Introduction

Might have reasons (domain information) to favor some hypotheses/predictions over others a priori

Bayesian methods work with probabilities, and have two main roles:

- Provide practical learning algorithms:
  - Naïve Bayes learning
  - Bayesian belief network learning
  - Combine prior knowledge (prior probabilities) with observed data
  - Requires prior probabilities
- Provides useful conceptual framework
  - Provides “gold standard” for evaluating other learning algorithms
  - Additional insight into Occam’s razor

Outline

- Bayes Theorem
- Example
- Bayes optimal classifier
- Naïve Bayes classifier
- Example: Learning over text data
- Bayesian belief networks

Bayes Theorem

We want to know the probability that a particular label $r$ is correct given that we have seen data $D$

Conditional probability: $P(r \mid D) = P(r \cap D) / P(D)$

Bayes theorem:

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

- $P(r)$ is prior probability of label $r$ (might include domain information)
- $P(D)$ = probability of data $D$
- $P(r \mid D)$ = probability of $r$ given $D$
- $P(D \mid r)$ = probability of $D$ given $r$

Note: $P(r \mid D)$ increases with $P(D \mid r)$ and $P(r)$ and decreases with $P(D)$

Basic Formulas for Probabilities

- Product Rule: probability $P(A \cap B)$ of a conjunction of two events $A$ and $B$:
  $$P(A \cap B) = P(A \mid B) P(B) = P(B \mid A) P(A)$$
- Sum Rule: probability of a disjunction of two events $A$ and $B$:
  $$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$
- Theorem of total probability: if events $A_1, \ldots, A_n$ are mutually exclusive with $\sum_{i=1}^{n} P(A_i) = 1$, then
  $$P(B) = \sum_{i=1}^{n} P(B \mid A_i) P(A_i)$$

Example

Does a patient have cancer or not?

A patient takes a lab test and the result is positive. The test returns a correct positive result in 98% of the cases in which the disease is actually present, and a correct negative result in 97% of the cases in which the disease is not present. Furthermore, 0.008% of the entire population have this cancer.

Then

$$P(\text{cancer}) = P(\neg\text{cancer}) =$$
$$P(+ \mid \text{cancer}) = P(+) \mid \neg\text{cancer}) =$$

Now consider new patient for whom the test is positive. What is our diagnosis?

$$P(+ \mid \text{cancer}) P(\text{cancer}) =$$
$$P(+ \mid \neg\text{cancer}) P(\neg\text{cancer}) =$$

So diagnosis is
Bayes Optimal Classifier

Bayes rule lets us get a handle on the most probable label for an instance.

Bayes optimal classification of instance \( x \):

\[
\arg\max_{r_j \in R} P(r_j \mid x)
\]

where \( R \) is set of possible labels (e.g., \{+,-\}).

On average, no other classifier using same prior information
and same hypothesis space can outperform Bayes optimal!

\( \Rightarrow \) Gold standard for classification

Naïve Bayes Classifier

Problem: Estimating \( P(r_j) \) easily done, but there are exponentially many combinations of values of \( x_1, \ldots, x_n \).

E.g., if we want to estimate

\[
P(\text{Sunny, Hot, High, Weak} \mid \text{PlayTennis} = \text{No})
\]

from the data, need to count among the “No” labeled instances how many exactly match \( x \) (few or none)

Naïve Bayes assumption:

\[
P(x_1, x_2, \ldots, x_n \mid r_j) = \prod_i P(x_i \mid r_j)
\]

so naïve Bayes classifier:

\[
r_{NB} = \arg\max_{r_j \in R} P(r_j) \prod_i P(x_i \mid r_j)
\]

Now have only polynomial number of probs to estimate

Naïve Bayes Algorithm

Naïve_Bayes_Learn

- For each target value \( r_j \)
  - \( P(r_j) \) ← estimate \( P(r_j) \) = fraction of exs with \( r_j \)
  - For each attribute value \( v_k \) of each attr \( x_i \in x \)
    - \( P(v_k \mid r_j) \) ← estimate \( P(v_k \mid r_j) \) = fraction of \( r_j \)-labeled instances with \( v_k \)

Classify_New_Instance(x)

\[
r_{NB} = \arg\max_{r_j \in R} P(r_j) \prod_{x_i \in x} P(x_i \mid r_j)
\]

Bayes Optimal Classifier

Applying Bayes Rule

Let instance \( x \) be described by attributes \((x_1, x_2, \ldots, x_n)\)

Then, most probable label of \( x \) is:

\[
r^* = \arg\max_{r_j \in R} P(r_j \mid x_1, x_2, \ldots, x_n) = \arg\max_{r_j \in R} \frac{P(x_1, x_2, \ldots, x_n \mid r_j) P(r_j)}{P(x_1, x_2, \ldots, x_n)}
\]

In other words, if we can estimate \( P(r_j) \) and \( P(x_1, x_2, \ldots, x_n \mid r_j) \) for all possibilities, then we can give a Bayes optimal prediction of the label of \( x \) for all \( x \):

- How do we estimate \( P(r_j) \) from training data?
- What about \( P(x_1, x_2, \ldots, x_n \mid r_j) ? \)

Naïve Bayes Classifier (cont’d)

Along with decision trees, neural networks, nearest neighbor, SVMs, boosting, 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

Training Examples (Mitchell, Table 3.2):

<table>
<thead>
<tr>
<th></th>
<th>Outlook</th>
<th>Temp</th>
<th>Humid</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>W</td>
<td>No</td>
</tr>
<tr>
<td>02</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>03</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>04</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>05</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>06</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>07</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>08</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>09</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

Instance \( x \) to classify:

\( \text{Outlook} = \text{sun}, \text{Temp} = \text{cool}, \text{Humid} = \text{high}, \text{Wind} = \text{strong} \)
Naïve Bayes Example

Assign label $y_{NB} = \arg\max_{y \in \mathbb{Y}} P(r_j | y) = \prod_{i=1}^{n} P(x_i | r_j)$

$$P(y) \cdot P(sun | y) \cdot P(cool | y) \cdot P(high | y) \cdot P(strong | y)$$

$$= \left(\frac{9}{14}\right) \cdot \left(\frac{2}{9}\right) \cdot \left(\frac{3}{9}\right) \cdot \left(\frac{3}{9}\right) = 0.0053$$

$$P(n) \cdot P(sun | n) \cdot P(cool | n) \cdot P(high | n) \cdot P(strong | n)$$

$$= \left(\frac{5}{14}\right) \cdot \left(\frac{3}{5}\right) \cdot \left(\frac{1}{5}\right) \cdot \left(\frac{4}{5}\right) \cdot \left(\frac{3}{5}\right) = 0.0206$$

So $y_{NB} = n$

Conditional independence assumption is often violated, i.e.,

$$P(x_1, x_2, \ldots, x_n | r_j) \neq \prod_{i=1}^{n} P(x_i | r_j)$$

but it works surprisingly well anyway. Note that we don’t need estimated posteriors $P(r_j | x)$ to be correct; need only that

$$\arg\max_{r_j \in \mathbb{R}} P(r_j) \prod_{i} P(x_i | r_j) = \arg\max_{r_j \in \mathbb{R}} P(r_j) P(x_1, \ldots, x_n | r_j)$$

Sufficient conditions given in Domingos & Pazzani [1996]

Naïve Bayes Subtleties

What if none of the training instances with target value $r_j$ have attribute value $v_k$? Then

$$P(v_k | r_j) = 0, \text{ and } P(r_j) \prod_{i} P(v_k | r_j) = 0$$

Typical solution is to use $m$-estimate:

$$\hat{P}(v_k | r_j) = \frac{n_k + mp}{n + m}$$

where

- $n_k$ is number of training examples for which $r = r_j$, $v = v_k$
- $n$ is number of examples for which $r = r_j$, and $v = v_k$
- $p$ is prior estimate for $P(v_k | r_j)$
- $m$ is weight given to prior (i.e., number of “virtual” examples)
- Sometimes called pseudocounts

Bayesian Belief Networks

- Sometimes naïve Bayes assumption of conditional independence too restrictive
- But inferring probabilities is intractable without some such assumptions
- Bayesian belief networks (also called Bayes Nets) describe conditional independence among subsets of variables
- Allows combining prior knowledge about dependencies among variables with observed training data

Bayesian Application: Text Classification

- Target concept $\text{Spam?} : \text{Document} \rightarrow \{+, -, 0\}$
- Each document is a vector of words (one attribute per word position), e.g., $x_1 = “each”, x_2 = “document”,$ etc.
- Naïve Bayes very effective despite obvious violation of conditional independence assumption (⇒ words in an email are independent of those around them)
- Set $P(+) = \text{fraction of training emails that are spam, } P(–) = 1 – P(+)$
- To simplify matters, we will assume position independence, i.e., we only model the words in spam/not spam, not their position
  - For every word $w$ in our vocabulary, $P(w | +) = \text{probability that w appears in any position of +-labeled training emails (factoring in prior m-estimate)}$

Bayesian Networks

- Summary
- Predicting Labels
- Generative Models
- Definition
- Example
- Naïve Bayes Classifier
- The Bayesian Network

Pseudocode [Mitchell]

// Learn Naive Bayes (Example, V)
Examples is a set of text documents along with their target values, V is the set of all possible target values.

1. collect all words, punctuation, and other tokens that occur in Examples
   - Vocabulary = the set of all distinct words and other tokens occurring in any test document
2. calculate the required $P(w|\cdot)$ and $P(w|\cdot, \cdot, \cdot, \cdot, \cdot, \cdot)$ probability terms
   - For each target value $\eta_j$ in V do
     - $\text{doc}_{\eta_j}$ = the subset of documents from Examples for which the target value is $\eta_j$
     - $P(w|\cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot)$ = average document count by concatenating all members of $\text{doc}_{\eta_j}$
   - $\text{alldoc}_{\cdot}$ = total number of distinct word positions in Test
   - For each word $w$, in Vocabulary
     - $\text{doc}_{\eta_j, w}$ = number of times word $w$ occurs in $\text{doc}_{\eta_j}$
     - $P(w|\cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot) = \frac{\text{doc}_{\eta_j, w}}{\text{alldoc}_{\cdot}}$

// ClassifyNaiveBayesText(Doc)
// Naive Bayes Classifier

// NaiveBayes Classifier

// Bayes Net

Naïve Bayes Application: Text Classification

- Learning and Application
- Subtleties
- The Algorithm
- The Theorem
- Naive Bayes Classifier
- The Bayesian Network

Bayesian Belief Networks

- Sometimes naïve Bayes assumption of conditional independence too restrictive
- But inferring probabilities is intractable without some such assumptions
- Bayesian belief networks (also called Bayes Nets) describe conditional independence among subsets of variables
- Allows combining prior knowledge about dependencies among variables with observed training data
Bayesian Belief Networks

**Definition:** $X$ is conditionally independent of $Y$ given $Z$ if the probability distribution governing $X$ is independent of the value of $Y$ given the value of $Z$; that is, if

$$(\forall x_1, y_1, z_1) \ P(X=x_1 | Y=y_1, Z=z_1) = P(X=x_1 | Z=z_1)$$

more compactly, we write

$$P(X \mid Y, Z) = P(X \mid Z)$$

**Example:** Thunder is conditionally independent of Rain, given Lightning

$P(\text{Thunder} \mid \text{Rain, Lightning}) = P(\text{Thunder} \mid \text{Lightning})$

Naïve Bayes uses conditional independence and product rule to justify

$$P(X, Y \mid Z) = P(X \mid Y, Z) P(Y \mid Z) = P(X \mid Z) P(Y \mid Z)$$

Bayesian Belief Networks

**Definition**

- Directed acyclic graph
- Each node is asserted to be conditionally independent of its nondescendants, given its immediate predecessors
- Can think of edges in a Bayes net as representing a causal relationship between nodes
- We sometimes call Bayes nets generative (vs discriminative) models since they can be used to generate instances $(Y_1, \ldots, Y_n)$ according to joint distribution

Bayesian Belief Networks

**Special Case**

Since each node is conditionally independent of its nondescendants given its immediate predecessors, what model does this represent, given that $C$ is class and $x_i$s are attributes?

Bayesian Belief Networks

**Generative Models**

Represents joint probability distribution over variables $(Y_1, \ldots, Y_n)$, e.g., $P(\text{Storm, BusTourGroup, \ldots, ForestFire})$

- In general, for $y_i$ = value of $Y_i$

$$P(Y_1, \ldots, Y_n) = \prod_{i=1}^{n} P(y_i \mid \text{Parents}(Y_i))$$

where $\text{Parents}(Y_i)$ denotes immediate predecessors of $Y_i$ in graph

- E.g., $P(S \mid B, C, \neg L, \neg T, \neg F) = P(S) \cdot P(B) \cdot P(C \mid B, S) \cdot P(\neg L \mid S) \cdot P(\neg T \mid \neg L) \cdot P(\neg F \mid S, \neg L, \neg C)$

Bayesian Belief Networks

**Causality**

Can think of edges in a Bayes net as representing a causal relationship between nodes

- E.g., rain causes wet grass
- Probability of wet grass depends on whether there is rain

Bayesian Belief Networks

**Predicting Most Likely Label**

We sometimes call Bayes nets generative (vs discriminative) models since they can be used to generate instances $(Y_1, \ldots, Y_n)$ according to joint distribution

Can use for classification

- Label $r$ to predict is one of the variables, represented by a node
- If we can determine the most likely value of $r$ given the rest of the nodes, can predict label
- One idea: Go through all possible values of $r$, and compute joint distribution (previous slide) with that value and other attribute values, then return one that maximizes
Bayesian Belief Networks

E.g., if Storm (S) is the label to predict, and we are given values of B, C, ¬L, ¬T, and ¬F, can use formula to compute 
P(S, B, C, ¬L, ¬T, ¬F) and P(¬S, B, C, ¬L, ¬T, ¬F), then predict more likely one

Easily handles unspecified attribute values

Issue: Takes time exponential in number of values of unspecified attributes

More efficient approach: Pearl's message passing algorithm for chains and trees and polytrees (at most one path between any pair of nodes)

Learning of Bayesian Belief Networks

Several variants of this learning task

- Network structure might be known or unknown
- Training examples might provide values of all network variables, or just some

If structure known and all variables observed, then it's as easy as training a naïve Bayes classifier:

  - Initialize CPTs with pseudocounts
  - If, e.g., a training instance has set S, B, and ¬C, then increment that count in C’s table
  - Probability estimates come from normalizing counts

E.g., if E.g., observe ForestFire, Storm, BusTourGroup, Thunder, but not Lightning, Campfire

- Similar to training neural network with hidden units; in fact can learn network conditional probability tables using gradient ascent
- Converge to network h that (locally) maximizes \( P(D | h) \), i.e., maximum likelihood hypothesis
- Can also use EM (expectation maximization) algorithm
  - Use observations of variables to predict their values in cases when they're not observed
  - EM has many other applications, e.g., hidden Markov models (HMMs)

Learning of Bayesian Belief Networks

Suppose structure known, variables partially observable

- Combine prior knowledge with observed data
- Impact of prior knowledge (when correct!) is to lower the sample complexity
- Active research area
  - Extend from boolean to real-valued variables
  - Parameterized distributions instead of tables
  - More effective inference methods

Bayesian Belief Networks

Learning of Bayesian Belief Networks

Several variants of this learning task

- Network structure might be known or unknown
- Training examples might provide values of all network variables, or just some

If structure known and all variables observed, then it's as easy as training a naïve Bayes classifier:

  - Initialize CPTs with pseudocounts
  - If, e.g., a training instance has set S, B, and ¬C, then increment that count in C’s table
  - Probability estimates come from normalizing counts

E.g., if E.g., observe ForestFire, Storm, BusTourGroup, Thunder, but not Lightning, Campfire

- Similar to training neural network with hidden units; in fact can learn network conditional probability tables using gradient ascent
- Converge to network h that (locally) maximizes \( P(D | h) \), i.e., maximum likelihood hypothesis
- Can also use EM (expectation maximization) algorithm
  - Use observations of variables to predict their values in cases when they're not observed
  - EM has many other applications, e.g., hidden Markov models (HMMs)