Sapienza University of Rome

Master in Artificial Intelligence and Robotics
Master in Engineering in Computer Science

Machine Learning

A.Y. 2021/2022

Prof. Luca Iocchi

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 $A_1, \ldots, A_n$ 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*:

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

Sapienza University of Rome, Italy - Machine Learning (2021/2022)

Classification as Probabilistic estimation

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

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

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
- $P(D)$ = 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}:

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

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 $f : X \mapsto V$, $V = \{v_1, ..., v_k\}$, 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 $x \notin D \Rightarrow P(h_i|x, D) = P(h_i|D)$

Sapienza University of Rome, Italy - Machine Learning (2021/2022)

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:

$$
\begin{array}{lll}
P(h_1|D) = 0.4, & P(\ominus|x, h_1) = 0, & P(\oplus|x, h_1) = 1 \\
P(h_2|D) = 0.3, & P(\ominus|x, h_2) = 1, & P(\oplus|x, h_2) = 0 \\
P(h_3|D) = 0.3, & P(\ominus|x, h_3) = 1, & P(\oplus|x, h_3) = 0
\end{array}
$$

therefore

$$
\begin{aligned}
\sum_{h_i \in H} P(\oplus|x, h_i) P(h_i|D) &= 0.4 \\
\sum_{h_i \in H} P(\ominus|x, h_i) P(h_i|D) &= 0.6
\end{aligned}
$$

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.

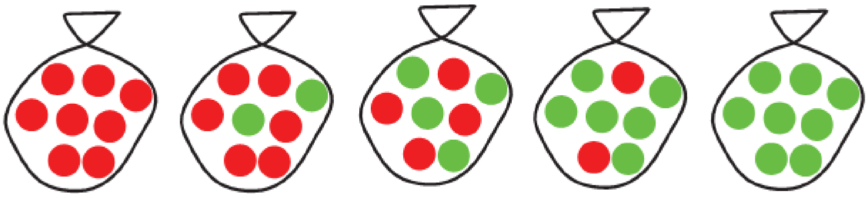
It maximizes the probability that the new instance x is classified correctly, i.e., $\text{argmax}_{v_j \in V} P(v_j|x, D)$.

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

Bayesian Learning Example

Five kinds of bags of 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:

$$\mathrm{P}(H) = < 0.1, 0.2, 0.4, 0.2, 0.1 >$$

Likelihood for lime candy:

$$\mathrm{P}(l|H) = < 0, 0.25, 0.5, 0.75, 1 >$$

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

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

Bayesian Learning Example

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

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

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

Bayesian Learning Example

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

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

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

Sapienza University of Rome, Italy - Machine Learning (2021/2022)

Bayesian Learning Example

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

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

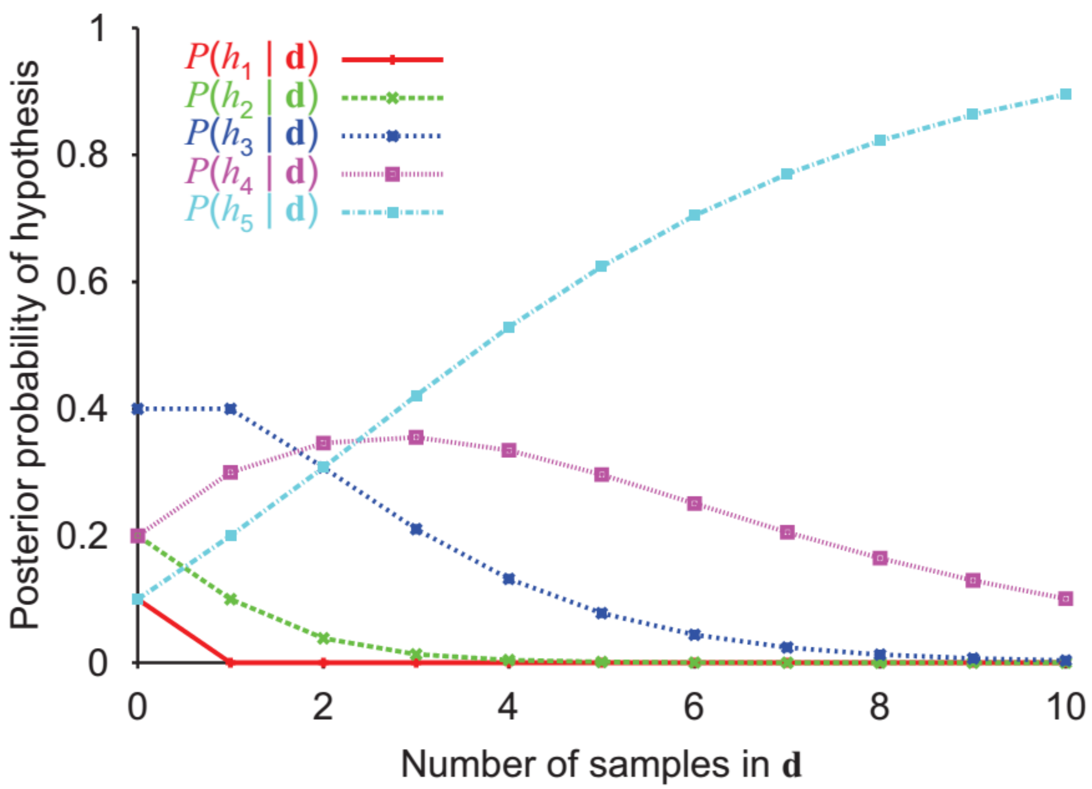
$$\begin{aligned} \mathrm{P}(H|D_3) &= \alpha < 0, 0.25, 0.5, 0.75, 1 > \cdot < 0, 0.038, 0.308, 0.346, 0.308 > \\ &= \alpha < 0, 0.01, 0.154, 0.260, 0.308 > \\ &= < 0, 0.013, 0.211, 0.355, 0.421 > \end{aligned}$$

Bayesian Learning Example

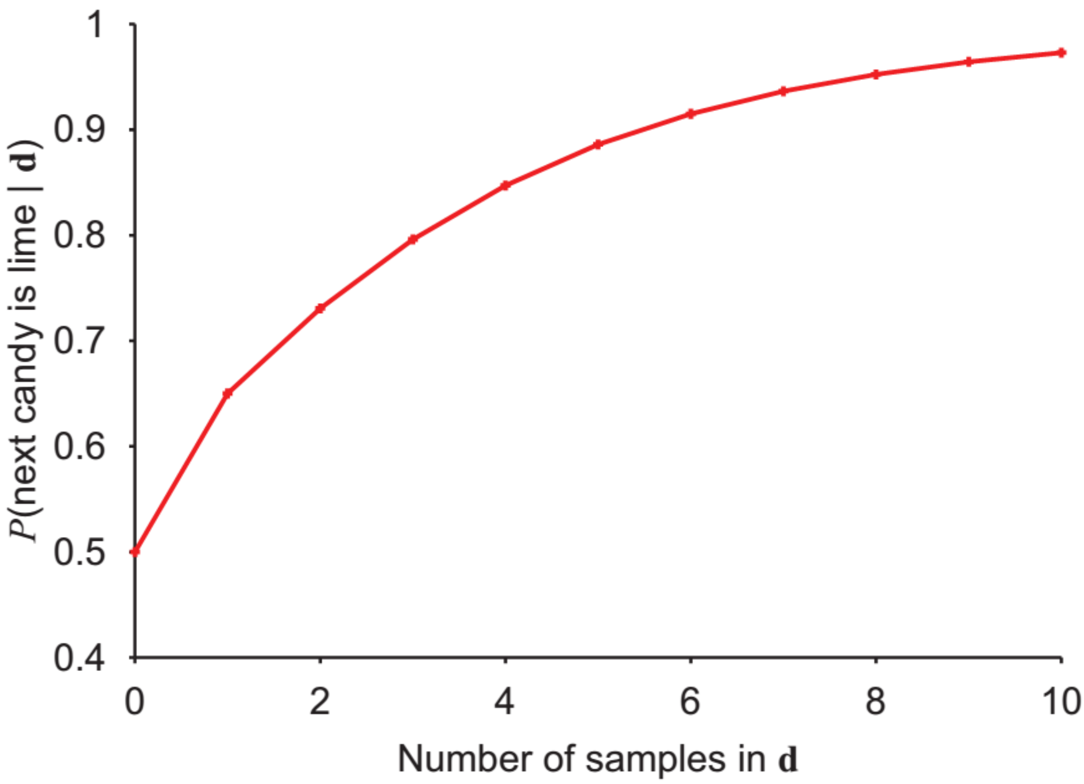
What is probability of having another lime candy after $D_3 = \{l, l, l\}$?

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

Bayesian Learning Example



Bayesian Learning Example



Bayesian Learning Example 2

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

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

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

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

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

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

General approach

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

Maximum likelihood estimation

$$\Theta_{ML} = \underset{\Theta}{\mathrm{argmax}} \log P(d_i|\Theta)$$

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

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

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 $X_1, \ldots X_n$, each random variable following Bernoulli distribution

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

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

(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

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

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 $\sum_{i=1 \ldots d} k_i = n$)

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

(e.g., rolling a d-sided die n times and observing k times a particular value, extracting k lime candies after n extractions form 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 $f : X \to V$, where each instance x is described by attributes $\langle a_1, a_2, \ldots, a_n \rangle$.

Compute

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

without explicit representation of hypotheses.

Naive Bayes Classifier

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

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

(Bayes rule)

Naive Bayes Classifier

Naive Bayes assumption:

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

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

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

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

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

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

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

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.

$$P(PlayTennis = yes) = P(y) = 9/14 = 0.64$$
$$P(PlayTennis = no) = P(n) = 5/14 = 0.36$$

$$P(Wind = strong|y) = 3/9 = 0.33$$
$$P(Wind = strong|n) = 3/5 = 0.60$$

$$\ldots$$

$$P(y)\ P(sun|y)\ P(cool|y)\ P(high|y)\ P(strong|y) = .005$$
$$P(n)\ P(sun|n)\ P(cool|n)\ P(high|n)\ P(strong|n) = .021$$

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

Naive Bayes Remarks

Conditional independence assumption is often violated

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

...but it works surprisingly well anyway.

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

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

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

Sapienza University of Rome, Italy - Machine Learning (2021/2022)

Learning to classify text

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

Learn target function $f : Docs \mapsto \{c1, \ldots, c_k\}$

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 $C = \{c_1, \ldots, c_k\}$

Target function $f : Docs \mapsto C$

Data set $D = \{< doc, c >_i\}$

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

$doc = w_1 w_2 \cdots w_m$ $(m = length(doc))$

Naive Bayes conditional independence assumption

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

- $P(p_i = w_i|c_j)$: 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

Bag of words representation

Vocabulary $V = \{w_1, \dots, w_n\}$
set of all words appearing in any document $doc \in MyDocs$

$n = |V|$: size of vocabulary

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

BoW representation looses information (order of words important!)

Bag of words representation

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

Two options for representing $doc \in Docs$ as a fixed-length feature vector $d =< d_1, \dots, d_n >$:

1. boolean feature vector: $d_i = 1$ if w_i appears in doc, 0 otherwise (Multivariate Bernoulli distribution)
2. ordinal feature vector: $d_i = k$ if word w_i occurs k times in doc (Multinomial distribution)

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 $d =< d_1, \ldots, d_n >$ 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 $d =< d_1, \ldots, d_n >$ of generic $doc \in Docs$:

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

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*.

$\textsc{Learn\_Naive\_Bayes\_text\_Be}(D, C)$

$V \leftarrow$ 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*.

$\textsc{Learn\_Naive\_Bayes\_text\_Mu}(D, C)$

$V \leftarrow$ 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.

$\textsc{Classify\_Naive\_Bayes\_text}(doc)$

remove from *doc* all words not included in vocabulary V

return

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

Text Classification improvements

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