

Machine Learning

Lecture 13

Bayesian Learning

(Based on Chapter 6 of Mitchell T.,
Machine Learning, 1997)

An Introduction

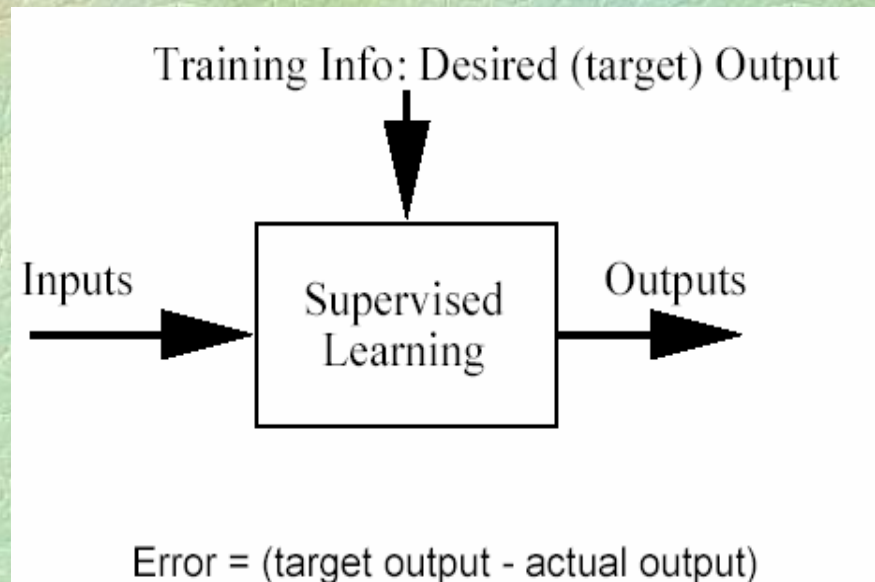
- *Bayesian Decision Theory* came long before Version Spaces, Decision Tree Learning and Neural Networks. It was studied in the field of Statistical Theory and more specifically, in the field of *Pattern Recognition*.
- Bayesian Decision Theory is at the basis of important learning schemes such as the *Naïve Bayes Classifier*, Learning *Bayesian Belief Networks* and the *EM Algorithm*.
- Bayesian Decision Theory is also useful as it provides a framework within which many non-Bayesian classifiers can be studied (See [Mitchell, Sections 6.3, 4,5,6]).

- Bayesian classification

- E.g. How to decide if a patient is ill or healthy, based on
 - A probabilistic model of the observed data
 - Prior knowledge

Classification problem

- Training data: examples of the form $(d, h(d))$
 - where d are the data objects to classify (inputs)
 - and $h(d)$ are the correct class info for d ,
 $h(d) \in \{1, \dots, K\}$
- Goal: given d_{new} , provide $h(d_{\text{new}})$



Bayes' Rule

$$p(h | d) = \frac{P(d | h)P(h)}{P(d)}$$

Understanding Bayes' rule

d = data

h = hypothesis (model)

- rearranging

$$p(h | d)P(d) = P(d | h)P(h)$$

$$P(d, h) = P(d, h)$$

the same joint probability

on both sides

Who is who in Bayes' rule

$P(h)$: prior belief (probability of hypothesis h before seeing any data)

$P(d | h)$: likelihood (probability of the data if the hypothesis h is true)

$P(d) = \sum_h P(d | h)P(h)$: data evidence (marginal probability of the data)

$P(h | d)$: posterior (probability of hypothesis h after having seen the data d)

Bayes Theorem

- **Goal:** To determine the most probable hypothesis, given the data D plus any initial knowledge about the prior probabilities of the various hypotheses in H .
- ***Prior probability of h , $P(h)$:*** it reflects any background knowledge we have about the chance that h is a correct hypothesis (before having observed the data).
- ***Prior probability of D , $P(D)$:*** it reflects the probability that training data D will be observed given no knowledge about which hypothesis h holds.
- ***Conditional Probability of observation D , $P(D/h)$:*** it denotes the probability of observing data D given some world in which hypothesis h holds.

Bayes Theorem (Cont'd)

- *Posterior probability of h , $P(h/D)$* : it represents the probability that h holds given the observed training data D . It reflects our confidence that h holds after we have seen the training data D and it is the quantity that Machine Learning researchers are interested in.
- *Bayes Theorem* allows us to compute $P(h/D)$:

$$P(h/D) = P(D/h)P(h)/P(D)$$

Does patient have cancer or not?

- A patient takes a lab test and the result comes back positive. It is known that the test returns a correct positive result in only 98% of the cases and a correct negative result in only 97% of the cases. Furthermore, only 0.008 of the entire population has this disease.
 - 1. What is the probability that this patient has cancer?
 - 2. What is the probability that he does not have cancer?
 - 3. What is the diagnosis?

hypothesis 1: 'cancer'
hypothesis 2: '¬cancer' } *hypothesis space H*
 - *data: '+'*

$$1. P(\text{cancer} \mid +) = \frac{P(+ \mid \text{cancer})P(\text{cancer})}{P(+)} = \frac{\dots\dots\dots}{\dots\dots\dots} = \dots\dots\dots$$

$$P(+ \mid \text{cancer}) = 0.98$$

$$P(\text{cancer}) = 0.008$$

$$\begin{aligned}
 P(+) &= P(+ \mid \text{cancer})P(\text{cancer}) + P(+ \mid \neg\text{cancer})P(\neg\text{cancer}) \\
 &= \dots\dots\dots
 \end{aligned}$$

$$P(+ \mid \neg\text{cancer}) = 0.03$$

$$P(\neg\text{cancer}) = \dots\dots\dots$$

$$2. P(\neg\text{cancer} \mid +) = \dots\dots\dots$$

3. *Diagnosis ??*

Choosing Hypotheses

- *Maximum Likelihood* hypothesis:

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

- Generally we want the most probable hypothesis given training data. This is the *Maximum A Posteriori* hypothesis:

$$h_{MAP} = \arg \max_{h \in H} P(h | d)$$

- Useful observation: it does not depend on the denominator $P(d)$

Now we compute the diagnosis

- To find the Maximum Likelihood hypothesis, we evaluate $P(d|h)$ for the data d , which is the positive lab test and chose the hypothesis (diagnosis) that maximises it:

$$P(+ | cancer) = \dots\dots\dots$$

$$P(+ | \neg cancer) = \dots\dots\dots$$

$$\Rightarrow \text{Diagnosis} : h_{ML} = \dots\dots\dots$$

- To find the Maximum A Posteriori hypothesis, we evaluate $P(d|h)P(h)$ for the data d , which is the positive lab test and chose the hypothesis (diagnosis) that maximises it. This is the same as choosing the hypotheses gives the higher posterior probability.

$$P(+ | cancer)P(cancer) = \dots\dots\dots$$

$$P(+ | \neg cancer)P(\neg cancer) = \dots\dots\dots$$

$$\Rightarrow \text{Diagnosis} : h_{MAP} = \dots\dots\dots$$

Maximum A Posteriori (MAP) Hypothesis and Maximum Likelihood

- **Goal:** To find the most probable hypothesis h from a set of candidate hypotheses H given the observed data D .
- **MAP Hypothesis, $h_{MAP} = \operatorname{argmax}_{h \in H} P(h/D)$**
$$= \operatorname{argmax}_{h \in H} P(D/h)P(h)/P(D)$$
$$= \operatorname{argmax}_{h \in H} P(D/h)P(h)$$
- If every hypothesis in H is equally probable a priori, we only need to consider the likelihood of the data D given h , $P(D/h)$. Then, h_{MAP} becomes the **Maximum Likelihood**,

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

Some Results from the Analysis of Learners in a Bayesian Framework

- If $P(h)=1/|H|$ and if $P(D|h)=1$ if D is consistent with h , and 0 otherwise, then every hypothesis in the version space resulting from D is a MAP hypothesis.
- Under certain assumptions regarding noise in the data, minimizing the mean squared error (what common neural nets do) corresponds to computing the maximum likelihood hypothesis.
- When using a certain representation for hypotheses, choosing the smallest hypotheses corresponds to choosing MAP hypotheses (An attempt at justifying Occam's razor)

Bayes Optimal Classifier

- One great advantage of Bayesian Decision Theory is that it gives us a lower bound on the classification error that can be obtained for a given problem.
- **Bayes Optimal Classification:** The most probable classification of a new instance is obtained by combining the predictions of all hypotheses, weighted by their posterior probabilities:

$$\mathit{argmax}_{v_j \in V} \sum_{h_i \in H} P(v_h/h_i)P(h_i/D)$$

where V is the set of all the values a classification can take and v_j is one possible such classification.

- Unfortunately, Bayes Optimal Classifier is usually too costly to apply! \implies *Naïve Bayes Classifier*

Naïve Bayes Classifier

- Let each instance x of a training set D be described by a conjunction of n attribute values $\langle a_1, a_2, \dots, a_n \rangle$ and let $f(x)$, the target function, be such that $f(x) \in V$, a finite set.

- Bayesian Approach:**

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

- Naïve Bayesian Approach:** We assume that the attribute values are conditionally independent so that $P(a_1, a_2, \dots, a_n | v_j) = \prod_i P(a_i | v_j)$ [and not too large a data set is required.]

Naïve Bayes Classifier:

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

Naïve Bayes Classifier

- What can we do if our data d has several attributes?
- Naïve Bayes assumption: Attributes that describe data instances are conditionally independent given the classification hypothesis

$$P(\mathbf{d} | h) = P(a_1, \dots, a_T | h) = \prod_t P(a_t | h)$$

- it is a simplifying assumption, obviously it may be violated in reality
- in spite of that, it works well in practice
- The Bayesian classifier that uses the Naïve Bayes assumption and computes the MAP hypothesis is called Naïve Bayes classifier
- One of the most practical learning methods
- Successful applications:
 - Medical Diagnosis
 - Text classification

Example. 'Play Tennis' data

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

Naïve Bayes solution

- *Classify any new datum instance $\mathbf{x}=(a_1, \dots, a_T)$ as:*

$$h_{\text{Naive Bayes}} = \arg \max_h P(h)P(\mathbf{x} | h) = \arg \max_h P(h) \prod_t P(a_t | h)$$

- To do this based on training examples, we need to estimate the parameters from the training examples:
 - For each target value (hypothesis) h

$$\hat{P}(h) := \text{estimate } P(h)$$

- For each attribute value a_t of each datum instance

$$\hat{P}(a_t | h) := \text{estimate } P(a_t | h)$$

- Based on the examples in the table, classify the following datum \mathbf{x} :

- $\mathbf{x}=(\text{Outl}=\text{Sunny}, \text{Temp}=\text{Cool}, \text{Hum}=\text{High}, \text{Wind}=\text{strong})$

- That means: Play tennis or not?

$$h_{NB} = \arg \max_{h \in \{\text{yes}, \text{no}\}} P(h)P(\mathbf{x} | h) = \arg \max_{h \in \{\text{yes}, \text{no}\}} P(h) \prod_t P(a_t | h)$$

$$= \arg \max_{h \in \{\text{yes}, \text{no}\}} P(h)P(\text{Outlook} = \text{sunny} | h)P(\text{Temp} = \text{cool} | h)P(\text{Humidity} = \text{high} | h)P(\text{Wind} = \text{strong} | h)$$

- Working:

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

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

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

$$P(\text{Wind} = \text{strong} | \text{PlayTennis} = \text{no}) = 3 / 5 = 0.60$$

etc.

$$P(\text{yes})P(\text{sunny} | \text{yes})P(\text{cool} | \text{yes})P(\text{high} | \text{yes})P(\text{strong} | \text{yes}) = 0.0053$$

$$P(\text{no})P(\text{sunny} | \text{no})P(\text{cool} | \text{no})P(\text{high} | \text{no})P(\text{strong} | \text{no}) = \mathbf{0.0206}$$

\Rightarrow answer : $\text{PlayTennis}(x) = \text{no}$

Bayesian Belief Networks

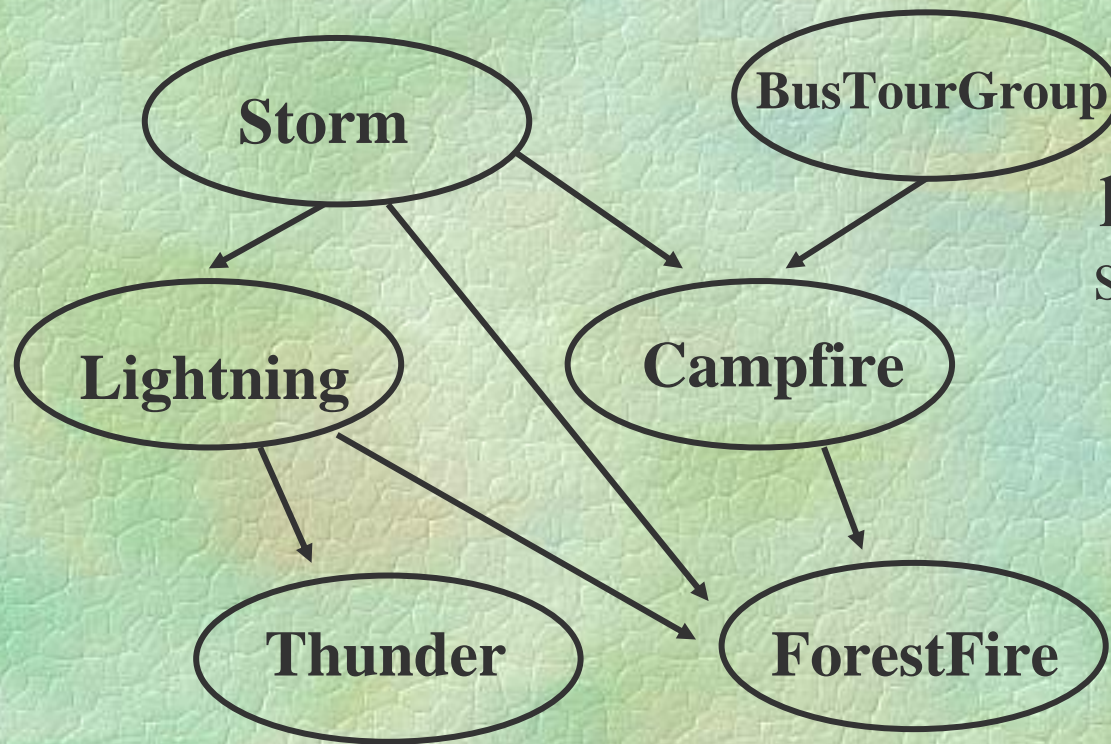
- The *Bayes Optimal Classifier* is often too costly to apply.
- The *Naïve Bayes Classifier* uses the conditional independence assumption to defray these costs. However, in many cases, such an assumption is overly restrictive.
- *Bayesian belief networks* provide an *intermediate* approach which allows stating conditional independence assumptions that apply to *subsets* of the variable.

Conditional Independence

- We say that X is *conditionally independent* of Y given Z if the probability distribution governing X is independent of the value of Y given a value for Z .
- i.e., $(\forall x_i, y_j, z_k) P(X=x_i/Y=y_j, Z=z_k) = P(X=x_i/Z=z_k)$
- or, $P(X/Y, Z) = P(X/Z)$
- This definition can be extended to sets of variables as well: we say that the set of variables $X_1 \dots X_l$ is conditionally independent of the set of variables $Y_1 \dots Y_m$ given the set of variables $Z_1 \dots Z_n$, if

$$P(X_1 \dots X_l / Y_1 \dots Y_m, Z_1 \dots Z_n) (= P(X_1 \dots X_l / Z_1 \dots Z_n))$$

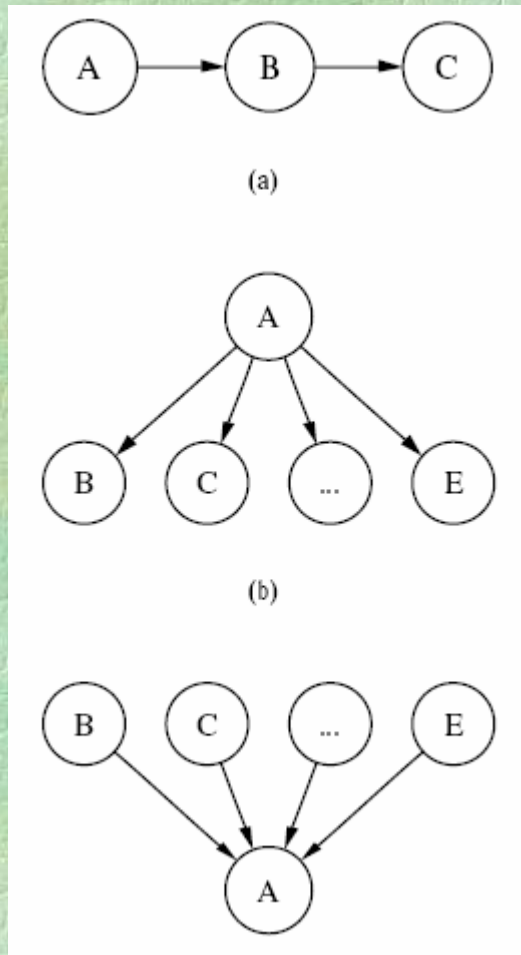
Representation in Bayesian Belief Networks



Associated with each node is a conditional probability table, which specifies the conditional distribution for the variable given its immediate parents in the graph

Each node is asserted to be conditionally independent of its non-descendants, given its immediate parents

Kinds of relations between variables in Bayesian nets



- a) Sequence, influence may be distributed from A to C and back while value of B is unknown
- b) Divergence, influence may be distributed on children of A while A is unknown
- c) Convergence, about A nothing unknown except that may be obtained through its parents

Inference in Bayesian Belief Networks

- A Bayesian Network can be used to compute the probability distribution for any subset of network variables given the values or distributions for any subset of the remaining variables.
- Unfortunately, exact inference of probabilities in general for an arbitrary Bayesian Network is known to be NP-hard.
- In theory, approximate techniques (such as Monte Carlo Methods) can also be NP-hard, though in practice, many such methods were shown to be useful.

Learning Bayesian Belief

Networks

3 Cases:

1. The network structure is given in advance and all the variables are fully observable in the training examples. ==> Trivial Case: just estimate the conditional probabilities.
2. The network structure is given in advance but only some of the variables are observable in the training data. ==> Similar to learning the weights for the hidden units of a Neural Net: Gradient Ascent Procedure
3. The network structure is not known in advance. ==> Use a heuristic search or constraint-based technique to search through potential structures.

The EM Algorithm: Learning with unobservable relevant variables.

- **Example:** Assume that data points have been uniformly generated from k distinct Gaussian with the same known variance. The problem is to output a hypothesis $h = \langle \mu_1, \mu_2, \dots, \mu_k \rangle$ that describes the means of each of the k distributions. In particular, we are looking for a maximum likelihood hypothesis for these means.
- We extend the problem description as follows: for each point x_i , there are k hidden variables z_{i1}, \dots, z_{ik} such that $z_{il} = 1$ if x_i was generated by normal distribution l and $z_{iq} = 0$ for all $q \neq l$.

The EM Algorithm (Cont'd)

- An arbitrary initial hypothesis $\mathbf{h} = \langle \mu_1, \mu_2, \dots, \mu_k \rangle$ is chosen.
- The EM Algorithm iterates over two steps:
 - *Step 1 (Estimation, E)*: Calculate the expected value $E[z_{ij}]$ of each hidden variable z_{ij} , assuming that the current hypothesis $\mathbf{h} = \langle \mu_1, \mu_2, \dots, \mu_k \rangle$ holds.
 - *Step 2 (Maximization, M)*: Calculate a new maximum likelihood hypothesis $\mathbf{h}' = \langle \mu_1', \mu_2', \dots, \mu_k' \rangle$, assuming the value taken on by each hidden variable z_{ij} is its expected value $E[z_{ij}]$ calculated in step 1. Then replace the hypothesis $\mathbf{h} = \langle \mu_1, \mu_2, \dots, \mu_k \rangle$ by the new hypothesis $\mathbf{h}' = \langle \mu_1', \mu_2', \dots, \mu_k' \rangle$ and iterate.

The EM Algorithm can be applied to more general problems