第9讲 Naïve Bayes

- 概率基础回顾
- 极大似然估计
- Naïve Bayes

概率基础回顾

- 联合概率(**joint probability**)

  表示A事件和B事件同时发生的概率，$\boldsymbol{P(A \cap B)}$

- 边际概率(**marginal probability**)

  在A和B的样本空间中，只看A或B的概率，称之边际概率

- 条件概率(**conditional probability**)

  在发生A的条件下，发生B的概率，称为$\boldsymbol{P(B|A)}$

$$P(B \mid A)=\frac{P(A \cap B)}{P(A)}, P(A \mid B)=\frac{P(A \cap B)}{P(B)}$$

概率基础回顾

- 贝叶斯定理（Bayes theorem）

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

$$\Downarrow$$

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

贝叶斯定理是关于随机事件A和B的条件概率和边缘概率的一则定理

概率基础回顾

- **全概率(Total probability)**

假设$\{B_n$: n=1, 2, 3,...$\}$是一个概率空间的有限或可数无限的分割，且每个集合B_n是一个可测集合，则对任意事件A有全概率公式：

$$\Pr(A)=\sum_n \Pr(A\cap B_n)=\sum_n \Pr(A\mid B_n)\Pr(B_n)$$

概率基础回顾

- **乘法法则(Multiplicative rule)**

$$P(A \cap B) = P(B) \times P(A|B) = P(A) \times P(B|A)$$

- **独立事件(Independent events)**

设事件 A 和事件 B 满足以下条件：

$$P(A \cap B) = P(A) \times P(B)$$

或：$P(A) > 0,\ P(B|A) = P(B)$

$$P(B) > 0,\ P(A|B) = P(A)$$

则称 A 与 B 为独立事件

概率基础回顾

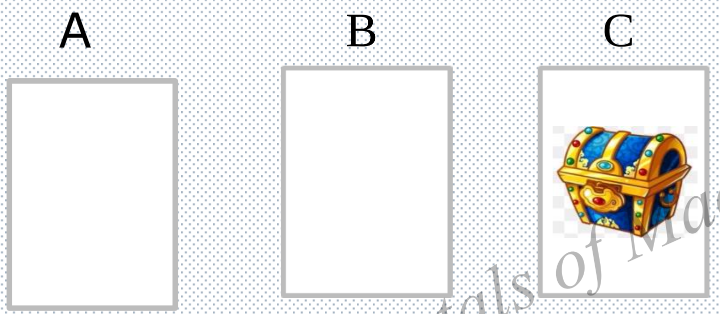
◆ 示例：概率计算

	赞成($B1$)	反对($B2$)	**合计**
男性($A1$)	40	120	160
女性($A2$)	10	30	40
合计	50	150	200

- 联合概率: **P(**男性，赞成**) = ?**
- 条件概率: **P(**赞成|男性**) = ?**
- 边际概率: **P(**赞成**) = P(B1) = ?**

概率基础回顾

◆ 示例--换门游戏



嘉宾选定一扇门未开启

↓

主持人开启一扇空门

↓

询问嘉宾是否换门？

↓

换门得宝？不换得宝？

换门得宝	**不换门得宝**
换门不得宝	不换门不得宝

极大似然估计

极大似然估计(MLE)

- 输入：模型(全部或者部分未知参数)和样本数据集
- 输出：模型的未知参数
- 处理：根据样本值选择模型参数，使得产生给定样本的可能性(概率)最大

极大似然估计

- **似然函数(likelihood function)**：一种关于统计模型参数的函数

离散型数据概率分布：假定一个关于参数θ、具有离散型概率分布P的随机变量X，则在给定X的输出x_i时，参数θ的似然函数：

$$L(\theta \mid X) = P(X \mid \theta) = P(X;\theta) = P_\theta(X) = \prod_{i=1}^{N} P_\theta(X = x_i),\ X = [x_1, x_2, \ldots, x_N]$$

连续型数据概率分布：假定一个关于参数θ、具有连续概率密度函数f的随机变量X，则在给定X的输出x_i时，参数θ的似然函数

$$L(\theta \mid X) = f_\theta(X) = \prod_{i=1}^{N} f_\theta(X = x_i),\ X = [x_1, x_2, \ldots, x_N]$$

MLE: $\hat{\theta} = \arg\max_{\theta} L(\theta \mid x)$

极大似然估计

MLE求解步骤

① 写出似然函数；

② 对似然函数取对数、整理；

③ 求导数；

④ 解似然方程

极大似然估计

◆ **示例**　设总体 $X \sim N(\mu,\sigma^2)$，样本 $(x_1,\cdots,x_n)$，其中 μ 和 σ^2 未知，试求 μ 和 σ^2 的最大似然估计量

$$L=\prod_{i=1}^{n}\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}=\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n e^{-\sum_{i=1}^{n}\frac{(x_i-\mu)^2}{2\sigma^2}},$$

$$\ln L=-\frac{n}{2}\ln(2\pi)-\frac{n}{2}\ln\sigma^2-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2,$$

令
$$\begin{cases}\dfrac{\partial \ln L}{\partial \mu}=\dfrac{1}{\sigma^2}\sum_{i=1}^{n}(x_i-\mu)=0\\[2ex]\dfrac{\partial \ln L}{\partial \sigma^2}=-\dfrac{n}{2\sigma^2}+\dfrac{1}{2(\sigma^2)^2}\sum_{i=1}^{n}(x_i-\mu)^2=0\end{cases}$$

极大似然估计

令
$$\begin{cases} \dfrac{\partial \ln L}{\partial \mu} = \dfrac{1}{\sigma^2} \sum (x_i - \mu) = 0 \\ \dfrac{\partial \ln L}{\partial \sigma^2} = -\dfrac{n}{2\sigma^2} + \dfrac{1}{2(\sigma^2)^2} \sum (x_i - \mu)^2 = 0 \end{cases}$$

解得
$$\hat{u} = \frac{1}{n}\sum_{i=1}^{n} x_i = \bar{X}, \quad \hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{X})^2$$

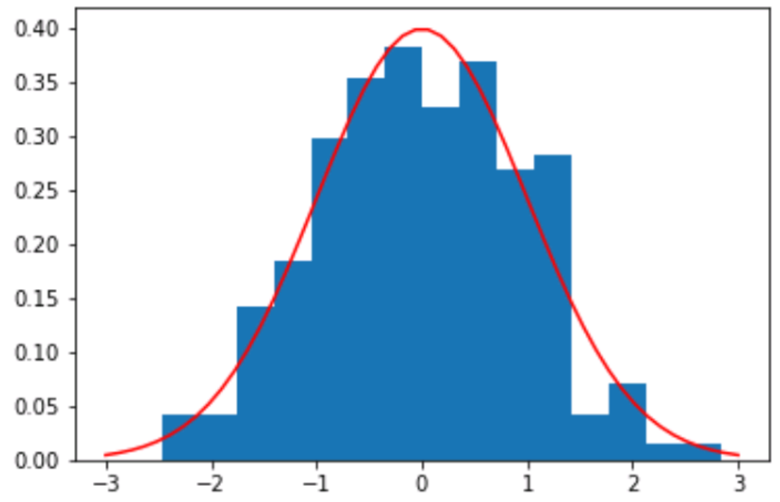
即为 μ 和 σ^2 的最大似然估计量.

极大似然估计

◆ 示例：MLE

```
from scipy.stats import norm
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np

x_norm = norm.rvs(size=200)
x_mean, x_std = norm.fit(x_norm)
print ('mean, ', x_mean)
print ('x_std, ', x_std)
plt.hist(x_norm, normed=True, bins=15)
x = np.linspace(-3, 3, 50)
plt.plot(x, norm.pdf(x), 'r-')
```

```
mean,  -0.009373023041395184
x_std,  0.9759251326915617

[<matplotlib.lines.Line2D at 0x1065c7b8>]
```



Naïve Bayes

- 朴素贝叶斯分类器(**Naïve Bayes classifier)**：
  每个数据样本包含N个特征：$\boldsymbol{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$
  属于m个不同类别之一：$\boldsymbol{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m\}$
  - **X** 属于类别的概率:

似然概率

$$h^*(x) = \arg\max_{c\in y} P(c\,|\,X) = \arg\max_{c\in y} \frac{P(X,c)}{P(X)} = \frac{P(X\,|\,c)P(c)}{P(X)}$$

  - 类条件独立假设(在给定类别情况下，所有特征相互独立)

$$P(X\,|\,c) = \prod_{i=1}^{N} P(x_i|c)$$

贝叶斯(Thomas Bayes，1702-1761)
英国数学家

Naïve Bayes

- 示例:

判断是否购买计算机

age	income	student	credit_rating	buys_computer
youth	high	no	fair	no
youth	high	no	excellent	no
middle_aged	high	no	fair	yes
senior	medium	no	fair	yes
senior	low	yes	fair	yes
senior	low	yes	excellent	no
middle_aged	low	yes	excellent	yes
youth	medium	no	fair	no
youth	low	yes	fair	yes
senior	medium	yes	fair	yes
youth	medium	yes	excellent	yes
middle_aged	medium	no	excellent	yes
middle_aged	high	yes	fair	yes
senior	medium	no	excellent	no

Naïve Bayes

- **示例：判断是否购买计算机**

X = (age=youth, income=medium, student=yes, credit_rating=fair)

buy computer?

- **P(buy=yes|X) > P(buy=no|X) ?**
- ***P(buy=yes|X) = P(X|buy=yes) * P(buy=yes) = 0.028***
- ***P(buy=no|X) = P(X|buy=no) * P(buy=no) = 0.007***

- P(buy=yes) = 9/14 = 0.643, P(buy=no) = 5/14 = 0.357
- P(age=youth|buy=yes) = 2/9 = 0.222
- P(X|buy=yes) = P(age=youth|buy=yes) * P(income=medium|buy=yes) * P(student=yes|buy=yes) * P(credit=fair|but=yes) = 0.044
- P(X|buy=no) = 0.019

Naïve Bayes

- 示例：假设在文本分类中，有3个类： c1、c2、c3，在指定的1000个训练样本中，某个词语x在各个类别中观测计数分别为0，990，10，则x的概率为

  0，0.99，0.01

- 零概率问题：在计算实例的概率时，如果某个量x，在训练集中没有出现过，会导致整个实例的概率结果为0。

Naïve Bayes

- 拉普拉斯平滑(Laplace Smoothing)：

$$P(x_i = x_{i*} \mid c) = \frac{N_{Ci*} + \lambda}{N_C + \alpha \times \lambda}$$

实际中，经常加 $\lambda(1 \geq \lambda \geq 0)$代替简单加1，如果对*N*个计数都加上λ，这时分母加上$a*\lambda$，a为类别数

当λ=1时，称作Laplace平滑，当0<λ<1时，称作Lidstone平滑，λ=0时，不做平滑

Naïve Bayes

- 示例：假设在文本分类中，有3个类：C1、C2、C3，在指定的1000个训练样本中，某个词语x在各个类别中观测计数分别为0，990，10，则x的概率为

  0，0.99，0.01

  拉普拉斯平滑：

  1/1003 = 0.001，991/1003=0.988，11/1003=0.011

Naïve Bayes

- 示例：根据如下训练数据集建模判断性别

性别	身高(英尺)	体重(磅)	脚的尺寸(英寸)
男	6	180	12
男	5.92	190	11
男	5.58	170	12
男	5.92	165	10
女	5	100	6
女	5.5	150	8
女	5.42	130	7
女	5.75	150	9

Naïve Bayes

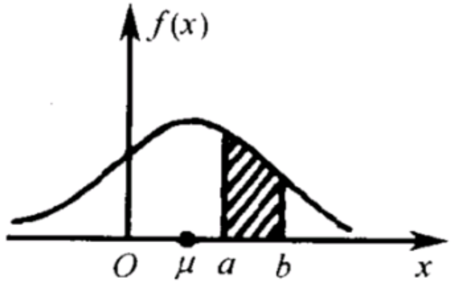
- 概率密度函数 (probability density function，PDF) $f(x)$应该满足下述两个条件：

$$f(x) \geq 0$$

$$\int_{-\infty}^{+\infty} f(x)dx = 1$$

随机变量X在a与b之间的概率

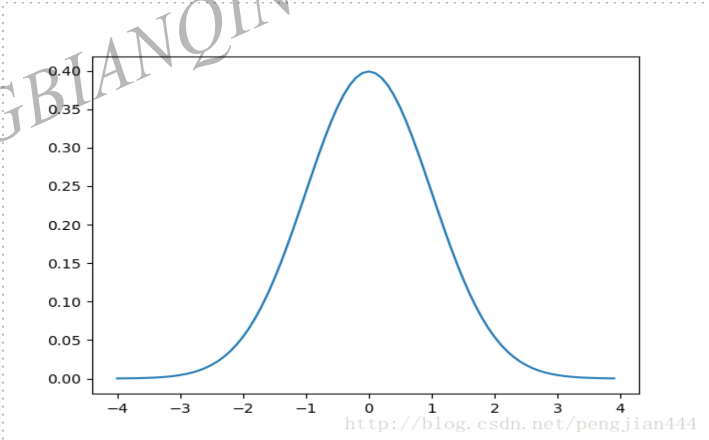
$$P(a < X < b = \int_{a}^{b} f(x)dx$$



(b)

Naïve Bayes

- 假定连续值特征的概率服从高斯分布（正态分布）



高斯分布的概率密度函数

$$f(x,\ \mu,\ \sigma)=\frac{1}{\sqrt{2\pi}\sigma}e^{\frac{-(x-\mu)^2}{2\sigma^2}}，\ -\infty<x<\infty$$

用来反映各特征值的相对可能性

$$P(X\,|\,c)=f(x,\ \mu,\ \sigma)$$

$$\hat{\mu}=\frac{1}{n}\sum_{i=1}^{n}x_i=\bar{X}\ ,$$

估算训练集中不同类别的不同特征值的均值、方差

$$\hat{\sigma}^2=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{X})^2$$

Naïve Bayes

- 示例：判断性别

性别	均值(身高)	方差(身高)	均值(体重)	方差(体重)	均值(脚的尺寸)	方差(脚的尺寸)
男性	5.855	3.5033e-02	176.25	1.2292e+02	11.25	9.1667e-01
女性	5.4175	9.7225e-02	132.5	5.5833e+02	7.5	1.6667e+00

Naïve Bayes

◆ 示例：判断性别

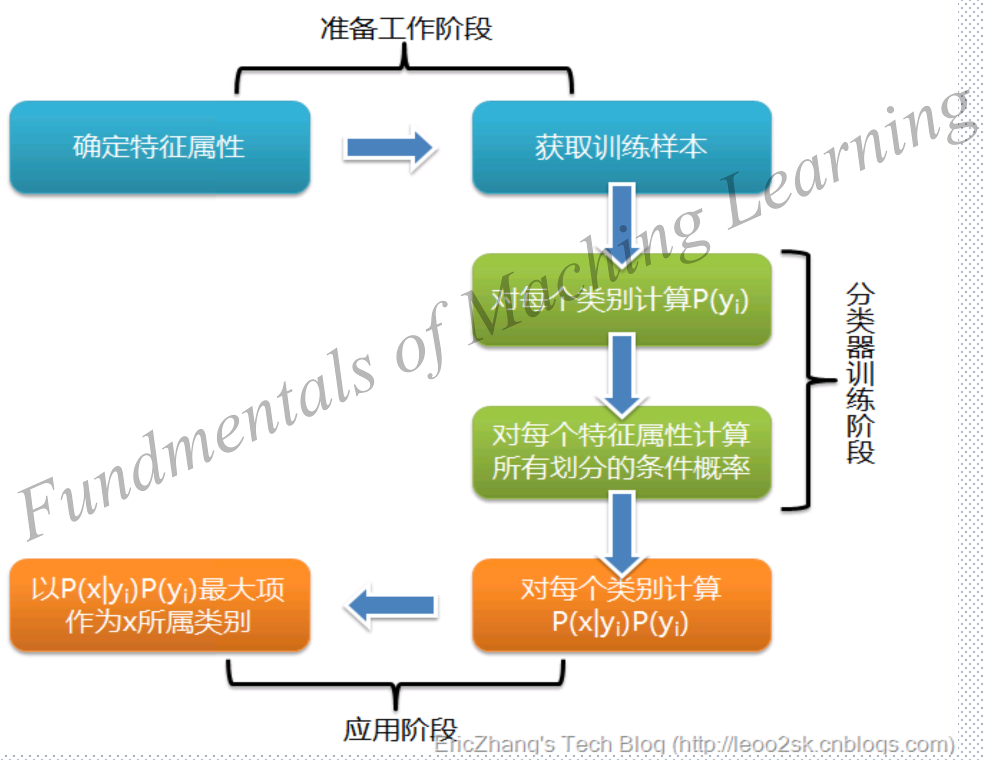
预测：某人身高6英尺，体重130磅，脚的尺寸为8英尺，
请推断某人是男性还是女性？

- $P(male) = P(female) = 0.5$
- $P(male|X) = P(male)p(height|male)p(weight|male)p(footsize|male)$
- $P(female|X) = P(female)p(height|female)p(weight|female)p(footsize|female)$

$\boldsymbol{P(male|X) = 6.1984e^{-09}}$
$\boldsymbol{P(female|X) = 5.3778e^{-04}}$

Naïve Bayes

Naïve Bayes流程



sklearn.naive_bayes

- **sklearn.naive_bayes**

  https://scikit-learn.org/stable/modules/classes.html#module-sklearn.naive_bayes

  naive_bayes.BernoulliNB

  naive_bayes.MultinomialNB

  naive_bayes.GaussianNB

sklearn.naive_bayes

- **BernoulliNB**：适合伯努利分布的数据集，即适合类特征为二元值的情况
  伯努利分布(Bernoulli distribution)：又称两点分布或0-1分布
  n次伯努利试验，每次试验的结果只有两种

$$f(x) = p^x(1-p)^{1-x} = \begin{cases} p & if\ \ x=1 \\ 1-p & if\ \ x=0 \end{cases}$$

二项分布(Binomial distribution)：
n 重伯努利试验成功次数的离散概率分布

$$P\{X=k\} = C_n^k p^k (1-p)^{n-k},\ k=0,1,2,\ldots,n$$

sklearn.naive_bayes

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

  __init__(*, alpha=1.0, binarize=0.0, fit_prior=True, class_prior=None

  - Parameters： alpha, fit_prior, class_prior
  - Attributes： class_count_, class_log_prior_, classes_, feature_count_, feature_log_prob_, n_features_
  - Methods： fit(X, y[, sample_weight]) , predict(X), predict_proba(X), predict_log_proba(X), score(X, y[, sample_weight])

sklearn.naive_bayes

◆ 示例：BernoulliNB

```
import numpy as np

# X包含4个特征：刮风，闷热，多云
X = np.array([[0, 1, 0],
              [1, 1, 1],
              [0, 1, 1],
              [0, 0, 0],
              [0, 1, 1],
              [0, 1, 0],
              [1, 0, 0]])
# y为样本标签，0代表没下雨，1代表下雨
y = np.array( [0, 1, 1, 0, 1, 0, 0])
```

```
# 构建BernoulliNB分类器
from sklearn.naive_bayes import BernoulliNB
clf = BernoulliNB()
clf.fit(X, y)

# 预测
Next_Day = [[0, 0, 1]]
pre = clf.predict(Next_Day)
if pre == [1]:
    print("下雨")
else:
    print("无雨")
```

```
clf.predict_proba(Next_Day)
```

```
array([[ 0.13848881,  0.86151119]])
```

sklearn.naive_bayes

- **MultinomialNB** ：适合类别特征多项式分布(**Multinomial Distribution**)的数据集，二项式分布的推广
  n次试验，每次结果有m（>2）个，且 m 个结果发生的概率互斥且之和为 1，则发生其中一个结果 X 的概率

  - 例如，扔骰子是典型的多项式分布
    若有6点的骰子重复扔n次
    试问有k次都是点数6朝上的概率

$$P\{X=k\}=C_n^k p_6^k (1-p_6)^{n-k},\ k=0,1,2,\ldots,n$$

sklearn.naive_bayes

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

  __init__(*, alpha=1.0, fit_prior=True, class_prior=None)

  - Parameters： alpha, fit_prior, class_prior
  - Attributes： class_count_, class_log_prior_, classes_, feature_count_, feature_log_prob_, n_features_ coef_, intercept_
  - Methods： fit(X, y[, sample_weight]) , predict(X), predict_proba(X), predict_log_proba(X), score(X, y[, sample_weight])

sklearn.naive_bayes

- **示例：MultinomialNB**

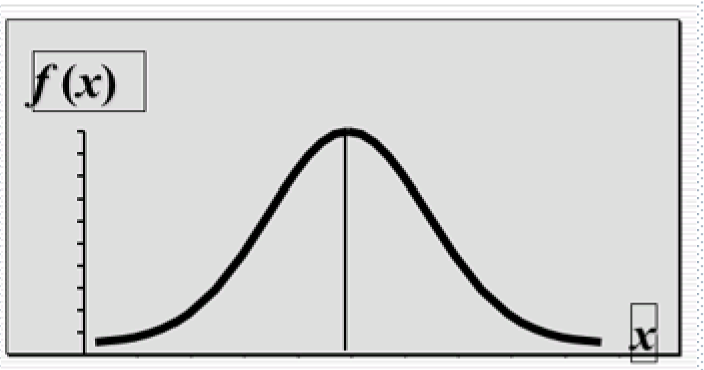
```
import numpy as np
X = np.random.randint(5, size=(6, 100))
y = np.array([1, 2, 3, 4, 5, 6])
from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB()
clf.fit(X, y)
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
print(clf.predict(X[2:3]))
```

```
[3]
```

sklearn.naive_bayes

- **GaussianNB**：适合高斯分布的数据集，即适合连续特征值
  - 高斯分布：也称正态分布(**normal distribution**)
  - 连续值特征，概率密度函数：

$$f(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2\sigma^2}(x-\mu)^2}, \quad -\infty<x<+\infty$$



sklearn.naive_bayes

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

  - Parameters： prior(), var_smoothing
  - Attributes： class_count_, classes_,
    epsilon_, sigma_, theta_
  - Methods： fit(X, y[, sample_weight]) , predict(X),
    predict_proba(X), predict_log_proba(X),
    score(X, y[, sample_weight])

sklearn.naive_bayes

◆ 示例：GaussianNB

```python
# 导入数据集生成工具
from sklearn.datasets import make_blobs
# 生成500个样本数据, 5个类别
X, y = make_blobs(n_samples=500, centers=5,random_
```

```python
print(X)
```

```
[[ -4.43344765e+00  -9.14511574e+00]
 [ -5.06998128e+00  -9.75464122e+00]
 [  6.54464509e+00   8.99873511e-01]
 [  3.25023324e-01   1.50633915e-01]
 [ -1.51028157e+00  -1.10581275e+00]
 [ -8.90489310e+00  -1.10427432e+01]
 [  9.28383472e-02  -2.00771121e-02]
 [ -6.21720086e+00  -1.11227678e+01]
 [  7.63027116e+00   8.69797933e+00]
 [  7.92430026e+00   1.04511206e-01]
```

```python
print(y)
```

```
[2 2 1 4 3 2 4 2 0 1 2 0 2 0 0 2 3 0 2 3 0 4 1 3 4 1 4 4 2 3 3 1 0 0 4 1 2
 0 2 4 4 3 1 0 3 0 0 3 3 2 2 4 4 4 2 0 3 1 0 0 4 0 4 3 2 0 0 3 2 0 0 0 3 2
 1 1 3 2 0 3 1 3 1 3 3 2 3 3 1 4 0 2 4 3 3 4 1 2 4 2 3 2 2 1 3 1 2 0 0 4 1
 4 4 0 2 1 0 4 0 1 0 0 0 3 0 2 0 3 1 3 0 1 2 0 3 4 1 1 2 1 2 1 2 2 3 2 0 4
 1 0 2 0 1 1 4 1 2 2 2 4 3 3 0 2 2 4 4 4 0 1 4 1 0 0 2 0 3 4 3 0 0 1 3 4 4
 0 2 4 4 3 0 1 1 1 4 2 0 1 4 2 2 1 3 1 3 4 3 3 2 3 3 2 3 4 3 3 1 1 3 0 4 4
 4 1 3 0 4 4 0 2 4 1 0 2 4 0 1 3 4 0 4 1 3 3 1 1 3 3 1 4 3 2 3 0 2 2 1 1 1
 2 2 0 1 2 4 3 0 2 3 0 2 0 4 4 1 0 0 4 1 1 1 4 3 0 1 2 1 1 0 3 3 2 2 1 4 3
 4 4 2 0 1 4 3 0 3 0 2 3 2 1 4 0 1 3 2 2 3 2 1 2 1 1 4 4 1 3 4 2 4 4 0 1 3
 0 1 1 2 1 3 4 0 2 2 3 1 4 2 3 1 1 4 0 3 4 0 2 3 2 0 1 1 3 3 0 2 2 1 2 0 0
 2 4 0 1 2 2 1 0 4 3 1 1 2 2 1 1 2 1 0 0 3 2 2 4 4 2 2 3 1 0 4 0 4 3 3 0 3
 1 0 1 4 4 4 1 3 0 2 2 4 0 3 2 2 1 4 4 4 0 4 3 2 4 3 3 1 3 0 0 2 0 0 3 3 2
 2 3 1 4 0 4 3 2 4 0 3 3 3 0 0 4 3 1 1 4 3 3 3 2 2 1 4 2 4 0 1 0 4 4 0 4 4
 3 3 4 2 4 1 4 0 1 0 1 4 4 4 1 2 1 1 3]
```

sklearn.naive_bayes

◆ 示例：GaussianNB

```
# 导入数据集生成工具
from sklearn.datasets import make_blobs
# 生成500个样本数据, 5个类别
X, y = make_blobs(n_samples=500, centers=5,random_state=8)

# 导入数据集拆分工具
from sklearn.model_selection import train_test_split

# 数据拆分成训练集和测试集
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=8)

# 构建GaussianNB分类器
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X_train, y_train)
print('模型得分: {:.3f}'.format(gnb.score(X_test, y_test)))
```

```
模型得分: 0.968
```

sklearn.naive_bayes

- sklearn的分类模型可信度评估
  - .predict()直接获得唯一的预测结果
  - .predict_proba()获取属于类别的概率
  - .predict_log_proba()获取属于类别的概率的对数

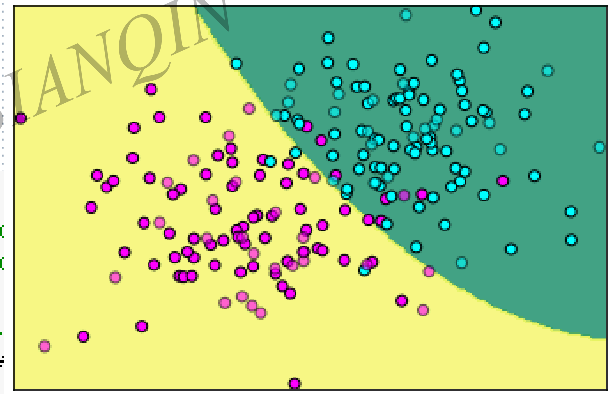
sklearn.naive_bayes

- 示例： .predict()直接获得唯一预测结果

```
# predict

# 导入GaussianNB模型
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
# 拆分数据集
from sklearn.model_selection import train_test_sp
X_train, X_test, y_train, y_test = train_test_spli
# 训练模型
gnb.fit(X_train, y_train)
# 获取模型预测值
predict = gnb.predict(X_test)
print('预测值形态: {}'.format(predict.shape))
print(predict[:5])
```

```
预测值形态: (50,)
[0 1 1 0 1]
```

```
# 坐标轴范围
x_min, x_max = X[:,0].min()-0.5, X[:,0].max()+
y_min, y_max = X[:,1].min()-0.5, X[:,1].max()+

xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.
                     np.arange(y_min, y_max
# 用不同色彩表示不同分类
Z = gnb.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

# 绘制等高线
plt.contourf(xx, yy, Z, cmap = plt.cm.summer, alpha=.8)
# 绘制散点图
plt.scatter(X_train[:, 0], X_train[:, 1], c=y_train, cmap=plt.cm.cool,edgecolor='k')
plt.scatter(X_test[:, 0], X_test[:, 1], c=y_test, cmap=plt.cm.cool,edgecolor='k', alpha=0.6)
# 坐标轴范围
plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
# 坐标轴单位
plt.xticks(())
plt.yticks(())
```



sklearn.naive_bayes

- **sklearn的一些分类模型中的预测概率**

.predict_proba()

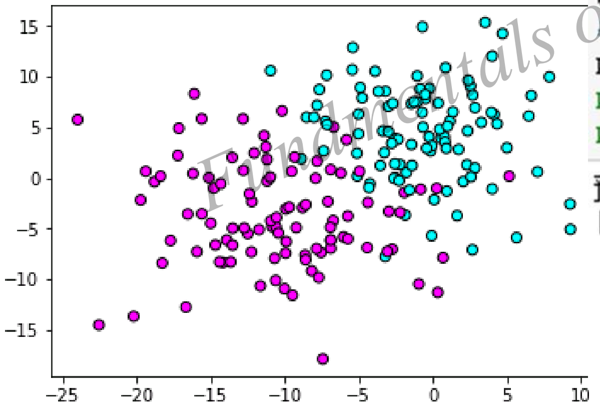
- 计算分类时每个样本属于不同类别概率
- 对于二元分类，它的形状是(**n_smples**, **2**)
- 对于多元分类，它的形状是(**n_smples**, **n_classes**)

sklearn.naive_bayes

◆ 示例：predict_proba()方法

```
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
# 导入数据集生成工具
from sklearn.datasets import make_blobs
X, y = make_blobs(n_samples=200, random_state=1,ce
plt.scatter(X[:,0],X[:,1],c=y, cmap=plt.cm.cool, e
```

<matplotlib.collections.PathCollection at 0x1091a9

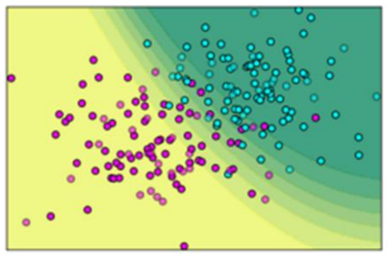



```
# 导入GaussianNB模型
from sklearn.naive_bayes impor
gnb = GaussianNB()
# 拆分数据集
from sklearn.model_selection im
X_train, X_test, y_train, y_tes
# 训练模型
gnb.fit(X_train, y_train)
# 获取模型预测概率
predict_proba = gnb.predict_prol
print('预测准确率形态: {}'.forma
print(predict_proba[:5])
```

```
预测准确率形态: (50, 2)
[[0.98849996 0.01150004]
 [0.0495985  0.9504015 ]
 [0.01648034 0.98351966]
 [0.8168274  0.1831726 ]
 [0.00282471 0.99717529]]
```

```
# predict_proba可视化
# 坐标轴范围
x_min, x_max = X[:,0].min()-0.5
y_min, y_max = X[:,1].min()-0.5
xx, yy = np.meshgrid(np.arange(
                     np.ara
Z = gnb.predict_proba(np.c_[xx.
Z = Z.reshape(xx.shape)
# 绘制等高线
plt.contourf(xx, yy, Z, cmap = plt.cm.summer, alpha=.8)
# 绘制散点图
plt.scatter(X_train[:, 0], X_train[:, 1], c=y_train, cmap=plt.cm.cool,
            edgecolor='k')
plt.scatter(X_test[:, 0], X_test[:, 1], c=y_test, cmap=plt.cm.cool,
            edgecolor='k', alpha=0.6)
# 坐标轴范围
plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
# 坐标轴单位
plt.xticks(())
plt.yticks(())
```

([], <a list of 0 Text yticklabel objects>)



Bayes Classifier

- Naïve Bayes分类假定类条件独立，即给定样本的类标号，属性的值相互条件独立，也因此称为“朴素”。
- 实际中变量之间可能存在依赖。贝叶斯信念网络描述联合条件概率分布，它允许在变量的子集间定义类条件独立性，它提供一种因果关系的图形。
- 贝叶斯信念网络可用于分类，它是图形模型。相比Naïve Bayes，它能够处理属性子集间有依赖关系的分类。