Machine Learning: Lecture 6

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]).

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):***

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

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,*** $h_{MAP} = \underset{h \in H}{argmax}\, P(h|D)$

$$= \underset{h \in H}{argmax}\, P(D|h)P(h)/P(D)$$

$$= \underset{h \in H}{argmax}\, P(D|h)P(h)$$

- 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***,

$$h_{ML} = \underset{h \in H}{argmax}\, P(D|h)P(h)$$

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:

$$\boldsymbol{argmax_{v_j \in V} \Sigma_{h_i \in H} P(v_h|h_i) P(h_i|D)}$$

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! ==> ***Naïve Bayes Classifier***

Naïve Bayes Classifier

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

$$v_{MAP} = \mathbf{argmax}_{vj \in V} P(v_j|a_1,a_2,..,a_n)$$
$$= \mathbf{argmax}_{vj \in V} [P(a_1,a_2,..,a_n|v_j)\ P(v_j)/P(a_1,a_2,..,a_n)]$$
$$= \mathbf{argmax}_{vj \in V} [P(a_1,a_2,..,a_n|v_j)\ P(v_j)$$

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

Naïve Bayes Classifier:

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

Bayesian Belief Networks

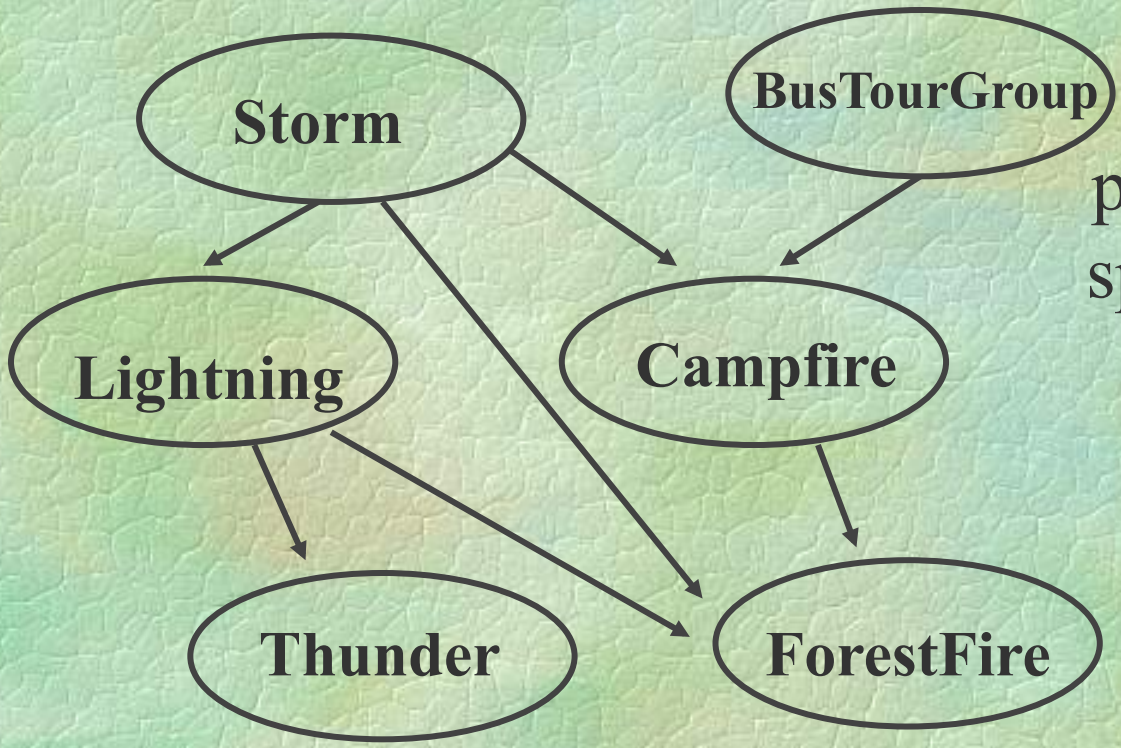
- The ***Bayes Optimal Classifier*** is often too costly to apply.
- The ***Naïve Bayes Classifier*** uses the conditional independence assumption to defray these costs. However, in many cases, such an assumption is overly restrictive.
- ***Bayesian belief networks*** provide an ***intermediate*** approach 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 $X_1 \ldots X_l$ is conditionally independent of the set of variables $Y_1 \ldots Y_m$ given the set of variables $Z_1 \ldots Z_n$, if

$$P(X_1 \ldots X_l|Y_1 \ldots Y_m,Z_1 \ldots Z_n(=P(X_1 \ldots X_l|Z_1 \ldots Z_n)$$

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

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 $h=<\mu_1, \mu_2 ..., \mu_k>$ 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 $z_{i1},..,z_{ik}$ 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 $h=<\mu_1, \mu_2 ,..., \mu_k>$ is chosen.
- The EM Algorithm iterates over two steps:
  - ***Step 1 (Estimation, E):*** Calculate the expected value E[zij] of each hidden variable zij, assuming that the current hypothesis $h=<\mu_1, \mu_2 ,..., \mu_k>$ holds.
  - ***Step 2 (Maximization, M):*** Calculate a new maximum likelihood hypothesis $h'=<\mu_1', \mu_2' ,..., \mu_k'>$, 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 $h=<\mu_1, \mu_2 ,..., \mu_k>$ by the new hypothesis $h'=<\mu_1', \mu_2' ,..., \mu_k'>$ and iterate.

The EM Algorithm can be applied to more general problems