ĐẠI HỌC TÔN ĐỨC THẮNG
TON DUC THANG UNIVERSITY

Lecture 2
Basic Principles of Statistical Machine Learning and Naive Bayes Classifiers

LÊ ANH CƯỜNG

Ton Duc Thang University

Outline

1. The general model of learning from examples
2. Empirical risk minimization inductive principle
3. Probability Theory and Bayesian Classification
4. Generative and Discriminative Models
5. Naive Bayesian Classification

The General Model of Learning from Examples

- Suppose that there is a functional relationship between two sets of objects X and Y:

$$f: X \rightarrow Y$$

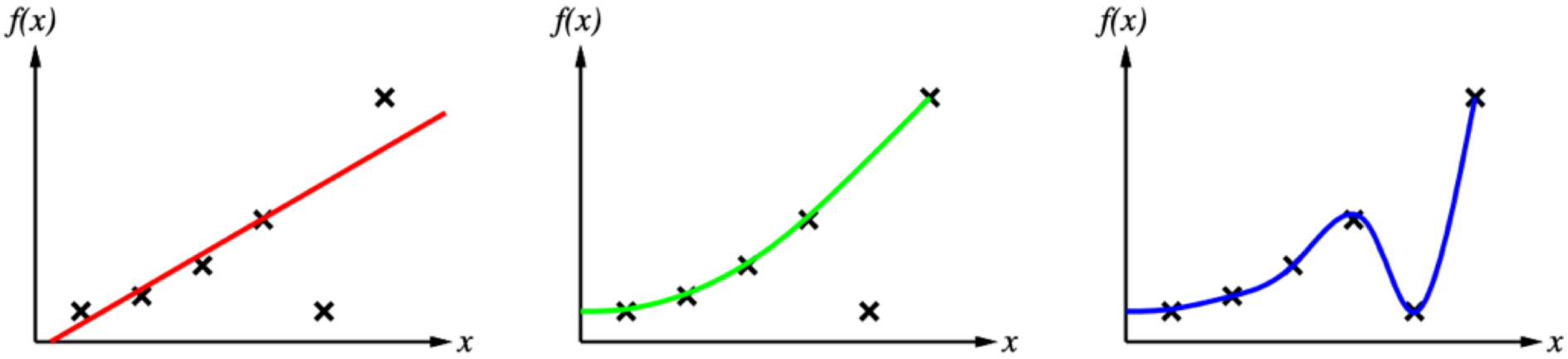
- Given a finite set of examples:

$$D = \{(\mathbf{x}_i, y_i) \mid i=1,2,\ldots,N\} \text{ , where } \mathbf{x}_i \in X \text{ and } y_i \in Y$$

- The task here is to derive (i.e. to learn) the objective function f

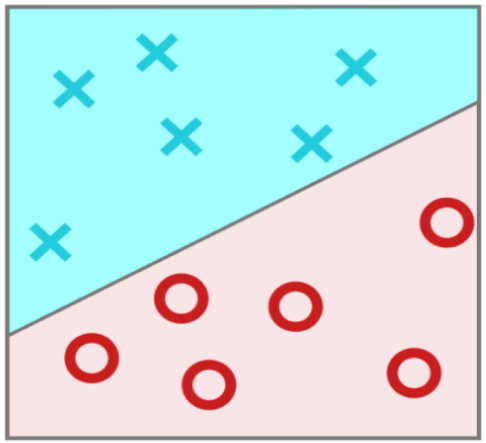
Objective of Learning

- Learn to generalize from a finite set of examples
- The learnt function then can predict output y given a new input $\mathbf{x}$

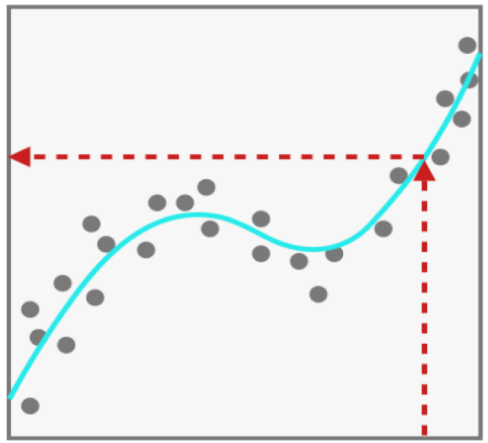


Classification and Regression

- $y = f(\mathbf{x})$
- If y is the real value ie Y = R then we have a regression problem
- If y is a value in a given finite discrete set, then we have a classification problem



Classification



Regression

Data Representation

- **x** is a vector of features

$$\mathbf{x} = (x_1, x_2, ..., x_d)$$

$$\mathbf{X} = R^d$$

- y is a real number in the regression problem
- y is in classification problem:
  - binary classification, y = {0,1} or {-1,+1}
  - multiple classes: y = {1, 2, ..., k} or one-hot vector (0,...0,1,0,...0)

Loss function

- Suppose that ($\boldsymbol{x}$,y) is an example. We want to find the difference between the ground true value y and the predicted value h($\boldsymbol{x}$)
- For regression:

$$L(y,h(x)) = \left(y - h(x)\right)^2$$

- For classification:

$$L(y,h(x)) = \begin{cases} 0 \text{ nếu } y = h(x) \\ \\ 1 \text{ nếu } y \neq h(x) \end{cases}$$

Expected Risk and Empirical Risk

- Expected risk/loss is the mean of L(y,h(**x**)) over the whole space ***X*** x ***Y***

$$R(h) = \iint L(y, h(x))p(x, y)dxdy$$

- Empirical risk/loss is the mean of L(y,h(**x**)) over the training dataset D

$$R_{emp}(h) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, h(x_i))$$

Empirical Risk

- For regression:

$$R_{emp}(h) = \frac{1}{N}\sum_{i=1}^{N}(y_i - h(x_i))^2$$

- For classification:

$$R_{emp}(h) = \frac{1}{N}\sum_{i=1}^{N}\delta(y_i, h(x_i))$$

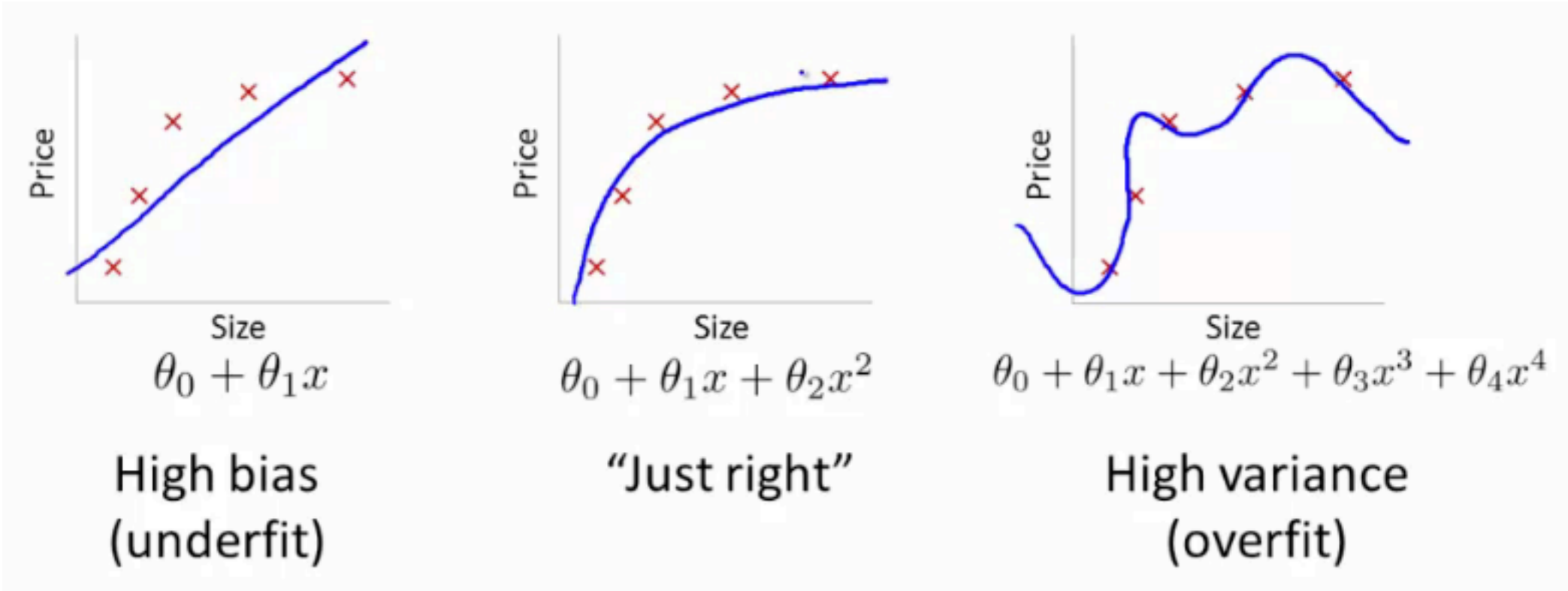
Empirical risk minimization inductive principle. (Nguyên lý quy nạp cực tiểu sai số thực nghiệm)

- We will consider the objective function f by the approximation function g as follows:

$$\mathbf{H} = \{h : X \to Y\}$$

$$g = \arg\min_{h} R_{emp}(h)$$

Overfitting



Probability Theory for Statistical Machine Learning

- Probability theory is a mathematical framework for quantifying our uncertainty about the world. It allows to reason effectively in situations where being certain is impossible. Probability theory is at the foundation of many machine learning algorithms.
- Probability Theory simply talks about how likely is the event to occur, and its value always lies between 0 and 1 (inclusive of 0 and 1)

Goals (X)	Probability P(X)
0	0.18
1	0.34
2	0.35
3	0.11
4	0.02

$$\sum_{i=0}^{n} p(A_i) = 1$$

Some basic probabilities

product rule $p(X,Y) = p(Y|X)p(X)$

sum rule $p(X) = \sum_{Y} p(X,Y)$

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

$$\sum_{i=1}^{\infty} p(E_i|A) = 1.$$

	Male	Female	Total
Football	120	75	195
Rugby	100	25	125
Other	50	130	180
	270	230	500

Probability Theory for Statistical Machine Learning

Discrete Probability Distribution: The mathematical definition of a discrete probability function, p(x), is a function that satisfies the following properties. This is referred as **Probability Mass Function**.

1. The probability that x can take a specific value is p(x), i.e.

$$P[X=x]=p(x)$$

2. p(x) is non-negative for all real x.

3. The sum of p(x) over all possible values of x is 1, i.e.

$$\sum_x P(x) = 1$$

Continuous Probability Distribution: The mathematical definition of a continuous probability function, f(x), is a function that satisfies the following properties. This is referred as **Probability Density Function**.

1. The probability that x is in between two points a and b is P(x)

$$P[a \leq x \leq b] = \int_a^b f(x)dx$$

2. f(x) is non-negative for all real x.

3. The sum of p(x) over all possible values of x is 1, i.e.

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

Probability Theory for Statistical Machine Learning

Formally, an "ordinary" classifier is some rule, or function, that assigns to a sample x a class label $\hat{y}$:

$$\hat{y} = f(x)$$

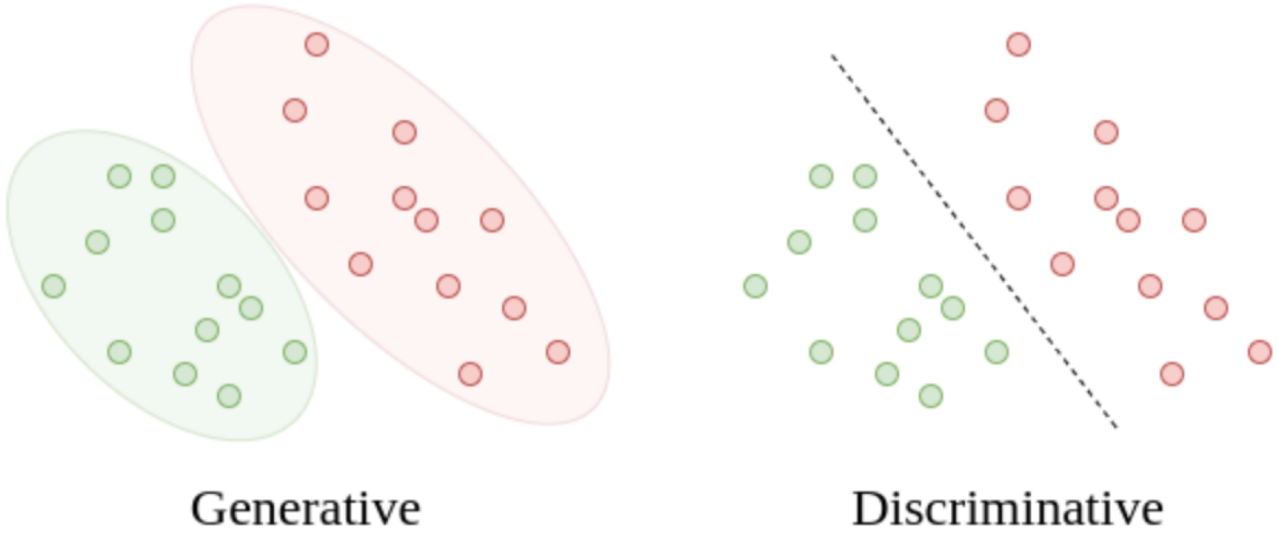
Probabilistic classifiers generalize this notion of classifiers: instead of functions, they are conditional distributions $\Pr(Y|X)$, meaning that for a given $x \in X$, they assign probabilities to all $y \in Y$ (and these probabilities sum to one). "Hard" classification can then be done using the optimal decision rule[2]:39–40

$$\hat{y} = \arg\max_y \Pr(Y = y|X)$$

Discriminate and Generative Models

Let's say you have input data x and you want to classify the data into labels y.

A generative model learns the **joint** probability distribution p(x,y) and a discriminative model learns the **conditional** probability distribution p(y|x)



Generative Discriminative

Discriminate and Generative Models

Let's say you have input data x and you want to classify the data into labels y. A generative model learns the **joint** probability distribution p(x,y) and a discriminative model learns the **conditional** probability distribution p(y|x)

Some popular discriminative algorithms are:
- k-nearest neighbors (k-NN)
- Logistic regression
- Support Vector Machines
- Decision Trees
- Random Forest
- Artificial Neural Networks (ANNs)

Some popular generative algorithms are:
- Naive Bayes Classifier
- Generative Adversarial Networks
- Gaussian Mixture Model
- Hidden Markov Model
- Probabilistic context-free grammar

Bayesian Classification

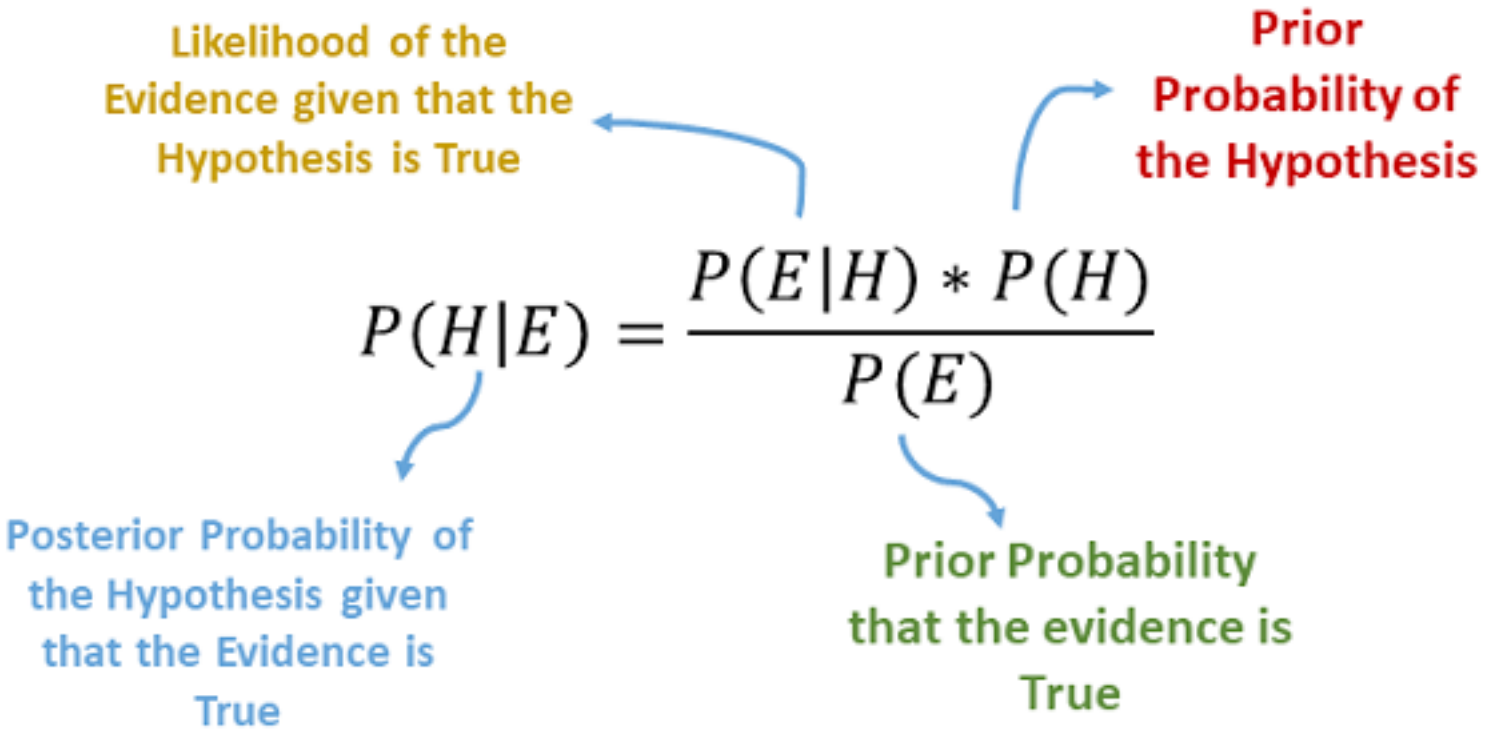
$$f: \mathbf{X} \rightarrow \mathbf{C} = \{c_1, \ldots, c_k\}$$

$$P(c_i|\boldsymbol{x}) = \frac{P(\boldsymbol{x}, c_i)}{P(\boldsymbol{x})} = \frac{P(\boldsymbol{x}|c_i) * P(c_i)}{P(\boldsymbol{x})}$$

$$c^* = \underset{c_i}{\operatorname{argmax}}\, P(c_i|\boldsymbol{x})$$

$$= \underset{c_i}{\operatorname{argmax}}\, P(\boldsymbol{x}|c_i) * P(c_i)$$

Bayesian Classification



Bayesian Classification and Expected Risk

$$L(c_i, h(x)) = \begin{cases} 0 \text{ nếu } h(x) = c_i \\ 1 \text{ nếu } h(x) \neq c_i \end{cases}$$

Then, the expected Risk at input **x** will be:

$$R(h/x) = \sum_{i=1}^{k} L(c_i, h(x)) P(c_i / x)$$

Bayesian Classification and Expected Risk

- Suppose $h(\mathbf{x}) = c_j$, then:

$$R(h/x) = \sum_{i=1}^{k} L(c_i, h(x)) P(c_i/x)$$

$$R(h/x) = \sum_{i \neq j} P(c_i/x)$$

- It means:

$$R(h/x) = 1 - P(c_j/x)$$

- So that to minimize the Expected Risk, it is equivalent to choose for maximizing $P(c_j|\mathbf{x})$

Maximum Likelihood Estimation

- We are given a data set $D = \{x_1, x_2, ..., x_N\}$
- Suppose that the given examples come from the probability distribution with parameter θ
- We need to estimate θ that maximize $p(D)$

$$\theta = \text{argmax } p(x_1, x_2, \ldots, x_N|\theta)$$

- $p(D)$ is likelihood of D

$$\theta = \text{argmax } \prod p(x_i|\theta)$$

Maximum Likelihood Estimation

$$\theta = \text{argmax} \prod p(x_i|\theta)$$

- To make the calculation more convenient, we can use Maximum Log-likelihood:

$$\theta = \text{argmax} \sum \log(p(x_i|\theta))$$

Example

Suppose the problem is that there are 5 students taking the test with scores of 3, 6, 5, 9, 8 respectively. To model the scores of these students, we assume that the data points are segregated. distributed according to the Gaussian distribution:

$$p(x_i|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x_i-\mu)^2}{2\sigma^2}\right)$$

Example

$$\mu, \sigma = \arg\max_{\mu,\sigma} \left[\prod_{i=1}^{N} p(x_i|\mu, \sigma^2) \right]$$

$$= \arg\max_{\mu,\sigma} \left[\frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(-\frac{\sum_{i=1}^{N}(x_i - \mu)^2}{2\sigma^2} \right) \right]$$

$$\frac{\partial J}{\partial \mu} = \frac{1}{\sigma^2} \sum_{i=1}^{N} (x_i - \mu)$$

$$\frac{\partial J}{\partial \sigma} = -\frac{N}{\sigma} + \frac{1}{\sigma^3} \sum_{i=1}^{N} (x_i - \mu)^2$$

$$\mu = \frac{\sum_{i=1}^{N} x_i}{N}$$

$$\sigma^2 = \frac{\sum_{i=1}^{N} (x_i - \mu)^2}{N}$$

$\mu = 6.2$ and $\sigma = 2.14$

Naive Bayesian Classification

Outlook	Tempreature	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

	Giờ dậy (x1)	Sức khoẻ (x2)	Thời tiết (x3)	Đi muộn (y)
1	Sớm	Tốt	Nắng	Không
2	Sớm	Xấu	Mưa	Không
3	Bình thường	Tốt	Nắng	Có
4	Muộn	Xấu	Nắng	Có
5	Sớm	Xấu	Nhiều mây	Không
6	Bình thường	Xấu	Nhiều mây	Không
7	Muộn	Tốt	Nắng	Có
8	Bình thường	Tốt	Nắng	Không
9	Sớm	Xấu	Nhiều mây	Có
10	Muộn	Tốt	Mưa	Có

Naive Bayesian Classification

1. Model
2. Parameter Estimation with Different Distribution of Data

NB Classification

- Bayesian classification

$$P(class|data) = \frac{P(data|class) * P(class)}{P(data)}$$

$$target = \underset{c_i}{\operatorname{argmax}}\, P(data|c_i) * P(c_i)$$

NB Classification

- How to estimate the model's parameters:

$$P(X|y) \text{ and } P(y)$$

- X is represented by a vector of feature values $X = (x_1, x_2, .., x_d)$

$$P(X|y) = P(x_1, x_2, .., x_d|y)$$

with a naive assumption we have

$$P(X|y) = P(x_1|y) * P(x_2|y) * ... * P(x_d|y)$$

NB classification

- How to estimate the model's parameters:

The task now is to calculate/estimate the probabilities:

$$P(c_j) \text{ and } P(x_j|c_k)$$

where $P(c_j)$ is the probability of a class c_j, and $P(x_j|c_k)$ is the probability of a value x_j (of a feature j^{th}) with the condition of class c_k.

These probabilities are estimated based on the probability distribution of:

$$P(c) \text{ and } P(x|c)$$

NB classification

- How to estimate the model's parameters:

These probabilities are estimated based on the probability distribution of:

$$P(c) \text{ and } P(x|c)$$

where: c is the class variable

x is feature value variable

for example: P(c=N), P(c=P)

P(outlook = sunny | c=P)

Outlook	Tempreature	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

NB Classification

- What are the parameters of the NB Model?

- What is the inference process of the NB Model?
  - given input: $X = (x_1, x_2, .., x_d)$
  - we have: ?

NB Classification

- What are the parameters of the NB Model?

$$P(c) \text{ and } P(x|c)$$

- What is the inference process of the NB Model?
  - given input: $X = (x_1, x_2, .., x_d)$
  - we have:

$$target = \underset{c_i}{\text{argmax}}\, P(x_1|c_i) * \ldots * P(x_d|c_i) * P(c_i)$$

NB Classification

- Parameter Estimation

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

- Then:

$$P(y = c_i) = \frac{count(c_i)}{N}$$

where N is the number of training examples.

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(x_i \mid y)$.

Multinomial NB

`MultinomialNB` implements the naive Bayes algorithm for multinomially distributed data, and is one of the two classic naive Bayes variants used in text classification (where the data are typically represented as word vector counts, although tf-idf vectors are also known to work well in practice). The distribution is parametrized by vectors $\theta_y = (\theta_{y1}, \ldots, \theta_{yn})$ for each class y, where n is the number of features (in text classification, the size of the vocabulary) and θ_{yi} is the probability $P(x_i \mid y)$ of feature i appearing in a sample belonging to class y.

The parameters θ_y is estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting:

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$

where $N_{yi} = \sum_{x \in T} x_i$ is the number of times feature i appears in a sample of class y in the training set T, and $N_y = \sum_{i=1}^{n} N_{yi}$ is the total count of all features for class y.

Gaussian NB

When working with continuous data, an assumption often taken is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. The likelihood of the features is assumed to be”

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

The parameters σ_y and μ_y are estimated using maximum likelihood.

Other NB Classifiers

- Complement Naive Bayes
- Bernoulli Naive Bayes
- Categorical Naive Bayes

Reference:

- https://scikit-learn.org/stable/modules/naive_bayes.html

Practice

- https://www.kaggle.com/code/prashant111/naive-bayes-classifier-in-python