

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

Interpretation of a Probability

- Frequentist
 - Relative frequency of an event occurring
 - What about rare events?
- Degree of belief
 - Our belief that the event will occur

About the Hypothesis Space $P(h)$

- Priors
 - Each h_i should be *Mutually exclusive*
 - Together the hypotheses must be *Totally exhaustive*
 - $\sum P(h_i) = 1$
 - Priors encode knowledge before we see the data

About the Data $P(D)$ and $P(D|H)$

- Data, $P(D)$
 - Data is considered to be a sample of all available data.
 - $P(D)$, probability the data will be observed given no knowledge of the hypothesis.
 - Constant for fixed data and if comparing hypotheses, can be ignored
- Likelihood, $P(D|h)$
 - Probability a hypothesis generated the observed data or probability of observing data given the hypothesis is true.
 - If the n instances are independent then
 - $P(D|h) = P(D_1|h) \cdot P(D_2|h) \dots P(D_n|h)$
 - Often use the Loglikelihood ($P(D|h)$).

Bayesian Posterior

- $P(h|D)$ is the posterior probability of the hypothesis (given the data).
- Usual aim of Bayesian learning is to find the MAP estimate
 - Most probable model in the model space
 - May be many highly probable models

A Simple Example

Does patient have cancer or not?

A patient takes a lab test and the result comes back positive. The test returns a correct positive result in only 98% of the cases in which the disease is actually present, and a correct negative result in only 97% of the cases in which the disease is not present. Furthermore, .008 of the entire population have this cancer.

$$\begin{array}{ll} P(\text{cancer}) = & P(\neg\text{cancer}) = \\ P(+|\text{cancer}) = & P(-|\text{cancer}) = \\ P(+|\neg\text{cancer}) = & P(-|\neg\text{cancer}) = \end{array}$$

Choosing the Hypothesis

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

Generally want the most probable hypothesis given the training data
Maximum a posteriori hypothesis h_{MAP} :

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

Assume all hypothesis have
equal probability

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

Basic Rules of Probability

- • *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

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

Brute Force MAP 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

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

Naïve Bayes Classifier

Along with decision trees, neural networks, nearest nbr, one of the most practical learning methods.

When to use

- Moderate or large training set available
- Attributes that describe instances are conditionally independent given classification

Successful applications:

- Diagnosis
- Classifying text documents

Read example on text classification in book.

Definition of NBC

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

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

Naive Bayes assumption:

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

which gives

$$\text{Naive Bayes classifier: } v_{NB} = \operatorname{argmax}_{v_j \in V} P(v_j) \prod_i P(a_i | v_j)$$

Coding the Algorithm

Naive_Bayes_Learn(*examples*)

For each target value v_j

$$\hat{P}(v_j) \leftarrow \text{estimate } P(v_j)$$

For each attribute value a_i of each attribute a

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

Classify_New_Instance(x)

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

NB: Simple Example

Consider *PlayTennis* again, and new instance

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

Want to compute:

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

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

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

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

NB: Subtleties

- Conditional independence assumption is often violated

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

- ...but it works surprisingly well anyway. Note don't need estimated posteriors $\hat{P}(v_j|x)$ correct; need only that

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

- see [Domingos & Pazzani, 1996] for analysis
- Naive Bayes posteriors often unrealistically close to 1 or 0

More On Optimality

The naive Bayesian classifier is known to be optimal when attributes are independent given the class. This project explores whether other sufficient conditions for its optimality exist. Empirical results showing that it performs surprisingly well in many domains containing clear attribute dependences suggest that the answer to this question may be positive. In this project we show that, although the Bayesian classifier's probability estimates are only optimal under quadratic loss if the independence assumption holds, the classifier itself can be optimal under zero-one loss (misclassification rate) even when this assumption is violated by a wide margin. The region of quadratic-loss optimality of the Bayesian classifier is in fact a second-order infinitesimal fraction of the region of zero-one optimality. This implies that the Bayesian classifier has a much greater range of applicability than previously thought. For example, naive Bayes is optimal for learning conjunctions and disjunctions, even though they violate the independence assumption. Further, studies in artificial domains show that it will often outperform more powerful classifiers for common training set sizes and numbers of attributes, even if its bias is "a priori" much less appropriate to the domain

<http://www.cs.washington.edu/ai/naive.html>

NB: Subtleties

2. what if none of the training instances with target value v_j have attribute value a_i ? Then

$$\hat{P}(a_i|v_j) = 0, \text{ and...}$$

$$\hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = 0$$

Typical solution is Bayesian estimate for $\hat{P}(a_i|v_j)$

$$\hat{P}(a_i|v_j) \leftarrow \frac{n_c + mp}{n + m}$$

where

- n is number of training examples for which $v = v_j$,
- n_c number of examples for which $v = v_j$ and $a = a_i$
- p is prior estimate for $\hat{P}(a_i|v_j)$
- m is weight given to prior (i.e. number of “virtual” examples)

Performance Issues

- No implicit search through model space
- Very fast to compute (just need to count)
- What type of probabilities are used to make predictions, how do these differ for decision trees?

Text Document Classification

Why?

- Learn which news articles are of interest
- Learn to classify web pages by topic

Naive Bayes is among most effective algorithms

What attributes shall we use to represent text documents??

Target concept *Interesting?*: *Document* $\rightarrow \{+, -\}$

1. Represent each document by vector of words
 - one attribute per word position in document
2. Learning: Use training examples to estimate
 - $P(+)$
 - $P(-)$
 - $P(doc|+)$
 - $P(doc|-)$

Naive Bayes conditional independence assumption

$$P(doc|v_j) = \prod_{i=1}^{length(doc)} P(a_i = w_k|v_j)$$

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

one more assumption: $P(a_i = w_k|v_j) = P(a_m = w_k|v_j), \forall i, m$

LEARN_NAIVE_BAYES_TEXT(*Examples*, *V*)

1. collect all words and other tokens that occur in *Examples*

- *Vocabulary* \leftarrow all distinct words and other tokens in *Examples*

2. calculate the required $P(v_j)$ and $P(w_k|v_j)$ probability terms

- For each target value v_j in *V* do

- $docs_j \leftarrow$ subset of *Examples* for which the target value is v_j

- $P(v_j) \leftarrow \frac{|docs_j|}{|Examples|}$

- $Text_j \leftarrow$ a single document created by concatenating all members of $docs_j$

- $n \leftarrow$ total number of words in $Text_j$ (counting duplicate words multiple times)

- for each word w_k in *Vocabulary*

- * $n_k \leftarrow$ number of times word w_k occurs in $Text_j$

- * $P(w_k|v_j) \leftarrow \frac{n_k+1}{n+|Vocabulary|}$

CLASSIFY_NAIVE_BAYES_TEXT(*Doc*)

- *positions* ← all word positions in *Doc* that contain tokens found in *Vocabulary*
- Return v_{NB} , where

$$v_{NB} = \operatorname{argmax}_{v_j \in V} P(v_j) \prod_{i \in \text{positions}} P(a_i | v_j)$$

Given 1000 training documents from each group

Learn to classify new documents according to which newsgroup it came from

comp.graphics	misc.forsale
comp.os.ms-windows.misc	rec.autos
comp.sys.ibm.pc.hardware	rec.motorcycles
comp.sys.mac.hardware	rec.sport.baseball
comp.windows.x	rec.sport.hockey