

Sapienza University of Rome

Master in Artificial Intelligence and Robotics

Machine Learning

A.Y. 2024/2025

Prof. Luca Iocchi

Luca Iocchi

5. Bayesian Learning

1 / 55

Sapienza University of Rome, Italy - Machine Learning (2024/2025)

5. Bayesian Learning

Luca Iocchi

Outline

- Bayes Theorem
- MAP, ML hypotheses
- MAP learners
- Bayes optimal classifier
- Naive Bayes learner
- Example: Learning over text data

References

T. Mitchell. Machine Learning. Chapter 6

Two Roles for Bayesian Methods

Provides practical learning algorithms:

- Naive Bayes learning (examples affect prob. that a hypothesis is correct)
- Combine prior knowledge (prior probabilities) with observed data
- Make probabilistic predictions (new instances classified by weighted combination of multiple hypotheses)
- Requires prior probabilities (often estimated from available data)

Provides useful conceptual framework

- Provides “gold standard” for evaluating other learning algorithms

Basic Formulas for Probabilities

- *Product Rule*: probability of conjunction of A and B:

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

- *Sum Rule*: probability of disjunction of A and B:

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$

- *Theorem of total probability*: if events A_1, \dots, A_n are mutually exclusive with $\sum_{i=1}^n P(A_i) = 1$, then

$$P(B) = \sum_{i=1}^n P(B|A_i)P(A_i)$$

- *Bayes theorem*:

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

Classification as Probabilistic estimation

Given target function $f : X \rightarrow V$, dataset D and a new instance x' , best prediction $\hat{f}(x') = v^*$

$$v^* = \operatorname{argmax}_{v \in V} P(v|x', D)$$

ALL THE SET OF
POSSIBLE
CLASSES

More general formulation: given D and x' , compute the probability distribution over V

$$P(V|x', D)$$

Learning as Probabilistic estimation

Given dataset D and hypothesis space H , compute a probability distribution over H given D .

$$P(H|D)$$

Bayes rule

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

- $P(h)$ = prior probability of hypothesis h (prior)
- $P(D)$ = prior probability of training data D (normalization factor)
- $P(h|D)$ = probability of h given D (posterior)
- $P(D|h)$ = probability of D given h (likelihood)

MAP Hypotheses

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

Generally we want the most probable hypothesis h given D

Maximum a posteriori hypothesis h_{MAP} :

$$\begin{aligned} h_{MAP} &\equiv \operatorname{argmax}_{h \in H} P(h|D) = \operatorname{argmax}_{h \in H} \frac{P(D|h)P(h)}{P(D)} \\ &= \operatorname{argmax}_{h \in H} P(D|h)P(h) \end{aligned}$$

ML Hypotheses

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

If assume $P(h_i) = P(h_j)$, we can further simplify, and choose the *Maximum likelihood* (ML) hypothesis

$$h_{ML} = \operatorname{argmax}_{h \in H} P(D|h)$$

Brute Force MAP Hypothesis Learner

1. For each hypothesis h in H , calculate the posterior probability

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

2. Output the hypothesis h_{MAP} with the highest posterior probability

$$h_{MAP} = \operatorname{argmax}_{h \in H} P(h|D)$$

Most Probable Classification of New Instances

h_{MAP} : most probable *hypothesis* given data D .

Given a new instance x' , what is its most probable *classification* of x' ?

$h_{MAP}(x')$ may not be the most probable classification !!!

Most Probable Classification of New Instances

Consider:

- Three possible hypotheses h_1, h_2, h_3 :
 $P(h_1|D) = 0.4, P(h_2|D) = 0.3, P(h_3|D) = 0.3$
- Given a new instance x ,
 $h_1(x) = \oplus, h_2(x) = \ominus, h_3(x) = \ominus$
- What is the most probable classification of x ?

Bayes Optimal Classifier

Consider target function $f : X \mapsto V$, $V = \{v_1, \dots, v_k\}$, data set D and a new instance $x \notin D$:

IF WE TAKE THE ARGMAX OF THIS WE WILL HAVE THE BEST SOLUTIONS



$$P(v_j|x, D) = \sum_{h_i \in H} P(v_j|x, h_i, D)P(h_i|x, D) = \sum_{h_i \in H} P(v_j|x, h_i)P(h_i|D)$$

total probability over H

probability that $h_i(x) = v_j$ is independent from D given h_i

$$\Rightarrow P(v_j|x, h_i, D) = P(v_j|x, h_i)$$

$$h_i \text{ does not depend on } x \notin D \Rightarrow P(h_i|x, D) = P(h_i|D)$$

Bayes Optimal Classifier

Bayes Optimal Classifier

Class of a new instance x :

$$v_{OB} = \arg \max_{v_j \in V} \sum_{h_i \in H} P(v_j|x, h_i)P(h_i|D)$$

Bayes Optimal Classifier

Example:

$$P(h_1|D) = 0.4, \quad P(\ominus|x, h_1) = 0, \quad P(\oplus|x, h_1) = 1$$

$$P(h_2|D) = 0.3, \quad P(\ominus|x, h_2) = 1, \quad P(\oplus|x, h_2) = 0$$

$$P(h_3|D) = 0.3, \quad P(\ominus|x, h_3) = 1, \quad P(\oplus|x, h_3) = 0$$

therefore

$$\sum_{h_i \in H} P(\oplus|x, h_i)P(h_i|D) = 0.4$$

$$\sum_{h_i \in H} P(\ominus|x, h_i)P(h_i|D) = 0.6$$

and

$$v_{OB} = \arg \max_{v_j \in V} \sum_{h_i \in H} P(v_j|x, h_i)P(h_i|D) = \ominus$$

Bayes Optimal Classifier

Optimal learner: no other classification method using the same hypothesis space and same prior knowledge can outperform this method on average.

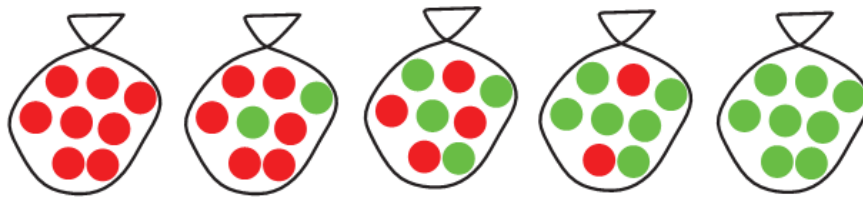
It maximizes the probability that the new instance x is classified correctly, i.e., $\arg\max_{v_j \in V} P(v_j|x, D)$.

Very powerful: labelling new instances x with $\arg\max_{v_j \in V} P(v_j|x, D)$ can correspond to none of the hypotheses in H .

Bayesian Learning Example

Five kinds of bags of candiers:

- ① 10% are h_1 : 100% cherry
- ② 20% are h_2 : 75% cherry, 25% lime
- ③ 40% are h_3 : 50% cherry, 50% lime
- ④ 20% are h_4 : 25% cherry, 75% lime
- ⑤ 10% are h_5 : 100% lime



Bayesian Learning Example

We choose a random bag (not knowing which type it is) and extract some candies from it.

What kind of bag is it? What is the probability of extracting a candy of a specific flavor next?

Bayesian Learning Example

Prior probability distribution:

$$P(H) = \langle 0.1, 0.2, 0.4, 0.2, 0.1 \rangle$$

Likelihood for lime candy:

$$P(I|H) = \langle 0, 0.25, 0.5, 0.75, 1 \rangle$$

Probability of extracting a lime candy (without data set):

$$\sum_{h_i} P(I|h_i)P(h_i) = 0 \cdot 0.1 + 0.25 \cdot 0.2 + 0.5 \cdot 0.4 + 0.75 \cdot 0.2 + 1 \cdot 0.1 = 0.5$$

Bayesian Learning Example

1. First candy is lime: $D_1 = \{I\}$

$$P(h_i|\{d_1\}) = \alpha P(\{d_1\}|h_i)P(h_i) \text{ (Bayes rule)}$$

$$\begin{aligned} P(H|D_1) &= \alpha \langle 0, 0.25, 0.5, 0.75, 1 \rangle \cdot \langle 0.1, 0.2, 0.4, 0.2, 0.1 \rangle \\ &= \alpha \langle 0, 0.05, 0.2, 0.15, 0.1 \rangle \\ &= \langle 0, 0.1, 0.4, 0.3, 0.2 \rangle \end{aligned}$$

$$\begin{aligned} 0 \cdot 0.1 &= 0 \\ 0.1 \cdot 0.25 &= 0.05 \\ &\vdots \end{aligned}$$

THAT
MEAN

Note: in these slides the operator \cdot represents element-wise product (not the dot product)

Bayesian Learning Example

2. Second candy is lime: $D_2 = \{I, I\}$

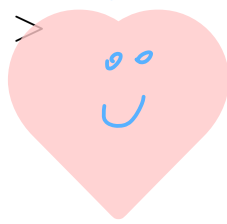
$$P(h_i|\{d_1, d_2\}) = \alpha P(\{d_1, d_2\}|h_i)P(h_i) \text{ (Bayes rule)}$$

$$= \alpha P(\{d_2\}|h_i) P(\{d_1\}|h_i)P(h_i) \text{ (independent data samples)}$$

$$\begin{aligned} P(H|D_2) &= \alpha < 0, 0.25, 0.5, 0.75, 1 > \cdot < 0, 0.1, 0.4, 0.3, 0.2 > \\ &= \alpha < 0, 0.025, 0.2, 0.225, 0.2 > \\ &= < 0, 0.038, 0.308, 0.346, 0.308 > \end{aligned}$$

PORTA NAPOLI!

da citta'
chio bell do
munni!



Bayesian Learning Example

3. Third candy is lime: $D_3 = \{I, I, I\}$

$$P(h_i|\{d_1, d_2, d_3\}) = \alpha P(\{d_1, d_2, d_3\}|h_i)P(h_i) \text{ (Bayes rule)}$$

$$= \alpha P(\{d_3\}|h_i) P(\{d_2\}|h_i) P(\{d_1\}|h_i)P(h_i) \text{ (independent data samples)}$$

$$\begin{aligned} P(H|D_3) &= \alpha < 0, 0.25, 0.5, 0.75, 1 > \cdot < 0, 0.038, 0.308, 0.346, 0.308 > \\ &= \alpha < 0, 0.01, 0.154, 0.260, 0.308 > \\ &= < 0, 0.013, 0.211, 0.355, 0.421 > \end{aligned}$$

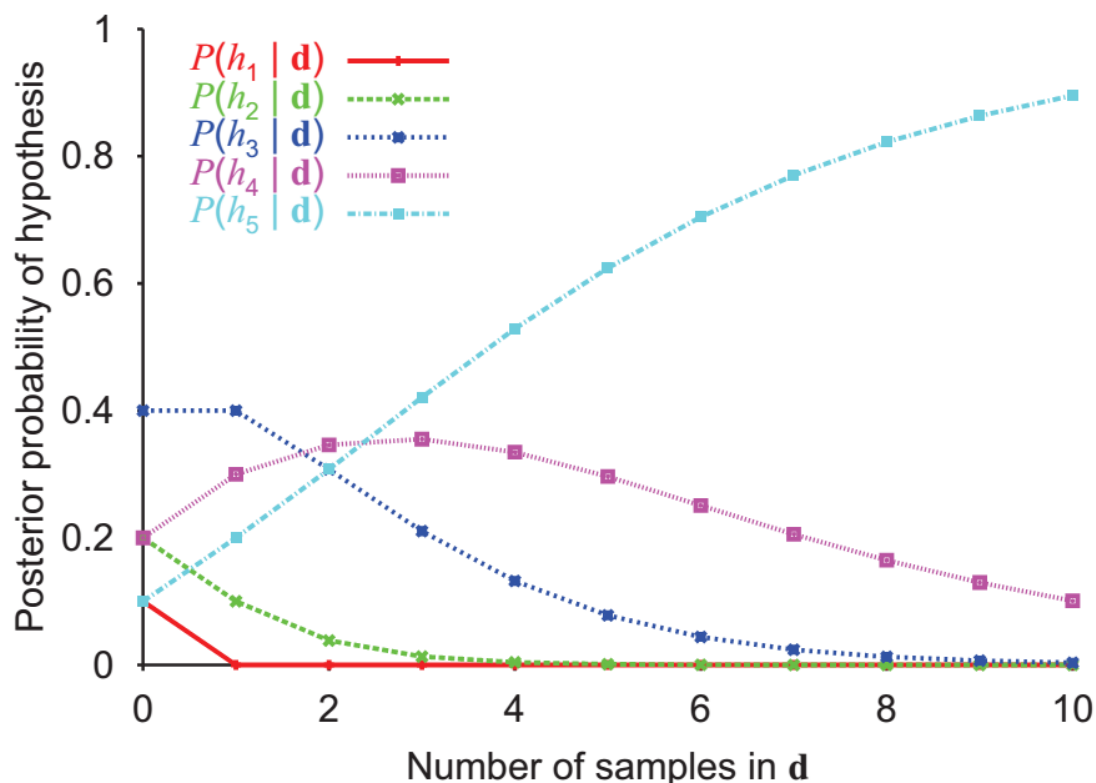
Bayesian Learning Example

What is probability of having another lime candy after $D_3 = \{l, l, l\}$?

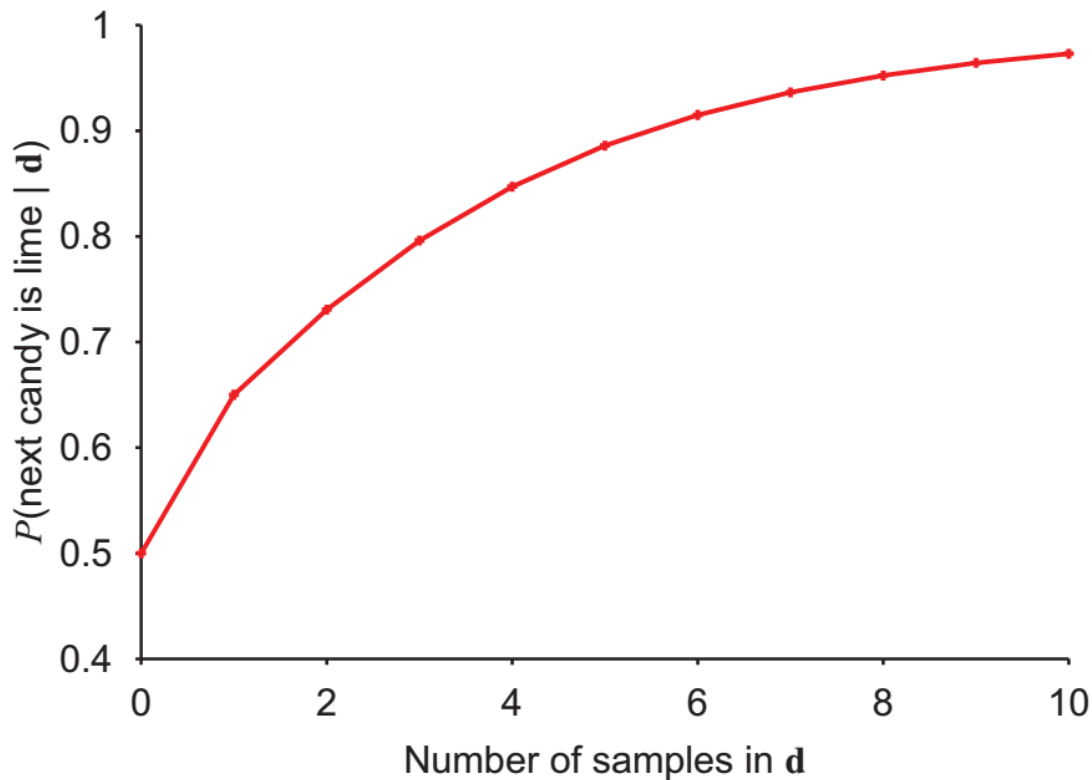
$$\begin{aligned}
 P(l|D_3) &= \sum_{h_i} P(l|h_i)P(h_i|D_3) \\
 &= 0 \cdot 0 + 0.25 \cdot 0.013 + 0.5 \cdot 0.211 + 0.75 \cdot 0.355 + 1 \cdot 0.421 \\
 &= 0.8
 \end{aligned}$$

Note: if we use $h_{MAP} = h_5$ to answer this question, the result will be 1, which is not correct!

Bayesian Learning Example



Bayesian Learning Example



Bayesian Learning Example 2

Consider a new manufacturer producing bags with an arbitrary choice of cherry/lime candies. $\theta \equiv \frac{\text{nr. of cherry candies}}{N} \in [0, 1]$.

Continuous space for hypotheses: h_θ

Data set: $D = \{c \text{ cherries}, l \text{ lime}\}$, $N = c + l$

$$P(c|h_\theta) = \theta$$

$$P(l|h_\theta) = 1 - \theta$$

- What is the ML hypothesis?

Bayesian Learning Example 2

$$h_{ML} = \underset{h_\theta}{\operatorname{argmax}} P(D|h_\theta) = \underset{h_\theta}{\operatorname{argmax}} L(D|h_\theta)$$

with $L(D|h_\theta) = \log P(D|h_\theta)$

$$P(D|h_\theta) = \prod_{j=1 \dots N} P(d_j|h_\theta) = \theta^c \cdot (1 - \theta)^l$$

$$L(D|h_\theta) = c \log \theta + l \log(1 - \theta)$$

$$\frac{dL(D|h_\theta)}{d\theta} = \frac{c}{\theta} - \frac{l}{1 - \theta} = 0 \Rightarrow \theta_{ML} = \frac{c}{c + l} = \frac{c}{N}$$

General approach

Given dataset $D = \{d_i\}$ with $d_i \in \{0, 1\}$,
assuming a probability distribution $P(d_i; \Theta)$

Maximum likelihood estimation

$$\Theta_{ML} = \underset{\Theta}{\operatorname{argmax}} \log P(D|\Theta)$$

Example: for Bernoulli distribution $P(X = k; \theta) = \theta^k (1 - \theta)^{1-k}$

$$\theta_{ML} = \dots = \frac{|\{d_i = 1\}|}{|D|}$$

Bernoulli distribution

Probability distribution of a binary random variable $X \in \{0, 1\}$

$$P(X = 1) = \theta \quad P(X = 0) = 1 - \theta$$

(e.g., observing head after flipping a coin, extracting a lime candy, ...).

$$P(X = k; \theta) = \theta^k (1 - \theta)^{1-k}$$

Multi-variate Bernoulli distribution

Joint probability distribution of a set of binary random variables X_1, \dots, X_n , each random variable following Bernoulli distribution

$$P(X_1 = k_1, \dots, X_n = k_n; \theta_1, \dots, \theta_n)$$

$$k_i \in \{0, 1\}$$

(e.g., observing head after flipping a coin **and** extracting a lime candy, ...).

Under the assumption that random variables X_i are mutually independent, Multi-variate Bernoulli distribution is the product of n Bernoulli distributions

$$P(X_1 = k_1, \dots; \theta_1, \dots, \theta_n) = \prod_{i=1}^n P(X_i = k_i; \theta_i) = \prod_{i=1}^n \theta_i^{k_i} (1 - \theta_i)^{1-k_i}$$

Binomial distribution

Probability distribution of k outcomes from n Bernoulli trials

$$P(X = k; n, \theta) = \binom{n}{k} \theta^k (1 - \theta)^{n-k}$$

(e.g., flipping a coin n times and observing k heads, extracting k lime candies after n extractions, ...).

Multinomial distribution

Generalization of binomial distribution for discrete valued random variables with d possible outcomes.

Probability distribution of k_1 outcomes for X_1 , ..., k_d outcomes for X_d , after n trials (with $\sum_{i=1\dots d} k_i = n$)

$$P(X_1 = k_1, \dots, X_d = k_d; n, \theta_1, \dots, \theta_d) = \frac{n!}{k_1! \dots k_d!} \theta_1^{k_1} \dots \theta_d^{k_d}$$

(e.g., rolling a d -sided die n times and observing k times a particular value, extracting k lime candies after n extractions from a bag containing d different flavors, ...).

Remarks

Probabilistic classification

$$\operatorname{argmax}_{v_j \in V} P(v_j | x, D)$$

- Bayes Optimal Classifier
provides best result, not practical when hypothesis space is large

Continuous model

- Maximum likelihood estimation
efficiently solved when analytical solutions are available

What are more practical and general solutions?

Naive Bayes Classifier

Naive Bayes Classifier uses conditional independence to approximate the solution.

X is *conditionally independent* of Y given Z

$$P(X, Y | Z) = P(X | Y, Z)P(Y | Z) = P(X | Z)P(Y | Z)$$

Naive Bayes Classifier

Assume target function $f : X \rightarrow V$, where each instance x is described by attributes $\langle a_1, a_2, \dots, a_n \rangle$.

Compute

$$\operatorname{argmax}_{v_j \in V} P(v_j | x, D) = \operatorname{argmax}_{v_j \in V} P(v_j | a_1, a_2 \dots a_n, D)$$

without explicit representation of hypotheses.

Naive Bayes Classifier

Given a data set D and a new instance $x = \langle a_1, a_2 \dots a_n \rangle$, most probable value of $f(x)$ is:

$$\begin{aligned} v_{MAP} &= \operatorname{argmax}_{v_j \in V} P(v_j | a_1, a_2 \dots a_n, D) \\ &= \operatorname{argmax}_{v_j \in V} \frac{P(a_1, a_2 \dots a_n | v_j, D) P(v_j | D)}{P(a_1, a_2 \dots a_n | D)} \\ &= \operatorname{argmax}_{v_j \in V} P(a_1, a_2 \dots a_n | v_j, D) P(v_j | D) \end{aligned}$$

(Bayes rule)

Naive Bayes Classifier

Naive Bayes assumption:

$$P(a_1, a_2, \dots, a_n | v_j, D) = \prod_i P(a_i | v_j, D)$$

Naive Bayes classifier

Class of new instance x :

$$v_{NB} = \operatorname{argmax}_{v_j \in V} P(v_j | D) \prod_i P(a_i | v_j, D)$$

Naive Bayes Algorithm

Target function $f : X \mapsto V$, $X = A_1 \times \dots \times A_n$, $V = \{v_1, \dots, v_k\}$,
data set D , new instance $x = \langle a_1, a_2 \dots a_n \rangle$.

Naive_Bayes_Learn(A, V, D)

for each target value $v_j \in V$

$\hat{P}(v_j | D) \leftarrow$ estimate $P(v_j | D)$

for each attribute A_k

for each attribute value $a_i \in A_k$

$\hat{P}(a_i | v_j, D) \leftarrow$ estimate $P(a_i | v_j, D)$

Classify_New_Instance(x)

$$v_{NB} = \operatorname{argmax}_{v_j \in V} \hat{P}(v_j | D) \prod_{a_i \in x} \hat{P}(a_i | v_j, D)$$

Naive Bayes estimation

$$\hat{P}(v_j|D) = \frac{|\{< \dots, v_j >\}|}{|D|}$$

$$\hat{P}(a_i|v_j, D) = \frac{|\{< \dots, a_i, \dots, v_j >\}|}{|\{< \dots, v_j >\}|}$$

Note: if none of the training instances with target value v_j have attribute value a_i , then $\hat{P}(a_i|v_j, D) = 0$ and thus $\hat{P}(v_j|D) \prod_i \hat{P}(a_i|v_j, D) = 0$

Naive Bayes estimation

Typical solution is Bayesian estimate with prior estimates

$$\hat{P}(a_i|v_j, D) = \frac{|\{< \dots, a_i, \dots, v_j >\}| + mp}{|\{< \dots, v_j >\}| + m}$$

where

- p is a prior estimate for $P(a_i|v_j, D)$
- m is a weight given to prior (i.e. number of “virtual” examples)

Example *PlayTennis*: Training data

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Naive Bayes: Example

Consider *PlayTennis* again, and new instance

$\langle \text{Outlook} = \text{sun}, \text{Temp} = \text{cool}, \text{Humid} = \text{high}, \text{Wind} = \text{strong} \rangle$

We want to compute:

$$v_{NB} = \operatorname{argmax}_{v_j \in V} P(v_j | D) \prod_i P(a_i | v_j, D)$$

without making any hypothesis space explicit.

Naive Bayes: Example

Note: easy notation with conditioning on D omitted.

$$P(\text{PlayTennis} = \text{yes}) = P(y) = 9/14 = 0.64$$

$$P(\text{PlayTennis} = \text{no}) = P(n) = 5/14 = 0.36$$

$$P(\text{Wind} = \text{strong}|y) = 3/9 = 0.33$$

$$P(\text{Wind} = \text{strong}|n) = 3/5 = 0.60$$

...

$$P(y) P(\text{sun}|y) P(\text{cool}|y) P(\text{high}|y) P(\text{strong}|y) = .005$$

$$P(n) P(\text{sun}|n) P(\text{cool}|n) P(\text{high}|n) P(\text{strong}|n) = .021$$

$$\rightarrow v_{NB} = n$$

Naive Bayes Remarks

Conditional independence assumption is often violated

$$P(a_1, \dots, a_n | v_j, D) \approx \prod_i P(a_i | v_j, D)$$

...but it works surprisingly well anyway.

Note: don't need estimated posteriors $\hat{P}(v_j | x, D)$ to be correct; need only that

$$\operatorname{argmax}_{v_j \in V} \hat{P}(v_j | D) \prod_i \hat{P}(a_i | v_j, D) = \operatorname{argmax}_{v_j \in V} P(v_j | D) P(a_1, \dots, a_n | v_j, D)$$

Issue: Naive Bayes posteriors often unrealistically close to 1 or 0

Learning to classify text

Input:

set of documents (sequences of words) $MyDocs \subset Docs$,
each classified as c_1, \dots, c_k

Learn target function $f : Docs \mapsto \{c_1, \dots, c_k\}$

Examples:

- spam classification (e-mail, SMS, ...)
- sentiment analysis (facebook/twitter posts, web reviews, ...)
- ...

Learning to Classify Text: Naive Bayes approach

Classification of documents $Docs$ in classes $C = \{c_1, \dots, c_k\}$

Target function $f : Docs \mapsto C$

Data set $D = \{ \langle doc, c \rangle_i \}$

Given a new document $doc \notin D$, compute

$$c_{NB} = \operatorname{argmax}_{c_j \in C} P(c_j | D) P(doc | c_j, D)$$

Learning to Classify Text: Naive Bayes approach

$$doc = w_1 w_2 \cdots w_m \quad (m = \text{length}(doc))$$

Naive Bayes conditional independence assumption

$$P(doc|c_j, D) = \prod_{i=1}^{\text{length}(doc)} P(p_i = w_i|c_j, D)$$

- $P(p_i = w_i|c_j)$: probability that, in document doc of class c_j , word in position i is w_i

One more assumption: $\forall i, k, P(p_i = w_i|c_j, D) = P(p_k = w_i|c_j, D)$, thus consider only:

- $P(w_i|c_j, D)$: probability that w_i occurs in document doc of class c_j

Word representations

Vocabulary $V = \{w_1, \dots, w_n\}$

set of all words appearing in any document $doc \in D$

$n = |V|$: size of vocabulary

Representation of doc : n -dimensional feature vector d , each feature d_i represents the encoding of word w_i in doc

Issues:

This representation loses context information (order of words important!)

Vectors are very large and sparse.

Word representations

Fix an (arbitrary) order on words in V : w_1, w_2, \dots, w_n

Representing $doc \in D$ as a fixed-length feature vector $d = \langle d_1, \dots, d_n \rangle$:

- ① boolean feature vector: $d_i = 1$ if w_i appears in doc , 0 otherwise
(Multivariate Bernoulli distribution) \rightarrow binary encoding
- ② ordinal feature vector: $d_i = k$ if word w_i occurs k times in doc
(Multinomial distribution) \rightarrow Bag of Words
- ③ real-valued feature vector (tf-idf): $d_i = tf(w_i, doc) idf(w_i, D)$
 \rightarrow term frequency-inverse document frequency (tf-idf)

Note that the meaning of $P(w_k|c_j, D)$ can be generalized to capture the *importance* of w_k to represent class c_j .

Multi-variate Bernoulli Naive Bayes distribution

For boolean feature vector $d = \langle d_1, \dots, d_n \rangle$ of generic $doc \in Docs$:

$$P(d|c_j, D) = \prod_{i=1}^n P(w_i|c_j, D)^{d_i} \cdot (1 - P(w_i|c_j, D))^{1-d_i}$$

Maximum-likelihood solution:

$$\hat{P}(w_i|c_j, D) = \frac{t_{i,j} + 1}{t_j + 2}$$

$t_{i,j}$: number of documents in D of class c_j containing word w_i

t_j : number of documents in D of class c_j

1, 2: parameters for Laplace smoothing

Multinomial Naive Bayes distribution

For ordinal feature vector $d = \langle d_1, \dots, d_n \rangle$ of generic $doc \in Docs$:

$$P(d|c_j, D) = \frac{n!}{d_1! \dots d_n!} \prod_{i=1}^n P(w_i|c_j, D)^{d_i}$$

Maximum-likelihood solution:

$$\hat{P}(w_i|c_j, D) = \frac{\sum_{doc \in D} tf_{i,j} + \alpha}{\sum_{doc \in D} tf_j + \alpha \cdot |V|}$$

$tf_{i,j}$: term frequency (# occurrences) of w_i in document doc of class c_j

tf_j : all-term frequency of document doc of class c_j

α : smoothing parameter ($\alpha = 1$ for Laplace smoothing)

Naive Bayes Text Classification algorithm

Estimate $\hat{P}(c_j)$ and $\hat{P}(w_i|c_j)$ using *Bernoulli distribution*.

LEARN_NAIVE_BAYES_TEXT_BE(D, C)

$V \leftarrow$ all distinct words in D

for each target value $c_j \in C$ do

$docs_j \leftarrow$ subset of D for which the target value is c_j

$t_j \leftarrow |docs_j|$: total number of documents in c_j

$\hat{P}(c_j) \leftarrow \frac{t_j}{|D|}$

for each word w_i in V do

$t_{i,j} \leftarrow$ number of documents in c_j containing word w_i

$\hat{P}(w_i|c_j) \leftarrow \frac{t_{i,j}+1}{t_j+2}$

Naive Bayes Text Classification algorithm

Estimate $\hat{P}(c_j)$ and $\hat{P}(w_i|c_j)$ using *Multinomial distribution*.

LEARN_NAIVE_BAYES_TEXT_MU(D, C)

$V \leftarrow$ all distinct words in D

for each target value $c_j \in C$ do

$docs_j \leftarrow$ subset of D for which the target value is c_j

$t_j \leftarrow |docs_j|$: total number of documents in c_j

$\hat{P}(c_j) \leftarrow \frac{t_j}{|D|}$

$TF_j \leftarrow$ total number of words in $docs_j$ (counting duplicates)

for each word w_i in V do

$TF_{i,j} \leftarrow$ total number of times word w_i occurs in $docs_j$

$\hat{P}(w_i|c_j) \leftarrow \frac{TF_{i,j}+1}{TF_j+|V|}$

Naive Bayes Text Classification algorithm

Use estimated $\hat{P}(c_j)$ and $\hat{P}(w_i|c_j)$ to classify a new document.

CLASSIFY_NAIVE_BAYES_TEXT(doc)

remove from doc all words not included in vocabulary V

return

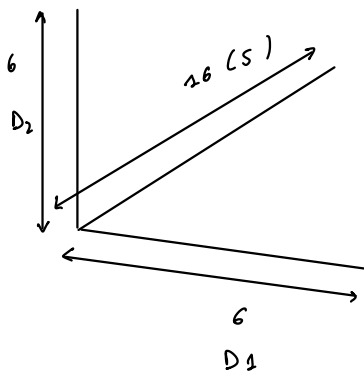
$$v_{NB} = \operatorname{argmax}_{c_j \in C} \hat{P}(c_j) \prod_{i=1}^{\text{length}(doc)} \hat{P}(w_i|c_j)$$

Text Classification improvements

- Stop words: remove from all the documents common words (“the”, “a”, etc.)
- Stemming: replace words with basic forms (“likes” → “like”, “liking” → “like”, etc.)
- Bi-gram, n-gram: token is a sequence of words
- ...

$$\left\{ \begin{aligned} a &= \ln \left(\frac{P(x|c_1)P(c_1)}{P(x|c_2)P(c_2)} \right) \\ \sigma(a) &= \frac{1}{1 + \exp(-a)} \end{aligned} \right. \Leftrightarrow \sigma(a) = \frac{1}{1 + \frac{1}{e^{\ln \left(\frac{P(x|c_1)P(c_1)}{P(x|c_2)P(c_2)} \right)}}} \Leftrightarrow \sigma(a) = \frac{1}{1 + \frac{P(x|c_2)P(c_2)}{P(x|c_1)P(c_1)}} \Leftrightarrow \frac{1}{\frac{P(x|c_2)P(c_2)}{P(x|c_1)P(c_1)} + 1}$$

$$P(12|1)$$



$$P(S|D_1 \wedge D_2) = \frac{P(D_1 \wedge D_2|S) P(D_1 \wedge D_2)}{P(S)}$$