Naïve Bayes

Wang Houfeng

Institute of Computational Linguistics

Peking University

Outline

- **Introduction**
- Maximum Likelihood Estimation
- Naïve Bayesian Classification

Introduction

Bayesian learning enables us to form predictions based on **probabilities**. It provides a framework for **probabilistic reasoning**. The advantages:

- There may be noise in the data;
- can provide “prior knowledge” in constructing a hypothesis;
- Predictions are probabilistic;
- The framework provides a view of optimal decision making

Probability Definition

- A *probability* is a function from an event space to a real number between 0 and 1 (inclusive), where,
  - $0 \leq P(A) \leq 1$,
    - 0: indicates impossibility
    - 1: indicates certainty
  - $P(\Omega) = 1$ *where,* Ω *:* sample space
  - $P(X) \leq p(Y)$ for any $X \subseteq Y$
  - If $A_1, A_2, \ldots, A_n$ is disjoint $(A_i \cap A_j) = \varnothing$, then

$$P\left(\bigcup_{i=1}^{n} A_i\right) = \sum_{i=1}^{n} P(A_i)$$

Interpretation of probability

- Relative Frequency
  - Suppose that an experiment is performed ***n*** times and A occurs ***f*** times. The relative frequency for an event A is:

$$\frac{\text{Number of times A occurs}}{n} = \frac{f}{n}$$

- If we let *n* get infinitely large,

$$P(A) = \lim_{n \to \infty} \frac{f}{n}$$

Interpretation of probability

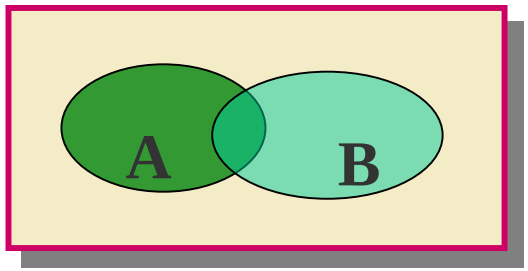
- In frequentist statistics, probabilities are associated only with the data, i.e., outcomes of repeatable observations:

  Probability = limiting frequency

Some Rules

- For any two events, **A** and **B**, the probability of their union, **P(A ∪ B)**, is

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$



- A Special Case, When two events A and B are **mutually exclusive,** $P(A \cap B) = 0$ and **P(A∪B) = P(A) + P(B).**

Independence

- Two events A and B are independent if and only if:

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

or

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

or

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

Probabilistic Classification

- Input: $\boldsymbol{x} = [x_1, x_2]^T$,Output: C $\in \{-1, 1\}$
- Prediction:

$$\text{choose} \begin{cases} C = 1 \text{ if } P(C = 1 | x_1, x_2) > 0.5 \\ C = -1 \text{ otherwise} \end{cases}$$

or equivalently

$$\text{choose} \begin{cases} C = 1 \text{ if } P(C = 1 | x_1, x_2) > P(C = -1 | x_1, x_2) \\ C = -1, \text{ otherwise} \end{cases}$$

Basic of Probability Learning

- Goal: find the best hypothesis from some space H of hypotheses, given the observed data D;
- Define best to be: most probable hypothesis in H;
- In order to do that, we need to assume a probability distribution over the H;
- In addition, we need to know something about the relation between the data observed and the hypotheses.

Bayes Theorem

Hypothesis space : H,
Dataset: D.
Four probabilities are introduced as follows:

- **P(h):** the prior probability of h∈H before data is observed. Reflects background knowledge; If no information, uniform distribution is chosen.
- **P(D):** the probability of seeing data D, it is **evidence**. (No knowledge of the hypothesis)
- **P(D|h):** is the probability of the data given h. It is called the **likelihood of h with respect to D**.
- **P(h|D):** The posterior probability of h after having seen data D. The probability h is the target.

Bayes Theorem

Bayes theorem relates the posterior probability of a hypothesis given the data with the three probabilities mentioned before:

Posterior probability **Likelihood** **Prior probability**

$$P(h \mid D) = \frac{P(D \mid h) \bullet P(h)}{P(D)}$$

Evidence

Hypotheses in Bayesian

- Hypotheses *h* refers to processes that could have *generated* the data *D*
- Bayesian inference provides a distribution over these hypotheses, given *D*
- $P(D|h)$ is the probability of *D* being generated by the process identified by *h*
- Hypotheses *h* are mutually exclusive: only one process could have generated *D*

The origin of Bayes' rule

- For any two random variables:

$$p(A,B) = p(A)\,p(B \mid A)$$
$$p(A,B) = p(B)\,p(A \mid B)$$

$$p(B)\,p(A \mid B) = p(A)\,p(B \mid A)$$

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

Bayes' rule in odds form

$$\frac{P(h_1 \mid D)}{P(h_2 \mid D)} = \frac{P(h_1, D) \big/ p(D)}{P(h_2, D) \big/ p(D)} = \frac{P(h_1, D)}{P(h_2, D)}$$

$$= \boxed{\frac{p(D \mid h_1)}{p(D \mid h_2)}} \frac{p(h_1)}{p(h_2)}$$

likelihood ratio

D:	data
h_1, h_2:	models
$P(h_1 \mid D)$:	posterior probability h_1 generated the data
$P(D \mid h_1)$:	likelihood of data under model h_1
$P(h_1)$:	prior probability h_1 generated the data

Comparing two hypotheses

$$\frac{P(h_1 \mid D)}{P(h_2 \mid D)} = \frac{p(D \mid h_1)}{p(D \mid h_2)} \frac{p(h_1)}{p(h_2)}$$

D: **HHTHT**

Hypotheses h_1:"fair coin"; Hypotheses h_2: "always heads"

$P(D|h_1) = 1/2^5$ $P(h_1) = 999/1000$

$P(D|h_2) = 0$ $P(h_2) = 1/1000$

$P(h_1|D) / P(h_2|D)$ = infinity

Comparing two hypotheses

$$\frac{P(h_1 \mid D)}{P(h_2 \mid D)} = \frac{p(D \mid h_1)}{p(D \mid h_2)} \frac{p(h_1)}{p(h_2)}$$

D: **HHHHH**

Hypotheses h_1:"fair coin"; Hypotheses h_2: "always heads"

$P(D|h_1) = 1/2^5$ $P(h_1) = 999/1000$

$P(D|h_2) = 1$ $P(h_2) = 1/1000$

$P(h_1|D) / P(h_2|D) \approx 30$

Comparing two hypotheses

$$\frac{P(h_1 \mid D)}{P(h_2 \mid D)} = \frac{p(D \mid h_1)}{p(D \mid h_2)} \frac{p(h_1)}{p(h_2)}$$

D: **HHHHHHHHHH**

h_1, h_2: "fair coin", "always heads"

$P(D|h_1) = 1/2^{10}$ $P(h_1) = 999/1000$

$P(D|h_2) = 1$ $P(h_2) = 1/1000$

$P(h_1|D) / P(h_2|D) \approx 1$

h_1 vs. h_2

- Hypotheses h_1:“fair coin”; Hypotheses h_2: “always heads”
- Which one is better?

For K (>2) Classes

$$P(h_i|D) = \frac{P(D|h_i)P(h_i)}{P(D)}$$

$$= \frac{P(D|h_i)P(h_i)}{\sum_{k=1}^{K} P(D|h_k)P(h_k)}$$

$$P(h_i) \geq 0 \quad \text{and} \quad \sum_{i=1}^{K} P(h_i) = 1$$

$$\text{choose} \quad h_i \quad \text{if} \quad P(h_i|D) = \max_k P(h_k|D)$$

Outline

- Introduction
- **Maximum Likelihood Estimation**
- Naïve Bayesian Classification

Maximum A Posteriori

- Now, attempts to find the most probable one h $\in$H, given the observed data.
- A method that looks for the hypothesis with maximum P(h|D) is called a **maximum a posteriori** method or MAP.

$$h_{MAP} = \underset{h \in H}{\text{Argmax}} \; P(h \mid D)$$

$$= \underset{h \in H}{\text{Argmax}} \; \frac{P(D \mid h) \bullet P(h)}{P(D)}$$

$$= \underset{h \in H}{\text{Argmax}} \; P(D \mid h) \bullet P(h)$$

$$where, \; P(D) \; is \; independent \; of \; h$$

Example: QA

- Query: *Does patient have cancer or not?*
- Two hypothesis: the patient has cancer, $\oplus$, the patient doesn't have cancer, $\ominus$;
- Prior knowledge: over the entire population of people .008 have cancer;
- The lab test returns a correct positive result in 98% of the cases in which cancer is actually present and a correct negative in 97% of the cases in which cancer is actually not present
- P(cancer) = .008, P(¬cancer) = .992
- P($\oplus$|cancer) = .98, P($\ominus$|cancer) = .02
- P($\oplus$|¬cancer)=.03, P($\ominus$|¬cancer)=.97
- So given a new patient with a positive lab test, should we diagnose the patient as having cancer or not??

Example: QA

$$P(Cancer) = 0.008 \qquad P(\oplus \mid Cancer) = 0.98 \qquad P(\oplus \mid \neg Cancer) = 0.03$$

$$P(\neg Cancer) = 0.992 \qquad P(\ominus \mid Cancer) = 0.02 \qquad P(\ominus \mid \neg Cancer) = 0.97$$

P(⊕|cancer)P(cancer) = (.98).008 = .0078

P(⊕|¬cancer)P(¬cancer)=(.03).992=.0298

- Which is the MAP hypothesis?

$$P(cancer \mid \oplus) = \frac{P(\oplus \mid cancer)P(cancer)}{P(\oplus)} = \frac{.0078}{.0078 + .0298} = .21$$

- So, the MAP hypothesis : h_{MAP} = ¬ cancer

Maximum Likelihood hypothesis

- Assume that a priori, hypotheses are equally probable.

$$P(h_i) = P(h_j), \forall h_i, h_j \in H$$

- Maximum Likelihood hypothesis can be got:

$$h_{ML} = \underset{\mathbf{h} \in \mathbf{H}}{\operatorname{argmax}} P(D \mid h)$$

- Now, just need to look for the hypothesis that best explains the data.

Maximum Likelihood Estimator

Let $D=\{x_1,x_2,...x_n\}$ is training set,

$$h_{ML}(h) = \underset{h\in H}{\operatorname{argmax}}\ P(D \mid h) = \underset{h\in H}{\operatorname{argmax}} \prod_{x_i} P(x_i \mid h)$$

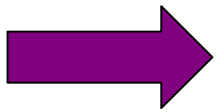
Let $$\frac{\partial h_{ML}(h)}{\partial h} = 0$$

An simple example

- Assuming
  - A coin has a probability p of heads, 1-p of tails.
  - Observation: We toss a coin N times, and the result is a set of Hs and Ts, and there are M Hs.
- What is the value of p based on MLE, given the observation?

$$L(\theta) = \log P(D \mid \theta) = \log[\, p^M (1-p)^{N-M} \,]$$
$$= M \log p + (N-M)\log(1-p)$$

$$\frac{dL(\theta)}{dp} = \frac{d(M\log p + (N-M)\log(1-p))}{dp} = \frac{M}{p} - \frac{N-M}{1-p} = 0$$



p= M/N

Bayesian Learning : Unbiased Coin

- Coin Flip
  - Sample space: $\Omega = \{Head, Tail\}$
  - Scenario: given coin is either fair or has a 60% bias in favor of *Head*
    - $h_1 \equiv$ fair coin: $P(Head) = 0.5$
    - $h_2 \equiv$ 60% bias towards *Head*: $P(Head) = 0.6$
  - Objective: to decide between the hypotheses
- *A Priori* Distribution on *H*
  - $P(h_1) = 0.75$, $P(h_2) = 0.25$

Bayesian Learning : Unbiased Coin

- Collection of Evidence
  - First piece of evidence: d=a single coin toss, comes up *Head*
  - Q: What does the agent believe now?
  - A: Compute $P(d) = P(d \mid h_1)\, P(h_1) + P(d \mid h_2)\, P(h_2)$
- Bayesian Inference: Compute $P(d) = P(d \mid h_1)\, P(h_1) + P(d \mid h_2)\, P(h_2)$
  - $P(Head) = 0.5 \bullet 0.75 + 0.6 \bullet 0.25 = 0.375 + 0.15 = 0.525$
  - This is the probability of the observation $d = Head$

Bayesian Learning : Unbiased Coin

- Bayesian Learning
  - Now apply Bayes's Theorem
    - $P(h_1 \mid d) = P(d \mid h_1)\, P(h_1) / P(d) = 0.375 / 0.525 = 0.714$
    - $P(h_2 \mid d) = P(d \mid h_2)\, P(h_2) / P(d) = 0.15 / 0.525 = 0.286$
    - *Belief has been revised downwards for h_1, upwards for h_2*
    - The agent still thinks that the fair coin is the more likely hypothesis
  - Suppose we were to use the ML approach (i.e., assume equal priors)
    - Belief is revised upwards from 0.5 for h_1
    - Data then supports the bias coin better

Bayesian Learning : Unbiased Coin

- More Evidence: Sequence D of 100 coins with 70 heads and 30 tails
  - $P(D) = (0.5)^{50} \bullet (0.5)^{50} \bullet 0.75 + (0.6)^{70} \bullet (0.4)^{30} \bullet 0.25$
  - Now $P(h_1 \mid d) << P(h_2 \mid d)$

Brute Force MAP Hypothesis Learner

- Intuitive Idea: Produce Most Likely *h* Given Observed *D*
- Algorithm Find-MAP-Hypothesis (*D*)
  - 1. FOR each hypothesis $h \in H$

    Calculate the conditional (i.e., posterior) probability:

$$P(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)}$$

  - 2. RETURN the hypothesis h_{MAP} with the highest conditional probability

$$h_{MAP} = \underset{h \in H}{\text{Argmax}} \; P(h \mid D)$$

Outline

- Introduction
- Maximum Likelihood Estimation
- **Naïve Bayes Classification**

Bayesian Classification

- Framework
  - Find most probable *classification* (as opposed to MAP *hypothesis*)
  - $f: X \rightarrow C$ (domain ≡ instance space, range ≡ finite set of values)
  - Instances $x \in X$ can be described as a collection of features $x \equiv (a_1, a_2, \ldots, a_n)$
  - Performance element: Bayesian classifier
    - Given: an example
    - Output: the most probable value $c_j \in C$

$$v_{MAP} = \arg\max_{c_j \in C} P(c_j \mid x) = \arg\max_{c_j \in C} P(c_j \mid a_1, a_2, \ldots, a_n)$$

$$= \arg\max_{c_j \in C} P(a_1, a_2, \ldots, a_n \mid c_j) P(c_j)$$

Bayesian Classification

- Parameter Estimation Issues
  - Easily estimating $P(c_j)$: only count the frequency(c_j) in $D = \{<x, c(x)>\}$
  - But infeasible to estimate $P(a_1, a_2, \ldots, a_n \mid c_j)$: too many 0 values
  - N*eed to make assumptions* that allow us to estimate $P(x \mid c)$
- Intuitive Idea
  - $h_{MAP}(x)$ is not necessarily the most probable classification!
  - Example
    - Three possible hypotheses: $P(h_1 \mid D) = 0.4$, $P(h_2 \mid D) = 0.3$, $P(h_3 \mid D) = 0.3$
    - Suppose that for new instance x, $h_1(x) = +$, $h_2(x) = -$, $h_3(x) = -$
    - What is the most probable classification of x?

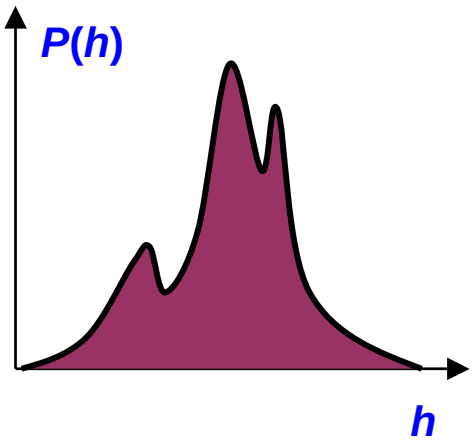
Bayes Optimal Classification (BOC)

$$c^* = c_{BOC} = \arg\max_{c_j \in C} \sum_{h_i \in H} \left[P(c_j \mid h_i) \cdot P(h_i \mid D) \right]$$

- Example:
  - $P(h_1 \mid D) = 0.4,\ P(- \mid h_1) = 0,\ P(+ \mid h_1) = 1$
  - $P(h_2 \mid D) = 0.3,\ P(- \mid h_2) = 1,\ P(+ \mid h_2) = 0$
  - $P(h_3 \mid D) = 0.3,\ P(- \mid h_3) = 1,\ P(+ \mid h_3) = 0$

$$\sum_{h_i \in H} \left[P(+ \mid h_i) \cdot P(h_i \mid D) \right] = 0.4$$

$$\sum_{h_i \in H} \left[P(- \mid h_i) \cdot P(h_i \mid D) \right] = 0.6$$



- Result: $c^* = c_{BOC} = \arg\max_{c_j \in C} \sum_{h_i \in H} \left[P(c_j \mid h_i) \cdot P(h_i \mid D) \right] = -$

New Issue

- BOC Computationally expensive. It computes the posterior prob for every **h∈H** and combines the predictions of each hypothesis to classify each new instance.
- Solution:
  - Choose a hypothesis h (eg. MAP)from H at random, according to the posterior probability distribution over H, and use it to predict the novel instances(Gibbs Classifier)!
  - We expect to simplify this estimation: **Naïve Bayes**

Naïve Bayes: Characterics

$$c_{MAP} = \arg\max_{c_j \in C} P(c_j \mid x) = \arg\max_{c_j \in C} P(c_j \mid a_1, a_2, \ldots, a_n)$$

$$= \arg\max_{c_j \in C} P(a_1, a_2, \ldots, a_n \mid c_j) P(c_j)$$

where, $x = (a_1, a_2, \ldots, a_n)$

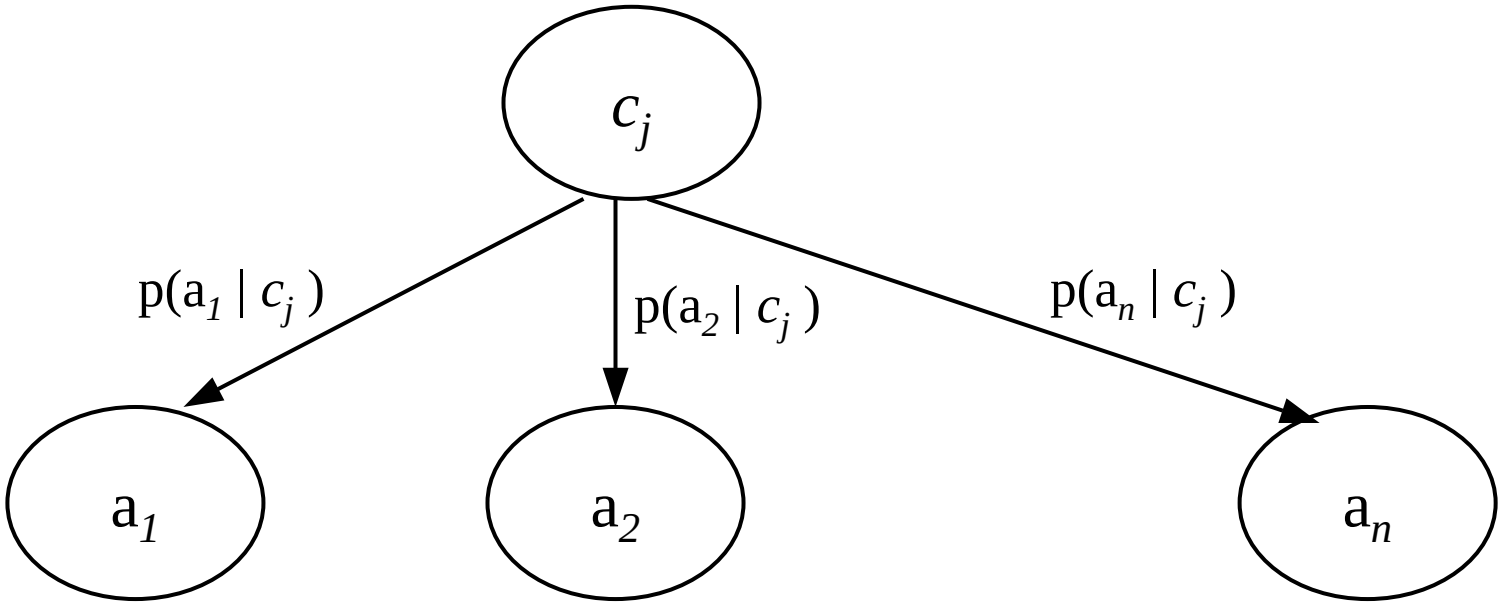
- Very difficult to compute the likelihood $P(a_1, a_2, \ldots, a_n \mid c_j)$
- To Simplify the assumption(Naïve Bayes):
  - attribute values *x* independent given target value *v*

$$P(a_1, a_2, \ldots, a_n \mid c_j) = \prod_i P(a_i \mid c_j)$$

$$c_{NB} = \arg\max_{c_j \in C} P(c_j) \prod_i P(a_i \mid c_j)$$

- Results comparable to ANN and decision trees in some domains
- Moderate or large training set available
- Successful Applications: **Classifying text documents**

Naive Bayes' Classifier



Given category c_j, a_i are independent:

$$p(\boldsymbol{x}|\, c_j) = p(a_1|\, c_j)\, p(a_2|\, c_j) \ldots p(a_n|\, c_j)$$

Naïve Bayes: Independence Issue

- Conditional Independence Assumption Often Violated
  - CI assumption: $P(a_1, a_2, \ldots, a_n \mid c_j) = \prod_k P(a_k \mid c_j)$
  - However, it works well surprisingly well anyway
  - *Note*
    - Don't need estimated conditional probabilities $\hat{P}(c_j \mid x)$ to be correct
    - Only need

$$c_{NB} = \arg\max_{c_j \in C} \left[\hat{P}(c_j) \prod_{k=1}^{n} \hat{P}(a_k \mid c_j) \right]$$

$$= \arg\max_{c_j \in C} \left[P(c_j) P(a_1, a_2, \ldots, a_n \mid c_j) \right]$$

Naïve Bayes Algorithm

- Simple (Naïve) Bayes Assumption $P(a_1, a_2, \ldots, a_n \mid c_j) = \prod_k P(a_k \mid c_j)$
- Simple (Naïve) Bayes Classifier $c_{NB} = \arg\max_{c_j \in C} P(c_j) \prod_k P(a_k \mid c_j)$
- Algorithm Naïve-Bayes-Learn
  - FOR each target value c_j

$$\hat{P}(c_j) \leftarrow estimate\left[P(c_j)\right]$$

FOR each attribute value a_k of each attribute x

$$\hat{P}(a_k \mid c_j) \leftarrow estimate\left[P(a_k \mid c_j)\right]$$

  - RETURN $\left\{ \hat{P}(a_k \mid c_j) \right\}$

Example

- **Concept: *PlayTennis***

$$c_{NB} = \arg\max_{c_j \in C} \left[\hat{P}(c_j) \prod_{k=1}^{n} \hat{P}(a_k \mid c_j) \right]$$

Day	Outlook	Temperature	Humidity	Wind	*PlayTennis*?
1	Sunny	Hot	High	Light	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Light	Yes
4	Rain	Mild	High	Light	Yes
5	Rain	Cool	Normal	Light	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Light	No
9	Sunny	Cool	Normal	Light	Yes
10	Rain	Mild	Normal	Light	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Light	Yes
14	Rain	Mild	High	Strong	No

Example

- Application of Naïve Bayes: to computation
  - ***P*(*PlayTennis* = {*Yes*, *No*})** 2 cases
  - ***P*(*Outlook* = {*Sunny, Overcast, Rain*} | *PT* = {*Yes*, *No*})** 6 cases
  - ***P*(*Temp* = {*Hot, Mild, Cool*} | *PT* = {*Yes*, *No*})** 6 cases
  - ***P*(*Humidity* = {*High, Normal*} | *PT* = {*Yes*, *No*})** 4 cases
  - ***P*(*Wind* = {*Light, Strong*} | *PT* = {*Yes*, *No*})** 4 cases
- Query: New Example *x* = <*Sunny*, *Cool*, *High*, *Strong*, ?>
  - **Desired inference: *P*(*PlayTennis* = *Yes* | *x*) = 1 - *P*(*PlayTennis* = *No* | *x*)**
  - ***P*(*PlayTennis* = *Yes*) = 9/14 = 0.64** ***P*(*PlayTennis* = *No*) = 5/14 = 0.36**
  - ***P*(*Outlook* = *Sunny* | *PT* = *Yes*) = 2/9** ***P*(*Outlook* = *Sunny* | *PT* = *No*) = 3/5**
  - ***P*(*Temperature* = *Cool* | *PT* = *Yes*) = 3/9** ***P*(*Temperature* = *Cool* | *PT* = *No*) = 1/5**
  - ***P*(*Humidity* = *High* | *PT* = *Yes*) = 3/9** ***P*(*Humidity* = *High* | *PT* = *No*) = 4/5**
  - ***P*(*Wind* = *Strong* | *PT* = *Yes*) = 3/9** ***P*(*Wind* = *Strong* | *PT* = *No*) = 3/5**

Example

$$c_{NB} = \arg\max_{c_j \in C} P(c_j) \prod_k P(a_k \mid c_j)$$

- Inference
  - *P*(*PlayTennis* = *Yes*, <*Sunny*, *Cool*, *High*, *Strong*>) =

    P(*Yes*) *P*(*Sunny* | *Yes*) *P*(*Cool* | *Yes*) *P*(*High* | *Yes*) *P*(*Strong* | *Yes*) ≈ 0.0053
  - *P*(*PlayTennis* = *No*, <*Sunny*, *Cool*, *High*, *Strong*>) =

    P(*No*) *P*(*Sunny* | *No*) *P*(*Cool* | *No*) *P*(*High* | *No*) *P*(*Strong* | *No*) ≈ 0.0206
  - *So,* $v_{NB} = No$
  - *By normalization:*

    0.0206 /(0.0053 + 0.0206) ≈ 0.795

Naïve Bayes: Two classes

$$c_{NB} = \arg\max_{c_j \in C} P(c_j) \prod_k P(a_k \mid c_j)$$

- Naïve Bayes method gives a method for predicting .
- In the case of two classes, $c \in \{0,1\}$ we predict that $c=1$ iff:

$$\frac{\mathbf{P(c_j = 1) \bullet \prod_{k=1}^{n} P(a_k \mid c_j = 1)}}{\mathbf{P(c_j = 0) \bullet \prod_{k=1}^{n} P(a_k \mid c_j = 0)}} > \mathbf{1}$$

Naïve Bayes: Two classes

$$c_{NB} = \arg\max_{c_j \in C} P(c_j) \prod_k P(a_k \mid c_j)$$

Denote: $\mathbf{p_k = P(a_k = 1 \mid c_j = 1)}$, $\mathbf{q_k = P(a_k = 1 \mid c_j = 0)}$

Now,

$\mathbf{P(a_k = 1 \mid c_j = 1) = p_k}$, and $\mathbf{P(a_k = 0 \mid c_j = 1) = (1 - p_k)}$

$\mathbf{P(a_k = 1 \mid c_j = 0) = q_k}$, and $\mathbf{P(a_k = 0 \mid c_j = 0) = (1 - q_k)}$

So,

$$\frac{\mathbf{P(c_j = 1) \bullet \prod_{k=1}^{n} P(a_k \mid c_j = 1)}}{\mathbf{P(c_j = 0) \bullet \prod_{k=1}^{n} P(a_k \mid c_j = 0)}}$$

$$= \frac{\mathbf{P(c_j = 1) \bullet \prod_{k=1}^{n} p_k^{a_k} (1 - p_k)^{1-a_k}}}{\mathbf{P(c_j = 0) \bullet \prod_{k=1}^{n} q_k^{a_k} (1 - q_k)^{1-a_k}}} > \mathbf{1}$$

Naïve Bayes: Two classes

$$c_{NB} = \arg\max_{c_j \in C} P(c_j) \prod_k P(a_k \mid c_j)$$

$$\frac{P(c_j = 1) \bullet \prod_{k=1}^{n} p_k^{a_k} (1 - p_k)^{1-a_k}}{P(c_j = 0) \bullet \prod_{k=1}^{n} q_k^{a_k} (1 - q_k)^{1-a_k}} = \frac{P(c_j = 1) \bullet \prod_{k=1}^{n} (1 - p_k) \left(\frac{p_k}{1 - p_k}\right)^{a_k}}{P(c_j = 0) \bullet \prod_{k=1}^{n} (1 - q_k) \left(\frac{q_k}{1 - q_k}\right)^{a_k}} > 1$$

Take logarithm; we predict $c_j = 1$ iff :

$$\log \frac{P(c_j = 1)}{P(c_j = 0)} + \sum_k \log \frac{1 - p_k}{1 - q_k} + \sum_k \left(\log \frac{p_k}{1 - p_k} - \log \frac{q_k}{1 - q_k}\right) a_k > 0$$

So, the naive Bayes is a linear separator with

$$w_k = \log \frac{q_k}{1 - q_k} - \log \frac{p_k}{1 - p_k} = \log \left(\frac{q_k}{p_k} \frac{1 - p_k}{1 - q_k} \right)$$

if $p_k = q_k$ then $w_k = 0$ and the feature is irrelevant

Naïve Bayes: Two classes

- In the case of two classes we have that:

$$\log \frac{P(c_j = 1 \mid x)}{P(c_j = 0 \mid x)} = \sum_k w_k a_k - b \quad \Rightarrow \quad \frac{P(c_j = 1 \mid x)}{P(c_j = 0 \mid x)} = \exp\left(\sum_k w_k a_k - b\right)$$

- but since

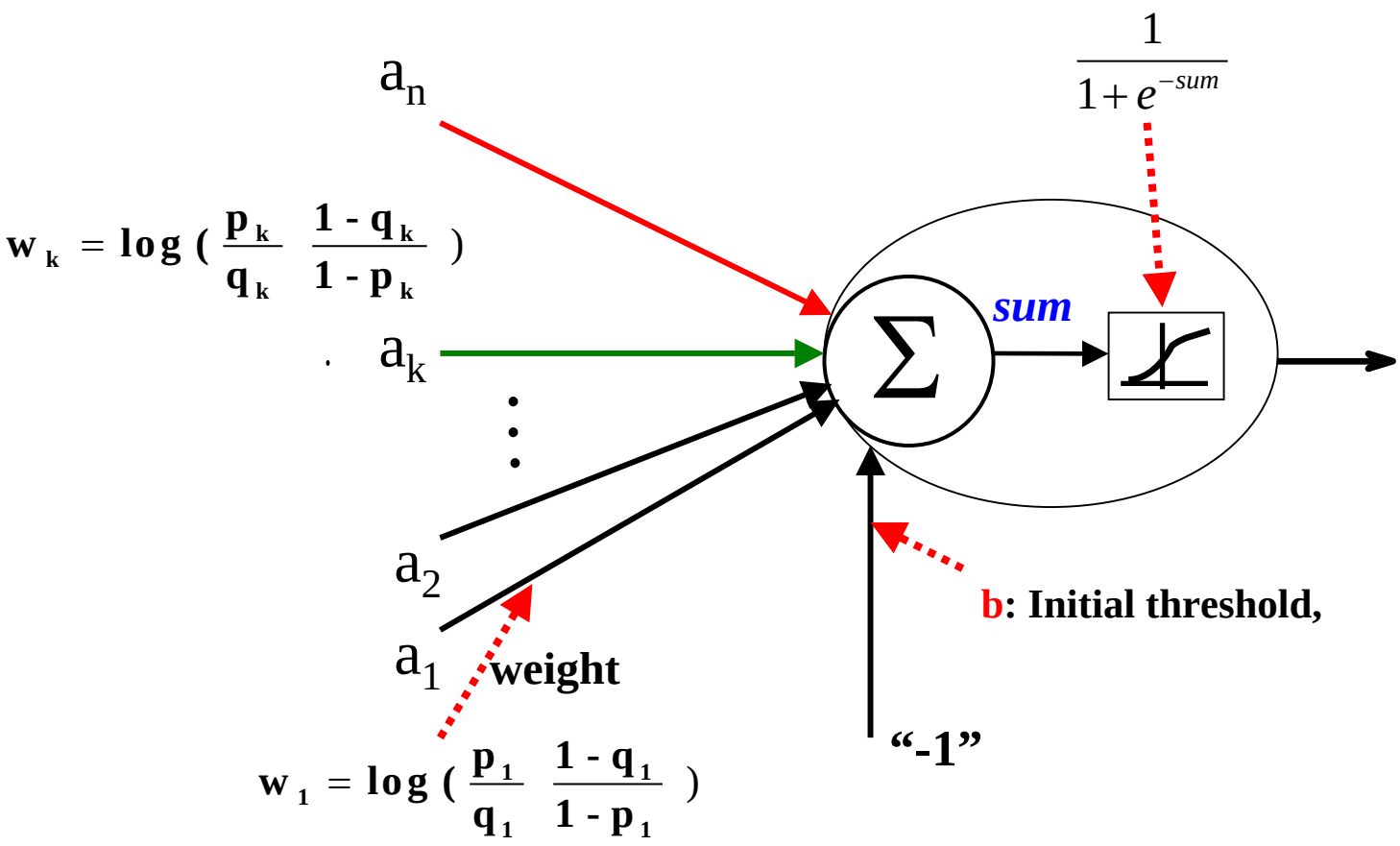
$$P(c_j = 0 \mid x) = 1 - P(c_j = 1 \mid x)$$

- Thus, we get:

$$P(c_j = 1 \mid x) = \frac{1}{1 + \exp(-\sum_k w_k a_k + b)}$$

- Which is simply the sigmoid function used in the neural network representation.

Naïve Bayes as a Perceptron



Naïve Bayes: Zero Probability

- If we never see something (0-time)in the train set:

$$P(a_k \mid c_j) = \frac{n_k}{n} = 0 \quad !$$

- How should we deal with them?
- We can smooth them by m-estimation:

m-estimate: $P(a_k \mid c_j) = \frac{n_k + mp}{n + m}$

where, p *is a prior distribute of* a_k ;

m *is* called equivalent sample size. It can be interpreted as augmenting the n acttual observations by an *additional* m *"virtual samples" distributed according to* p

Document Classification/ Categorization

Assign labels to each document or web-page:

- Labels are most often topics such as Sina-categories

  e.g., "体育," "财经," "教育"…
- Labels may be genres

  e.g., "editorials" "movie-reviews" "news"
- Labels may be opinion

  e.g., "like", "hate", "neutral"
- Labels may be domain-specific binary

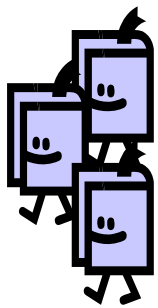
  e.g., "interesting-to-me" : "not-interesting-to-me"

  e.g., "spam" : "not-spam"

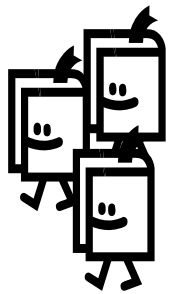
Learning to Classify Text

- Naïve Bayes has been used heavily on **text classification**.
- Instance space **D**: Text documents
- Text Categories (the set of targets): **C**
- Training Set: a set of examples (instances with label)
- How to classify a new text?
  - ✓ Document representation

Group A



Group B



Document representation

1. Attributes: Each word position is an attribute that takes the value corresponding to the word in that position. For example if a document has 100 words, then there are 100 attributes.

2. Attributes: Consider the number of words in the dictionary ~50,000. Consider each of these words an attribute and simply count how many times they appear in the document.

Attribute-1

- **Attributes:** One attribute for each word position;
- Number of attributes = L (length of longest document);
- Type of attribute = N (number of words);
- An instance: a list of length L;
- What are the probabilities needed to estimate ?

$$P(a_i = word_k \mid c_j) \quad \forall word_k$$

- Too many probabilities: (C x L x N ~ C x 100 x 50,000)
- New **assumption:** see the word in the document does not depend on its position

$$P(a_i = word_k \mid c) = P(a_m = word_k \mid c_j) \quad \forall i, m$$

- This reduces dimensionality to one attribute for each word.
- The number of probabilities to estimate is: C x N

Attribute-2

- **Attributes:** all the **N** words appearing in the training set; A document will be represented as a bag of its words;
- **Boolean:** An instance is a list of length N, 0-word is not in x; 1-word is in X;
- Problem: examples are too long
- List only the active features
- How many probabilities to estimate:
  - C x N

Estimating Probabilities

- **How to estimate** $P(w_k \mid c_j)$

$$P(w_k \mid c_j) = \frac{\text{Num of times word}_k \text{ occurs in training texts with label } c_j}{\text{Total number of times all words occurs in training texts with label } c_j} = \frac{n_k}{n}$$

- Sparsity of data is a problem
  - if n is small, the estimate is not accurate
  - if n_k is 0, it will dominate the estimate: we will never predict what if a word that never appeared in training set with label c_j but appears in the test data?

Smoothing

- There are many ways to smooth;
- An empirical issue.

Original: $P(w_k \mid c_j) = \frac{n_k}{n}$

m-estimate: $P(w_k \mid c_j) = \frac{n_k + mp}{n + m}$

if $mp = 1$ *and* $m = \mid Vocabulary \mid$, *we have*:

$$Laplace: \quad P(w_k \mid c_j) = \frac{n_k + 1}{n + \mid Vocabulary \mid}$$

Algorithm *Learn-Naïve-Bayes-Text (D, V)*

- 1. Collect all words, punctuation, and other tokens that occur in document set D
  - *Vocabulary* $\leftarrow$ {all distinct words, tokens occurring in any document $x \in D$}
- 2. Calculate required $P(c_j)$ and $P(x_i = w_k \mid c_j)$ probability terms
  - FOR each target value $c_j \in C$ DO
    - ***Docs* [*j*] $\leftarrow$ {documents *x* $\in$ *D* & *c*(*x*) = c_j }**

$$P(c_j) = \frac{|docs[j]|}{|D|}$$

    - ***Text* [*j*] $\leftarrow$ Concatenation (*docs* [*j*]) // form a single document**
    - ***n* $\leftarrow$ total number of distinct word positions in *text* [*j*]**
    - **FOR each word w_k in *Vocabulary***
      - » $n_k \leftarrow$ **number of times word w_k occurs in *text* [*j*]**

$$P(w_k \mid c_j) = \frac{n_k + 1}{n + |Vocabulary|}$$

- 3. RETURN $<\{P(c_j)\}, \{P(w_k \mid c_j)\}>$

Applying Naïve Bayes to Classify Text

- **Function *Classify-Naïve-Bayes-Text* (*x*, *Vocabulary*)**
  - Positions ← {word positions in document *x* that contain tokens found in Vocabulary}
  - RETURN $c_{NB} = \arg\max_{c_j \in C} P(c_j) \prod_{i \in Positions} P(a_i \mid c_j)$
- **Purpose of *Classify-Naïve-Bayes-Text***
  - Returns estimated target value for new document
  - a_i: denotes word found in the i^{th} position within *x*

Word Sense Disambiguation

- Problem: to look at the words around **an ambiguous word** in a large context **window**. Each content word contributes potentially useful information about which sense of the ambiguous word is likely to be used with it. The classifier combines the evidence from all features.
- The Naive Bayes is useful although the assumption is incorrect in the context of text processing:
  - The structure and linear ordering of words is ignored: bag of words model.
  - The presence of one word is independent of another, which is clearly untrue in text.

Word Sense Disambiguation

- Problem Definition
  - Given: *m* sentences, each containing a usage of a particular ambiguous word
  - Example: "The can will rust." (**auxiliary verb** versus **noun**)
  - Label: $c_j \equiv s \equiv$ correct word sense (e.g., $s \in$ {auxiliary verb, noun})
  - Representation: *m* examples (labeled attribute vectors $\langle (w_1, w_2, \ldots, w_n), s \rangle$)
  - Return: classifier $f: X \rightarrow C$ that disambiguates new $x \equiv (w_1, w_2, \ldots, w_n)$
- Solution Approach: Use Naïve Bayes

$$P(s \mid w_1, w_2, \ldots, w_n) = P(s) \prod_{i=1}^{n} P(w_i \mid s)$$

Topic Detection

- The task is to identify the most salient topic in a given document
- select a topic, t, from the set of possible topics, T;
- compute

$$v_{NB} = \arg\max_{t \in T} P(t) \prod_{i=1..N} P(w_i \mid t)$$

Comments on Naïve Bayes

- Tends to work well despite strong assumption of conditional independence.
- Experiments show it to be quite competitive with other classification methods.
- Although it does not produce accurate probability estimates when its independence assumptions are violated, it may still pick the correct maximum-probability class in many cases.
- Does not perform any search of the hypothesis space. Directly constructs a hypothesis from parameter estimates that are easily calculated from the training data.
- Not guarantee consistency with training data.
- Typically handles noise well since it does not even focus on completely fitting the training data.