = "prob"
NB$train(task, row_ids = split$train)
prediction = NB$predict(task, row_ids = split$test)
prediction
#> <PredictionClassif> for 2437 observations:
#>     row_ids     truth  response prob.edible prob.poisonous
#>           9 poisonous poisonous      0.0131         0.98690
#>          19 poisonous poisonous      0.0137         0.98629
#>          44 poisonous poisonous      0.0137         0.98629
#> ---
#>        8113    edible    edible      0.9984         0.00162
#>        8116    edible    edible      0.9985         0.00153
#>        8121    edible    edible      0.9984         0.00162
```

```r
autoplot(prediction, type = "roc")      # ROC 曲线
prediction$score(msr("classif.auc"))    # AUC 面积
#> classif.auc
#>       0.999
```



```
autoplot(prediction, type = "prc")      # PR 曲线
```



主要参考文献

[1] mlr3book. 2020. https://mlr3book.mlr-org.com/

[2] Jim Liang(梁劲). Getting Started with Machine Learning, 2019.

[3] 刘顺祥. 从零开始学 Python: 数据分析与挖掘. 清华大学出版社, 2018.