Bayes Classifier and Naive Bayes

Kun He (何琨)

Data Mining and Machine Learning Lab
(John Hopcroft Lab)
Huazhong University of Science & Technology

brooklet60@hust.edu.cn

2022年5月

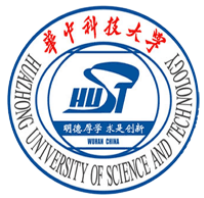


Table of Contents

Table of Contents

Basic idea

In machine learning, the Naive Bayes classifier is a series of simple probability classifiers based on the Bayesian theorem under strong independent assumptions.

- Training Data:$D = \{(\mathbf{x}_1,y_1),...,(\mathbf{x}_n,y_n)\}$,$(\mathbf{x}_i,y_i)$ is sampled i.i.d from unknown distribution $P(X, Y)$.So we obtain:

$$P(D) = P((\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)) = \prod_{\alpha=1}^{n} P(\mathbf{x}_\alpha, y_\alpha).$$

- If we do have enough data, we could estimate $P(X, Y)$ similar to the coin example in the previous lecture, where we imagine a gigantic die that has one side for each possible value of (x, y). We can estimate the probability that one specific side comes up through counting.
- Estimate $P(X, Y)$:

$$\hat{P}(\mathbf{x}, y) = \frac{\sum_{i=1}^{n} I(\mathbf{x}_i = x \wedge y_i = y)}{n}.$$

$$I(\mathbf{x}_i = x \wedge y_i = y) = 1 \quad if \quad \mathbf{x}_i = x \quad and \quad y_i = y,$$

and 0 otherwise.

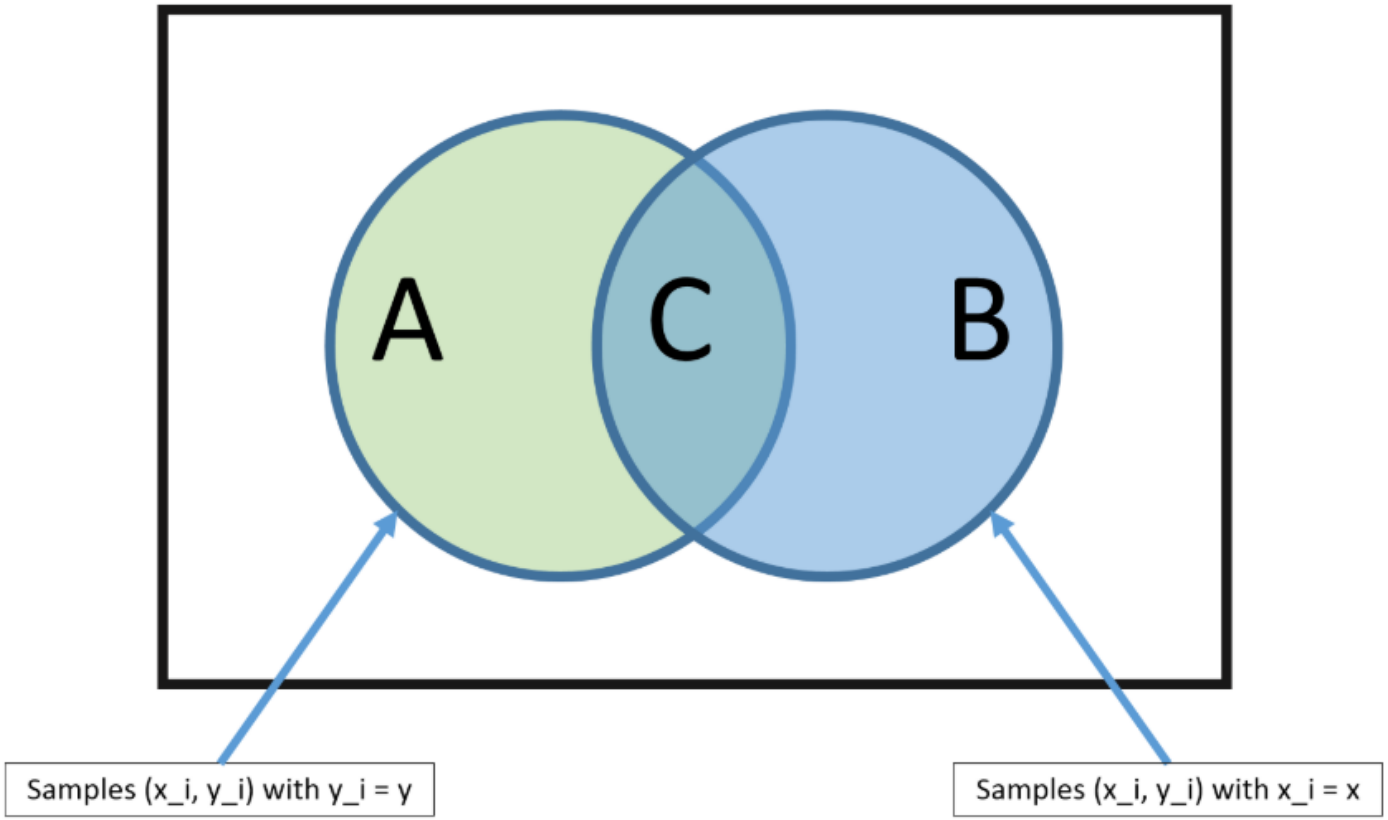
Basic idea

- Of course, if we are primarily interested in predicting the label y from the features x, we may estimate $P(Y|X)$ directly instead of $P(X, Y)$. We can then use the Bayes Optimal Classifier for a specific $\hat{P}(y|x)$ to make predictions.
- We can then use the Bayes Optimal Classifier for a specific $\hat{P}(y|x)$ to make predictions.

$$\hat{P}(y|\mathbf{x}) = \frac{\hat{P}(y, \mathbf{x})}{P(\mathbf{x})} = \frac{[\sum_{i=1}^{n} I(\mathbf{x}_i = \mathbf{x} \wedge y_i = y)]/n}{[\sum_{i=1}^{n} I(\mathbf{x}_i = \mathbf{x})]/n}.$$

$$= \frac{\sum_{i=1}^{n} I(\mathbf{x}_i = \mathbf{x} \wedge y_i = y)}{\sum_{i=1}^{n} I(\mathbf{x}_i = \mathbf{x})}$$

Visualization



Venn diagram

- The Venn diagram illustrates that the MLE method estimates:

$$\hat{P}(y|\mathbf{x}) = \frac{|C|}{|B|}.$$

Table of Contents

Bayes rule

If we can estimate $P(y)$ and $P(\mathbf{x} \mid y)$,since, by Bayes rule,

$$P(y|\mathbf{x}) = \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})}.$$

Estimating $P(y)$, $P(x|y)$

- Estimating $P(y)$ is easy: For example, if Y takes on discrete binary values estimating $P(Y)$ reduces to coin tossing. We simply need to count how many times we observe each outcome (in this case each class):
$$P(y = c) = \frac{\sum_{i=1}^{n} I(y_i = c)}{n} = \hat{\pi}_c$$
- Estimating $P(x|y)$, however, is **not easy**! E.g. Spam checker.
- The additional assumption that we make is the Naive Bayes assumption.

Naive Bayes Assumption:

$$P(\mathbf{x}|y) = \prod_{\alpha=1}^{d} P(x_\alpha|y), \text{where } x_\alpha = [\mathbf{x}]_\alpha \text{ is the value for feature } \alpha.$$

i.e.,feature values are **independent given the label!** This is a very **bold** assumption.

For example, a setting where the Naive Bayes classifier is often used is spam filtering. Here, the data is emails and the label is spam or not-spam. The Naive Bayes assumption implies that the words in an email are conditionally independent, given that you know that an email is spam or not. Clearly this is not true. Neither the words of spam or not-spam emails are drawn independently at random. However, the resulting classifiers can work well in practice even if this assumption is violated.

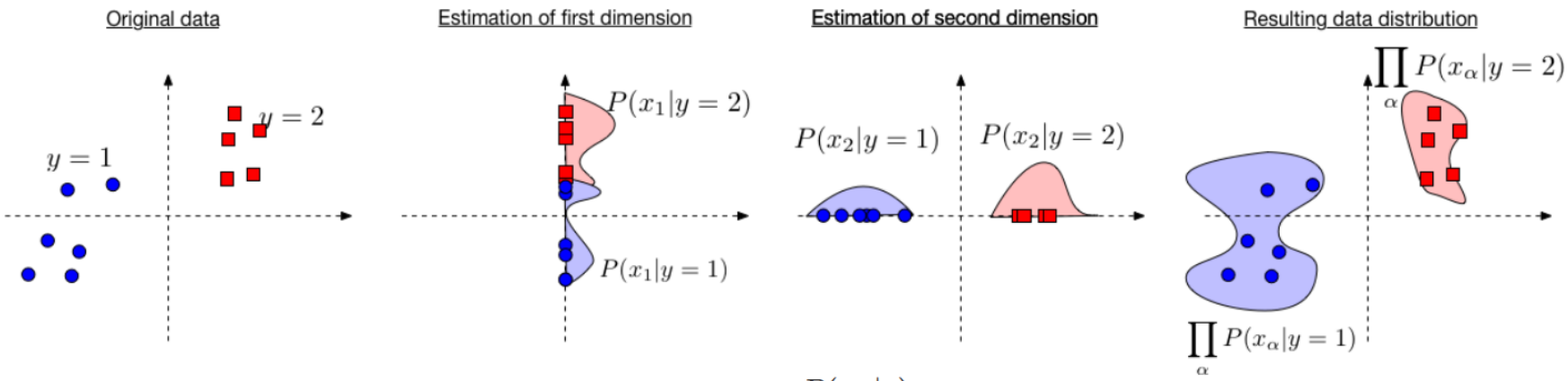


Illustration behind the Naive Bayes algorithm. We estimate $P(x_\alpha|y)$ independently in each dimension (middle two images) and then obtain an estimate of the full data distribution by assuming conditional independence $P(\mathbf{x}|y) = \prod_\alpha P(x_\alpha|y)$ (very right image).

Estimating $P(\mathbf{x} \mid y)$

So, for now, let's pretend the Naive Bayes assumption holds. Then the Bayes Classifier can be defined as follows.

Bayes Classifier

Because of the Naive Bayes assumption

$$h(\mathbf{x}) = \underset{y}{\operatorname{argmax}} P(y|\mathbf{x}) \tag{1}$$

$$= \underset{y}{\operatorname{argmax}} \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})} \tag{2}$$

$$= \underset{y}{\operatorname{argmax}} P(\mathbf{x}|y)P(y) \qquad (P(\mathbf{x}) \text{ does not depend on } y) \tag{3}$$

$$= \underset{y}{\operatorname{argmax}} \prod_{\alpha=1}^{d} P(x_\alpha|y)P(y) \qquad (\text{by the naive Bayes assumption}) \tag{4}$$

$$= \underset{y}{\operatorname{argmax}} \sum_{\alpha=1}^{d} \log(P(x_\alpha|y)) + \log(P(y)) \qquad (\text{as log is a monotonic function}) \tag{5}$$

Estimating $log(P(x_\alpha|y))$ is easy as we only need to consider one dimension. And estimating $P(y)$ is not affected by the assumption.

Table of Contents

Case # 1: Categorical features

Features:

$$[\mathbf{x}]_\alpha \in \{f_1, f_2, \cdots, f_{K_\alpha}\}.$$

不同的$[\mathbf{x}]_\alpha$，如性别，年龄，省份；

Each feature α falls into one of K_α categories. (Note that the case with binary features is just a specific case of this, where $K_\alpha = 2$.) An example of such a setting may be medical data where one feature could be gender (male / female) or marital status (single / married).

Illustration of categorical NB. For d dimensional data, there exist d independent dice for each class. Each feature has one die per class. We assume training samples were generated by rolling one die after another. The value in dimension i corresponds to the outcome that was rolled with the i^{th} die.

Categorical features

不同的$[\mathbf{x}]_\alpha$, 如性别，年龄，省份; $y = c$, 如健康状态

Model $P(x_\alpha \mid y)$:

$$P(x_\alpha = j|y = c) = [\theta_{jc}]_\alpha \text{ and } \sum_{j=1}^{K_\alpha}[\theta_{jc}]_\alpha = 1.$$

$[\theta_{jc}]_\alpha$ is the probability of feature α having the value j, given that the label is c. And the constraint indicates that x_α must have one of the categories $\{1, \ldots, K_\alpha\}$.

Parameter Estimation:

$$[\hat{\theta}_{jc}]_\alpha = \frac{\sum_{i=1}^{n} I(y_i = c)I(x_{i\alpha} = j) + \ell}{\sum_{i=1}^{n} I(y_i = c) + \ell K_\alpha}, \tag{6}$$

$$x_{i\alpha} = [\mathbf{x}_i]_\alpha,$$

I is a smoothing parameter.

By setting ℓ=0 we get an MLE estimator, $\ell > 0$ leads to MAP.

If we set ℓ=+1 we get Laplace smoothing.

In words,this means:

$$\frac{\text{number of samples with label c that have feature } \alpha \text{ with value } j}{\text{number of samples with label } c}.$$

例： c为患冠心病，α为年龄段， j为60至69

Prediction

Essentially the categorical feature model associate a special coin with each feature and label. The generative model that we are assuming is that the data was generated by first choosing the label (e.g. "healthy person"). That label comes with a set of d "dice", for each dimension one. The generator picks each die, tosses it and fills in the feature value with the outcome of the coin toss. So if there are C possible labels and d dimensions we are estimating $d \times C$ "dice" from the data. However, per data point only d dice are tossed (one for each dimension). Die α (for any label) has K_α possible "sides". Of course this is not how the data is generated in reality - but it is a modeling assumption that we make. We then learn these models from the data and during test time see which model is more likely given the sample.

Prediction

$$h(\mathbf{x}) = \underset{y}{\text{argmax}} \prod_{\alpha=1}^{d} P(x_\alpha|y)P(y)$$

$$\underset{y}{\text{argmax}}\ P(y = c \mid \mathbf{x}) \propto \underset{y}{\text{argmax}}\ \hat{\pi}_c \prod_{\alpha=1}^{d} [\hat{\theta}_{jc}]_\alpha$$

$$\hat{\pi}_c = P(y = c) = \frac{\sum_{i=1}^{n} I(y_i = c)}{n}$$

Case # 2: Multinomial features

Multinomial features

If feature values don't represent categories (e.g. male/female) but counts, we need to use a different model. E.g. in the text document categorization, feature value $x_\alpha = j$ means that in this particular document **x** the α^{th} word in my dictionary appears j times.
Let us consider the example of spam filtering. Imagine the α^{th} word is indicative towards *spam*.Then if $x_\alpha = 10$ means that this email is likely spam(as word α appears 10 times in it). And another email with $x'_\alpha = 20$ should be even more likely to be spam (as the spammy word appears twice as often). With categorical features this is not guaranteed.

Features: 第α种单词出现的数量:

$$x_\alpha \in \{0, 1, 2, \ldots, m\} \text{ and } m = \sum_{\alpha=1}^{d} x_\alpha \tag{7}$$

Each feature α represents a count and m is the length of the sequence. An example of this could be the count of a specific word α in a document of length m and d is the size of the vocabulary.

Model P(**x**|y)

Use the multinomial distribution:

$$P(\mathbf{x} \mid m, y = c) = \frac{m!}{x_1! \cdot x_2! \cdot \cdots \cdot x_d!} \prod_{\alpha=1}^{d} (\theta_{\alpha c})^{x_\alpha}$$

where $\theta_{\alpha c}$ is the probability of selecting x_α and $\sum_{\alpha=1}^{d} \theta_{\alpha c} = 1$.
So, we can use this to generate a spam email, i.e., a document $\mathbf{x}$ of class $y =$ spam by picking m words independently at random from the vocabulary of d words using $P(\mathbf{x} \mid y =$ spam$)$.
Parameter estimation:

$$\hat{\theta}_{\alpha c} = \frac{\sum_{i=1}^{n} I(y_i = c) x_{i\alpha} + \ell}{\sum_{i=1}^{n} I(y_i = c) m_i + \ell \cdot d} \tag{8}$$

where $m_i = \sum_{\beta=1}^{d} x_{i\beta}$ denotes the number of words in document i.The numerator sums up all counts for feature x_α and the denominator sums up all counts of all features across all data points.
In words:

$$\frac{\text{number of times word } \alpha \text{ appears in all spam emails}}{\text{number of words in all spam emails combined}}.$$

Prediction:

$$\underset{c}{\text{argmax}}\ P(y = c \mid \mathbf{x}) \propto \underset{c}{\text{argmax}}\ \hat{\pi}_c \prod_{\alpha=1}^{d} \hat{\theta}_{\alpha c}^{x_\alpha}$$

Case #3: Continuous features (Gaussian Naive Bayes)

Features: 如身高，血压，

$$x_\alpha \in \mathbb{R} \qquad \text{(each feature takes on a real value)} \tag{9}$$

Model $P(x_\alpha \mid y)$ Use Gaussian distribution:

$$P(x_\alpha \mid y = c) = \mathcal{N}\left(\mu_{\alpha c}, \sigma^2_{\alpha c}\right) = \frac{1}{\sqrt{2\pi}\sigma_{\alpha c}} e^{-\frac{1}{2}\left(\frac{x_\alpha - \mu_{\alpha c}}{\sigma_{\alpha c}}\right)^2} \tag{10}$$

Note that the model specified above is based on our assumption about the data - that each feature α comes from a class-conditional Gaussian distribution. The full distribution:

$$P(\mathbf{x}|y) \sim \mathcal{N}(\mu_y, \Sigma_y)$$

where Σ_y is a diagonal covariance matrix with

$$[\Sigma_y]_{\alpha,\alpha} = \sigma^2_{\alpha,y}$$

Parameter estimation:

As always, we estimate the parameters of the distributions for each dimension and class independently. Gaussian distributions only have two parameters, the mean and variance.

The mean $\mu_{\alpha,y}$ is estimated by the average feature value of dimension α from all samples with label y.

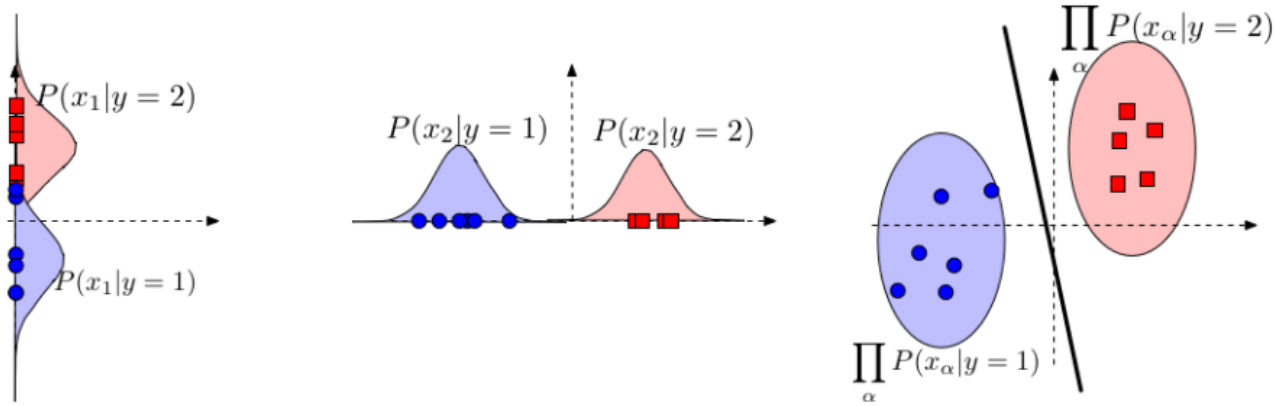
The (squared) standard deviation is simply the variance of this estimate.

$$\mu_{\alpha c} \leftarrow \frac{1}{n_c} \sum_{i=1}^{n} I(y_i = c) x_{i\alpha} \qquad \text{where } n_c = \sum_{i=1}^{n} I(y_i = c) \tag{11}$$

$$\sigma^2_{\alpha c} \leftarrow \frac{1}{n_c} \sum_{i=1}^{n} I(y_i = c)(x_{i\alpha} - \mu_{\alpha c})^2 \tag{12}$$

Table of Contents

Naive Bayes is a Linear Classifier



Naive Bayes leads to a linear decision boundary in many common cases. Illustrated here is the case where $P(x_\alpha|y)$ is Gaussian and where $\sigma_{\alpha,c}$ is identical for all c (but can differ across dimensions α). The boundary of the ellipsoids indicate regions of equal probabilities $P(\mathbf{x}|y)$. The red decision line indicates the decision boundary where $P(y=1|\mathbf{x}) = P(y=2|\mathbf{x})$.

1. Multinomial Features

Suppose that $y_i \in \{-1, +1\}$ and features are multinomial. So:

$$h(\mathbf{x}) = \underset{y}{\text{argmax}} \; P(y) \prod_{\alpha-1}^{d} P(x_\alpha \mid y) = \text{sign}(\mathbf{w}^\top \mathbf{x} + b)$$

$$\mathbf{w}^\top \mathbf{x} + b > 0 \Longleftrightarrow h(\mathbf{x}) = +1.$$

As before, we define:

$$P(x_\alpha | y = +1) \propto \theta_{\alpha+}^{x_\alpha}; P(Y = +1) = \pi_+.$$

$$[\mathbf{w}]_\alpha = \log(\theta_{\alpha+}) - \log(\theta_{\alpha-}) \tag{13}$$

$$b = \log(\pi_+) - \log(\pi_-) \tag{14}$$

If we use the above to do classification, we can compute for $\mathbf{w}^\top \mathbf{x} + b$

$$\mathbf{w}^\top \mathbf{x} + b > 0 \iff \sum_{\alpha=1}^{d} [\mathbf{x}]_\alpha \overbrace{(\log(\theta_{\alpha+}) - \log(\theta_{\alpha-}))}^{[\mathbf{w}]_\alpha} + \overbrace{\log(\pi_+) - \log(\pi_-)}^{b} > 0 \tag{15}$$

$$\iff \exp\left(\sum_{\alpha=1}^{d} [\mathbf{x}]_\alpha (\log(\theta_{\alpha+}) - \log(\theta_{\alpha-})) + \log(\pi_+) - \log(\pi_-)\right) > 1 \tag{16}$$

$$\iff \prod_{\alpha=1}^{d} \frac{\exp\left(\log \theta_{\alpha+}^{[\mathbf{x}]_\alpha} + \log(\pi_+)\right)}{\exp\left(\log \theta_{\alpha-}^{[\mathbf{x}]_\alpha} + \log(\pi_-)\right)} > 1 \tag{17}$$

$$\iff \prod_{\alpha=1}^{d} \frac{\theta_{\alpha+}^{[\mathbf{x}]_\alpha} \pi_+}{\theta_{\alpha-}^{[\mathbf{x}]_\alpha} \pi_-} > 1 \tag{18}$$

$$\iff \frac{\prod_{\alpha=1}^{d} P([\mathbf{x}]_\alpha | Y = +1)\pi_+}{\prod_{\alpha=1}^{d} P([\mathbf{x}]_\alpha | Y = -1)\pi_-} > 1 \tag{19}$$

$$\iff \frac{P(\mathbf{x}|Y = +1)\pi_+}{P(\mathbf{x}|Y = -1)\pi_-} > 1 \tag{20}$$

$$\iff \frac{P(Y = +1|\mathbf{x})}{P(Y = -1|\mathbf{x})} > 1 \tag{21}$$

$$\iff P(Y = +1|\mathbf{x}) > P(Y = -1|\mathbf{x}) \tag{22}$$

$$\iff \operatorname*{argmax}_y P(Y = y|\mathbf{x}) = +1 \tag{23}$$

Gaussian Naive Bayes

In the case of continuous features (Gaussian Naive Bayes), we can show that:

$$P(y \mid \mathbf{x}) = \frac{1}{1 + e^{-y(\mathbf{w}^\top \mathbf{x} + b)}}$$

This model is also known as **logistic regression**.

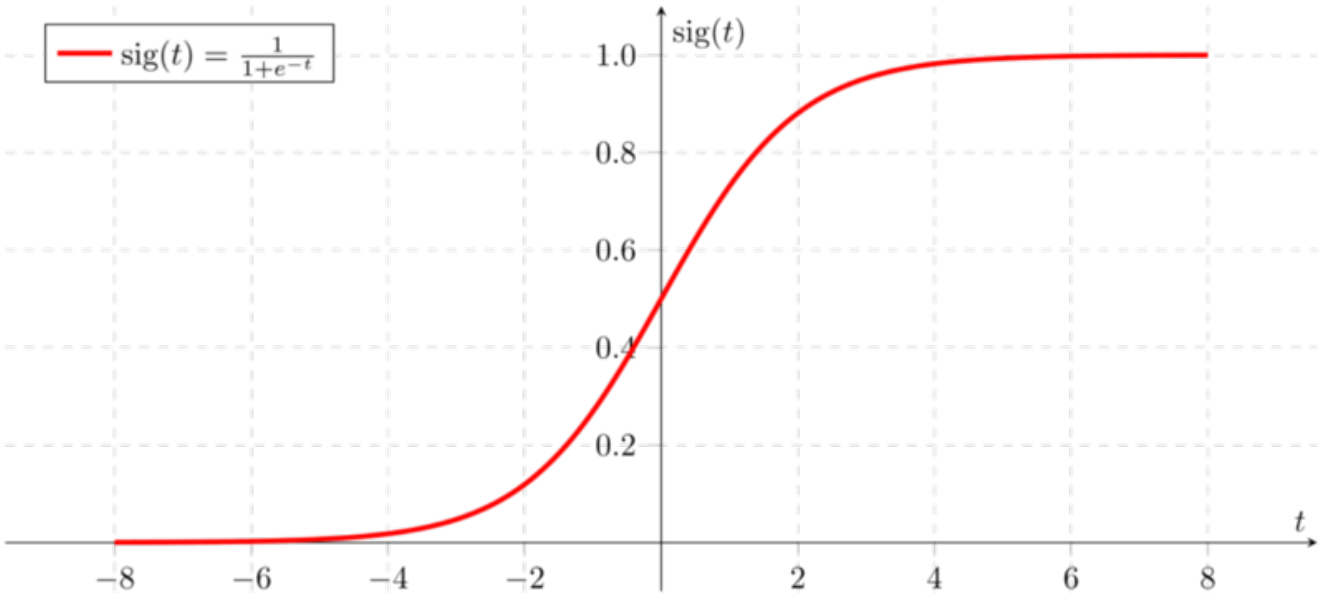


图: If 't' goes to infinity, y(predicted) will become 1 and if 't' goes to negative infinity, y(predicted) will become 0.

Gaussian Naive Bayes

$P(y|x)=\frac{1}{1+e^{-y(w^Tx+b)}}$ 恰是LR的MLE分布假设 $\underset{y}{argmin}\, p(y|x), y=\{0,1\}$

$$\theta = P(y=1|x) = \frac{P(y=1)p(x|y=1)}{p(y=1)p(x|y=1)+p(y=0)p(x|y=0)}$$

$$=\frac{1}{1+\exp(ln\frac{p(y=0)p(x|y=0)}{p(y=1)p(x|y=1)})} = \frac{1}{1+\exp(ln\frac{p(y=0)}{p(y=1)}+\sum_{i=1}^{n} ln\frac{p(x_i|y=0)}{p(x_i|y=1)})}$$

若属于高斯分布，则

$$\sum_{i=1}^{n} ln\frac{p(x_i|y=0)}{p(x_i|y=1)} = \sum_{i=1}^{n} ln\frac{\frac{1}{\sqrt{2\pi}\sigma^2}\exp(\frac{-(x_i-\mu_{i0})^2}{2{\sigma_i}^2})}{\frac{1}{\sqrt{2\pi}\sigma^2}\exp(\frac{-(x_i-\mu_{i1})^2}{2{\sigma_i}^2})} = -\sum_{i=1}^{n}(\frac{(x_i-\mu_{i0})^2}{2{\sigma_i}^2} - \frac{(x_i-\mu_{i1})^2}{2{\sigma_i}^2})$$

$$=-\sum_{i=1}^{n}(\frac{1}{2{\sigma_i}^2}(2(\mu_{i0}-\mu_{i1})x_i + ({\mu_{i0}}^2 - {\mu_{i1}}^2)) = -\sum_{i=1}^{n} w_i x_i + w_0$$

设p(y=1) $=\pi$, $p(y=0) = 1-\pi$

$$\theta = P(y=1|x) = \frac{1}{1+\exp(ln\frac{\pi}{1-\pi}-(w^Tx+w_0))} = \frac{1}{1+\exp(-y(w^Tx+b))}$$

结果证明Naive Bayes在高斯分布时本质上是Logistic Regression

Table of Contents

Filter spam with naive bayes

Core algorithm:Naive Bayesian classifier training function

```
def trainNB0(trainMatrix, trainCategory):计算训练的文档数目
    numTrainDocs = len(trainMatrix)计算文档的词条数
    numWords = len(trainMatrix[0])文档属于垃圾邮件类的概率
    pAbusive = sum(trainCategory)/float(numTrainDocs)初始化
    p0Num = ones(numWords); p1Num = ones(numWords)
    p0Denom = 2.0; p1Denom = 2.0
    for i in range(numTrainDocs):
        if trainCategory[i] == 1:统计计算词语属于垃圾邮件类的条件概率所需的数据
            p1Num += trainMatrix[i]
            p1Denom += sum(trainMatrix[i])
        else:统计计算属于非垃圾邮件类的条件概率所需的数据
            p0Num += trainMatrix[i]
            p0Denom += sum(trainMatrix[i])
    相除计算概率向量
    p1Vect = log(p1Num / p1Denom)
    p0Vect = log(p0Num / p0Denom)
    返回词语属于垃圾邮件类的条件概率向量，词语属于非垃圾邮件类的条件概率向量，文档属于垃
```

Classify

Classify

def classifyNB(vec2Classify, p0Vec, p1Vec, pClass1):

输入为需要分类的词向量，以及词语属于垃圾邮件类的条件概率向量，词语属于非垃圾邮件类的条件概率向量，文档属于垃圾邮件类的概率

```
p1=sum(vec2Classify*p1Vec)+log(pClass1)
p0=sum(vec2Classify*p0Vec)+log(1.0-pClass1)
if p1 > p0:
    return 1
else:
    return 0
```

Because:

$$p(c_i|\mathbf{w}) = \frac{p(\mathbf{w}|c_i)p(c_i)}{p(\mathbf{w})}, w : word\ \ vector; c_i : label$$

$$p(\mathbf{w}|c_i) = p(w_0, w_1, ..., w_N|c_i) = p(w_0|c_i)p(w_1|c_i)...p(w_N|c_i)$$

$$log(p(\mathbf{w}|c_i)p(c_i))$$

$$= log(p(w_0|c_i)p(w_1|c_i)...p(w_N|c_i)p(c_i))$$

$$= log(p(w_0|c_i)) + log(p(w_1|c_i)) + ... + log(p(w_N|c_i)) + log(p(c_i))$$

Table of Contents

Summary of Naive Bayes

Bayesian formula:

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

Assumption:

$$p(x_1, x_2, ..., x_n|y) = p(x_1|y)p(x_2|y)...p(x_n|y)$$

Likelihood function:

$$\prod_{i=1}^{n} p(x_i|y, \theta)$$

Log-likelihood:

$$\sum_{i=1}^{n} log(p(x_i|y, \theta))$$

Maximum likelihood estimation:

$$\underset{\theta}{\text{argmax}} \sum_{i=1}^{n} log(p(x_i|y, \theta))$$

Classify:

$$\underset{y}{\text{argmax}}\, p(y) \prod_{i=1}^{n} p(x_i|y, \theta) = \underset{y}{\text{argmax}}\, log(p(y)) + \sum_{i=1}^{n} log(p(x_i|y, \theta))$$

The End