

University of Rome “La Sapienza”

Master in Artificial Intelligence and Robotics

Machine Learning

A.Y. 2018/2019

Prof. Luca Iocchi

Sapienza University of Rome, Italy
Master in Artificial Intelligence and Robotics
Machine Learning (2018/19)

5. Bayesian Learning

Luca Iocchi

Outline

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

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 $P(A \wedge B)$ of a conjunction of two events A and B:

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

- *Sum Rule*: probability of a disjunction of two events 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

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

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

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} :

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

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

$$h_{ML} = \arg \max_{h \in H} P(D|h)$$

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 $x \notin D$:

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

total probability over H

$P(v_j|x, h_i)$: probability that $h_i(x) = v_j$ is independent from D given h_i

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

h_i 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.

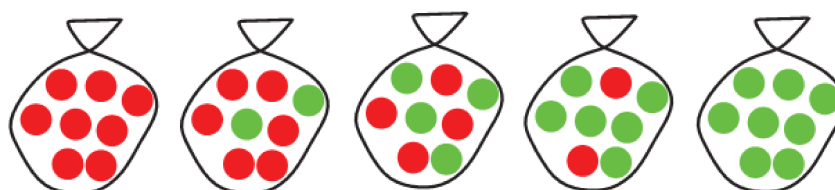
It maximizes the probability that the new instance x is classified correctly, i.e., $\operatorname{argmax}_{v_j \in V} P(v_j | x, D)$.

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

Bayesian Learning Example

Five kinds of bags of candies:

- 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 with 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: $D_1 = \{l\}$

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

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

Bayesian Learning Example

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

$$\begin{aligned} P(h_i|\{d_1, d_2\}) &= \alpha P(\{d_1, d_2\}|h_i)P(h_i) \text{ (Bayes rule)} \\ &= \alpha P(\{d_2\}|h_i) P(\{d_1\}|h_i)P(h_i) \text{ (independent data samples)} \end{aligned}$$

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

Bayesian Learning Example

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

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

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

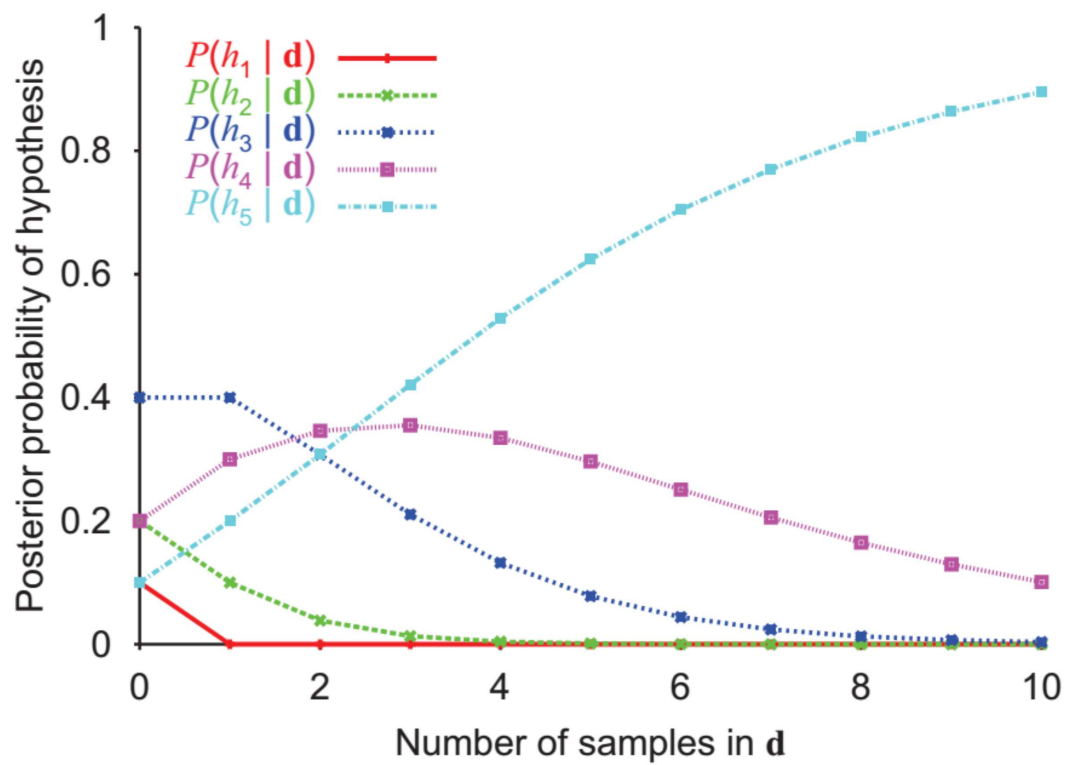
$$\begin{aligned} P(H|D_3) &= \alpha < 0, 0.25, 0.5, 0.75, 1 > \cdot < 0, 0.038, 0.308, 0.346, 0.308 > \\ &= \alpha < 0, 0.01, 0.154, 0.260, 0.308 > \\ &= < 0, 0.013, 0.211, 0.355, 0.421 > \end{aligned}$$

Bayesian Learning Example

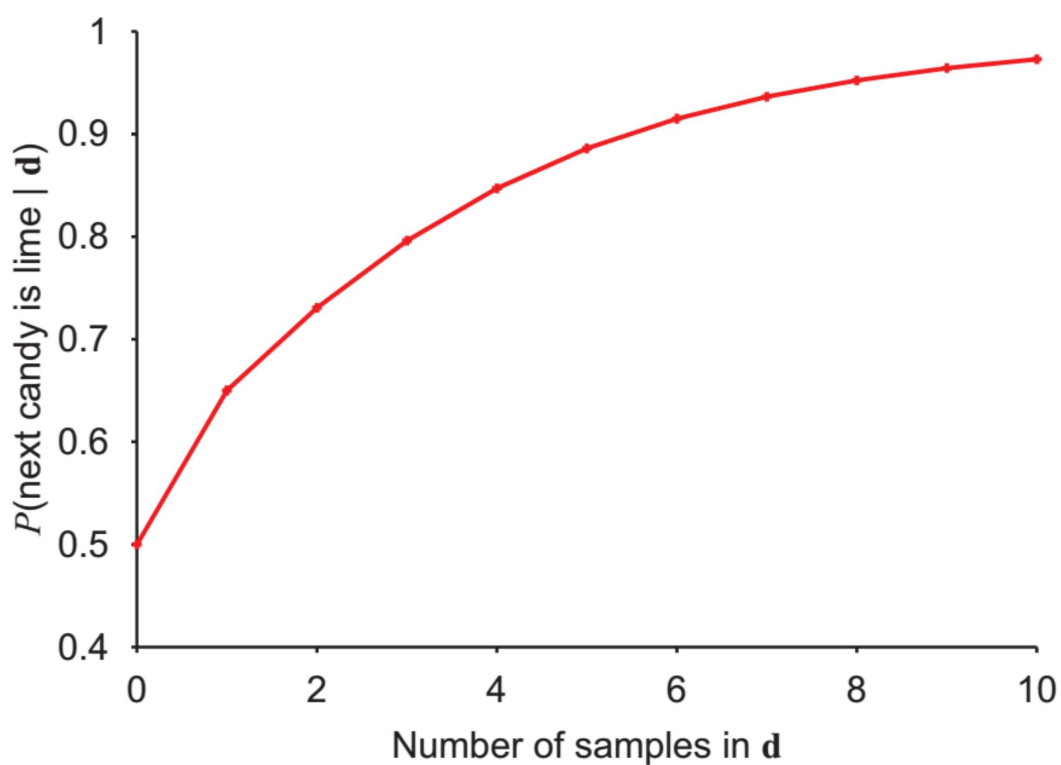
What is probability of having another lime candy after $D_3 = \{l, l, l\}$?

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

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: $\mathbf{d} = \{c \text{ cherries}, l \text{ lime}\}$, $N = c + l$

- What is the ML hypothesis?

Bayesian Learning Example 2

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

with $L(\mathbf{d}|h_\theta) = \log P(\mathbf{d}|h_\theta)$

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

$$L(\mathbf{d}|h_\theta) = c \log \theta + l \log(1 - \theta)$$

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

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, ...).

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

Given dataset $D = \{x_i\}$, maximum likelihood estimation

$$\theta_{ML} = \frac{|\{x_i = 1\}|}{|D|}$$

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

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

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

(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.

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 dice n times and observing k times a particular value, extracting k lime candies after n extractions from a bag containing d different flavors, ...).

Naive Bayes Classifier

Bayes optimal classifier provides best result, but it is not a practical method when hypothesis space is large.

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

$$\hat{P}(v_j|D) = \frac{|\{ \langle \dots, v_j \rangle \}|}{|D|}$$

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

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)

Naive Bayes: Example

Consider *PlayTennis* again, and new instance

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

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

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

...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)

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

Examples:

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

Bag of words representation

Vocabulary $V = \{w_k\}$: set of all the words appearing in any document of the data set.

$n = |V|$: size of the vocabulary

Bag of words representation of a text: n -dimensional feature vector

Note: BoW representation loses information (order of words in a text is important!)

Bag of words representation

Two options for representing each feature:

- ① boolean features: 1 if word appears in the text, 0 otherwise (multivariate Bernoulli distribution)
- ② ordinal features: number of occurrences of the words in the text (multinomial distribution)

Learning to Classify Text: Naive Bayes approach

Classification of documents $Docs$ in classes C .

Target function $f : Docs \mapsto C$, $C = \{c_1, \dots, c_k\}$

Data set $D = \{ \langle d_i, c_i \rangle \}$

Given a new document d_i , compute

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

Learning to Classify Text: Naive Bayes approach

Naive Bayes conditional independence assumption

$$P(d_i|c_j, D) = \prod_{i=1}^{length(d_i)} P(a_i = w_k|c_j, D)$$

where $P(a_i = w_k|c_j)$ is probability that word in position i is w_k , given c_j

one more assumption: $P(a_i = w_k|v_j, D) = P(a_m = w_k|v_j, D), \forall i, m$,
thus consider only $P(w_k|c_j, D)$.

Multi-variate Bernoulli Naive Bayes distribution

Feature vector for document d : n -dimensional vector 1 if word w_k appears in document d , 0 otherwise

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

$I(w_i \in d) = 1$ if $w_i \in d$, 0 otherwise

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

$t_{i,j}$: 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

Feature vector for document d : n -dimensional vector with number of occurrences of word w_i in document d

$$P(d|c_j, D) = Mu(d; n, \theta) = \dots$$

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

$tf_{i,j}$: term frequency (number of occurrences) of word w_i in document d of class c_j

tf_j : all term frequencies of document d 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(d)

remove from d all words not included in vocabulary V

return

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

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
- ...