Bayes Classifier and Naive Bayes

YuHua Li(李玉华)

Intelligent and Distributed Computing Lab,
Huazhong University of Science & Technology

idcliyuhua@hust.edu.cn

2019年04月09日

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:

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

- Estimate $P(X,Y)$:

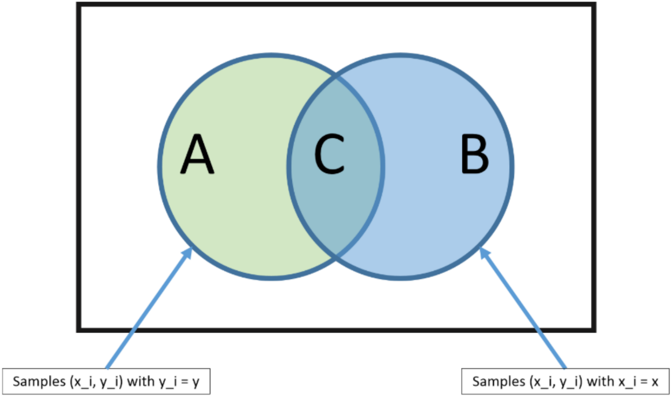
$$\hat{P}(\mathbf{x},y) = \frac{\sum_{i=1}^{n} I(\mathbf{x}_i = x \wedge y_i = y)}{n}.$$

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

- Estimate $P(y|\mathbf{x})$:predict the label y from the features $\mathbf{x}$

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

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 $P(y)$

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

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!

Bayes Classifier

Because of the Naive Bayes assumption

$$
\begin{aligned}
h(\mathbf{x}) &= \operatorname*{argmax}_y P(y|\mathbf{x}) && (1)\\
&= \operatorname*{argmax}_y \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})} && (2)\\
&= \operatorname*{argmax}_y P(\mathbf{x}|y)P(y) && (P(\mathbf{x}) \text{ does not depend on } y) \quad (3)\\
&= \operatorname*{argmax}_y \prod_{\alpha=1}^{d} P(x_\alpha|y)P(y) && (\text{by the naive Bayes assumption}) \quad (4)\\
&= \operatorname*{argmax}_y \sum_{\alpha=1}^{d} \log(P(x_\alpha|y)) + \log(P(y)) && (\text{as log is a monotonic function}) \quad (5)
\end{aligned}
$$

Table of Contents

Categorical features

Features:

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

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:

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

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

l is a smoothing parameter. By setting l=0 we get an MLE estimator, $l > 0$ leads to MAP. If we set l=+1 we get Laplace smoothing. in words,this means:

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

Prediction

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

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

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

Play-tennis example: estimating $P(x_i|C)$

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

P(p) = 9/14	
P(n) = 5/14	

outlook	
P(sunny|p) = 2/9	**P(sunny|n) = 3/5**
P(overcast|p) =4/9	**P(overcast|n) = 0**
P(rain|p) = 3/9	**P(rain|n) = 2/5**
temperature	
P(hot|p) = 2/9	**P(hot|n) = 2/5**
P(mild|p) = 4/9	**P(mild|n) = 2/5**
P(cool|p) = 3/9	**P(cool|n) = 1/5**
humidity	
P(high|p) = 3/9	**P(high|n) = 4/5**
P(normal|p) = 6/9	**P(normal|n) = 2/5**
windy	
P(true|p) = 3/9	**P(true|n) = 3/5**
P(false|p) = 6/9	**P(false|n) = 2/5**

Play-tennis example: classifying X

- An unseen sample X = <rain, hot, high, false>

- P(X|p)·P(p) =
  P(rain|p)·P(hot|p)·P(high|p)·P(false|p)·P(p) =
  3/9·2/9·3/9·6/9·9/14 = 0.010582
- P(X|n)·P(n) =
  P(rain|n)·P(hot|n)·P(high|n)·P(false|n)·P(n) =
  2/5·2/5·4/5·2/5·5/14 = 0.018286

- Sample X is classified in class n (don't play)

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:

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

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($\mathbf{x}|y$)

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 $P(\mathbf{x} \mid y = \text{spam})$. Parameter estimation:

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

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:

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

Prediction:

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

Continuous features

Features:

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

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 $\mu_{\alpha,y}$,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

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:

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

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

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

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

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

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

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

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

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

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

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

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

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.

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])文档属于侮辱类的概率
    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:统计计算词语属于侮辱类的条件概率所需的数据
            p1Num += trainMatrix[i]
            p1Denom += sum(trainMatrix[i])
        else:统计计算属于非侮辱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)
    返回词语属于侮辱类的条件概率向量，词语属于非侮辱类的条件概率向量，文档属于侮辱类概率.
```

Classify

```
def classifyNB(vec2Classify, p0Vec, p1Vec, pClass1):
  输入为需要分类的词向量，以及词语属于侮辱类的条件概率向量，词语属于非侮辱类的条件概率向量，文档属于侮辱类的概率
  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:

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

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:

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

Classify:

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

The End