Unit III

Part 2

Bayesian Learning

- Bayesian reasoning provides a probabilistic approach to inference. It is based on the assumption that the quantities of interest are governed by probability distributions and that optimal decisions can be made by reasoning about these probabilities together with observed data.
- Bayesian learning methods are relevant to our study of machine learning for two different reasons :
  - Bayesian learning algorithms that calculate explicit probabilities for hypotheses, such as the naive Bayes classifier, are among the most practical approaches to certain types of learning problems
  - Bayesian methods are important to our study of machine learning- they provide a useful perspective for understanding many learning algorithms that do not explicitly manipulate probabilities.

Features of Bayesian Learning

- Features of Bayesian learning methods include:
  - Each observed training example can incrementally decrease or increase the estimated probability that a hypothesis is correct.
    - This provides a more flexible approach to learning than algorithms that completely eliminate a hypothesis if it is found to be inconsistent with any single example.
  - Prior knowledge can be combined with observed data to determine the final probability of a hypothesis. In Bayesian learning, prior knowledge is provided by asserting
    - a prior probability for each candidate hypothesis, and
    - a probability distribution over observed data for each possible hypothesis.

Features of Bayesian Learning(Contd…)

- Bayesian methods can accommodate hypotheses that make probabilistic predictions
  - (e.g., hypotheses such as "this pneumonia patient has a 93% chance of complete recovery").
- New instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities.

Difficulties with Bayesian Methods

- Require initial knowledge of many probabilities
  - When these probabilities are not known in advance they are often estimated based on background knowledge, previously available data, and assumptions about the form of the underlying distributions.
- Significant computational cost is required to determine the Bayes optimal hypothesis in the general case (linear in the number of candidate hypotheses).
  - In certain specialized situations, this computational cost can be significantly reduced.

Bayes Theorem

- Bayes Theorem is named for English mathematician Thomas Bayes, who worked extensively in decision theory, the field of mathematics that involves probabilities.
- Bayes' theorem is a mathematical formula for determining conditional probability.
  - Conditional probability is the likelihood of an outcome occurring, based on a previous outcome occurring. Bayes' theorem provides a way to revise existing predictions or theories (update probabilities) given new or additional evidence.
- Bayes Theorem is also used widely in machine learning, where it is a simple, effective way to predict classes with precision and accuracy.
- The Bayesian method of calculating conditional probabilities is used in machine learning applications that involve classification tasks.
- A simplified version of the Bayes Theorem, known as the Naive Bayes Classification, is used to reduce computation time and costs.

Understanding Bayes Theorem.

- Bayes theorem provides a way to calculate the probability of a hypothesis based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself.
- Bayes' theorem can be used to determine the accuracy of medical test results by taking into consideration how likely any given person is to have a disease and the general accuracy of the test.
- Bayes' theorem relies on incorporating prior probability distributions in order to generate posterior probabilities.
- Prior probability, in Bayesian statistical inference, is the probability of an event before new data is collected.
- Posterior probability is the revised probability of an event occurring after taking into consideration new information. Posterior probability is calculated by updating the prior probability by using Bayes' theorem.
    - In statistical terms, the posterior probability is the probability of event A occurring given that event B has occurred.

Basic terms

- Consider bookstore manager has information about his customers' age and income. He wants to know how book sales are distributed across three age-classes of customers: youth (18-35), middle-aged (35-60), and seniors (60+).
- Let us term our data X. In Bayesian terminology, X is called evidence. We have some hypothesis H, where we have some X that belongs to a certain class C.
- Our goal is to determine the conditional probability of our hypothesis H given X, i.e., P(H | X).
- In simple terms, by determining P(H | X), we get the probability of X belonging to class C, given X.

- X has attributes of age and income – let’s say, for instance, 26 years old with an income of $2000.
- H is our hypothesis that the customer will buy the book.
  - Evidence – P(X) is known as evidence. It is simply the probability that the customer will, in this case, be of age 26, earning $2000.
  - Prior Probability – P(H), known as the prior probability, is the simple probability of our hypothesis – namely, that the customer will buy a book. This probability will not be provided with any extra input based on age and income. Since the calculation is done with lesser information, the result is less accurate.
  - Posterior Probability – P(H | X) is known as the posterior probability. Here, P(H | X) is the probability of the customer buying a book (H) given X (that he is 26 years old and earns $2000).
  - Likelihood – P(X | H) is the likelihood probability. In this case, given that we know the customer will buy the book, the likelihood probability is the probability that the customer is of age 26 and has an income of $2000.

- Given these, Bayes Theorem states:

  - P(h | D) = [P(D | h) * P(h)] / P(D)

  - P(h ID) increases with P(h) and with P(D |h) according to Bayes theorem. It is also reasonable to see that P(hl D) decreases as P(D) increases, because the more probable it is that D will be observed independent of h, the less evidence D provides in support of h.

Bayes Theorem - Example

- Sample Space for events A and B

A holds	**T**	**T**	**F**	**F**	**T**	**F**	**T**
B Holds	**T**	**F**	**T**	**F**	**T**	**F**	**F**

- P(A) = 4/7 P(B) = 3/ 7 P(B|A) = 2/4 P(A|B) = 2/3
- Is Bayes Theorem correct?
- P(B|A) = P(A|B)P(B) / P(A) = (2/3 * 3/7) / 4/7 = 2/4 - > CORRECT
- P(A|B) = P(B|A)P(A) / P(B) = (2/4 * 4/7) / 3/7 = 2/3 - > CORRECT

Maximum A Posteriori (MAP) Hypothesis, h_{MAP}

- The learner considers some set of candidate hypotheses H and is interested in finding the most probable hypothesis h ϵ H given the observed data D (or at least one of the maximally probable if there are several), such maximally probable hypothesis is called a maximum a posteriori (MAP) hypothesis.

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

- Notice in the final step above we dropped the term P(D) because it is a constant independent of h

Maximum Likelihood (ML) Hypothesis, h_{ML}

- If we assume that every hypothesis in H is equally probable a priori
  - i.e., P(hi) = P(hj) for all hi and hj in H.
- We only consider the term P(D|h) to find the most probable hypothesis.
- P(Dlh) is often called the likelihood of the data D given h
- And any hypothesis that maximizes P(D l h) is called a maximum likelihood (ML) hypothesis, $h_{ML.}$

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

Example - Does patient have cancer or not?

- 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 cancer.

P(cancer) = .008	P(notcancer) = .992
P(+|cancer) = .98	P(-|cancer) = .02
P(+|notcancer) = .03	P(-|notcancer) = .97

- A patient takes a lab test and the result comes back positive.

  P(+|cancer) P(cancer) = .98 * .008 = .0078

  P(+|notcancer) P(notcancer) = .03 * .992 = .0298 ->h_{MAP} is notcancer

- Since P(cancer|+) + P(notcancer|+) must be 1

  P(cancer|+) = .0078 / (.0078+.0298) = .21

  P(notcancer|+) = .0298 / (.0078+.0298) = .79

Brute-Force Bayes Concept Learning

- A Concept-Learning algorithm considers a finite hypothesis space $\mathbf{H}$
- defined over an instance space $\mathbf{X}$
- The task is to learn the target concept (a function) $\mathbf{c : X \rightarrow \{0,1\}.}$
- The learner gets a set of training examples $\mathbf{(<x_1,d_l> \ldots <x_m,d_m>)}$
- where $\mathbf{x_i}$ is an instance from $\mathbf{X}$ and $\mathbf{d_i}$ is its target value (i.e. $c(x_i) = d_i$).

- *Brute-Force Bayes Concept Learning Algorithm* finds the maximum a posteriori hypothesis (h_{MAP}), based on Bayes theorem.

Brute-Force MAP Learning Algorithm

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

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

- This algorithm may require significant computation, because it applies Bayes theorem to each hypothesis in H to calculate P(h|D) .
  - While this is impractical for large hypothesis spaces,
  - The algorithm is still of interest because it provides a standard against which we may judge the performance of other concept learning algorithms.

Brute-Force MAP Learning Algorithm

- BF MAP learning algorithm must specify values for P(h) and P(D|h).
- P(h) and P(D|h) must be chosen to be consistent with the assumptions:

1. The training data D is noise free (i.e., $d_i = c(x_i)$).
2. The target concept c is contained in the hypothesis space H
3. We have no a priori reason to believe that any hypothesis is more probable than any other.

- With these assumptions:

$$P(h) = \frac{1}{|H|} \quad \text{for all } h \text{ in } H$$

$$P(D|h) = \begin{cases} 1 \text{ if } d_i = h(x_i) \text{ for all } d_i \text{ in } D \\ 0 \text{ otherwise} \end{cases}$$

Brute-Force MAP Learning Algorithm

- So, the values of P(h|D) will be:

$$P(h|D) = \frac{0 \cdot P(h)}{P(D)} = 0 \text{ if } h \text{ is inconsistent with } D$$

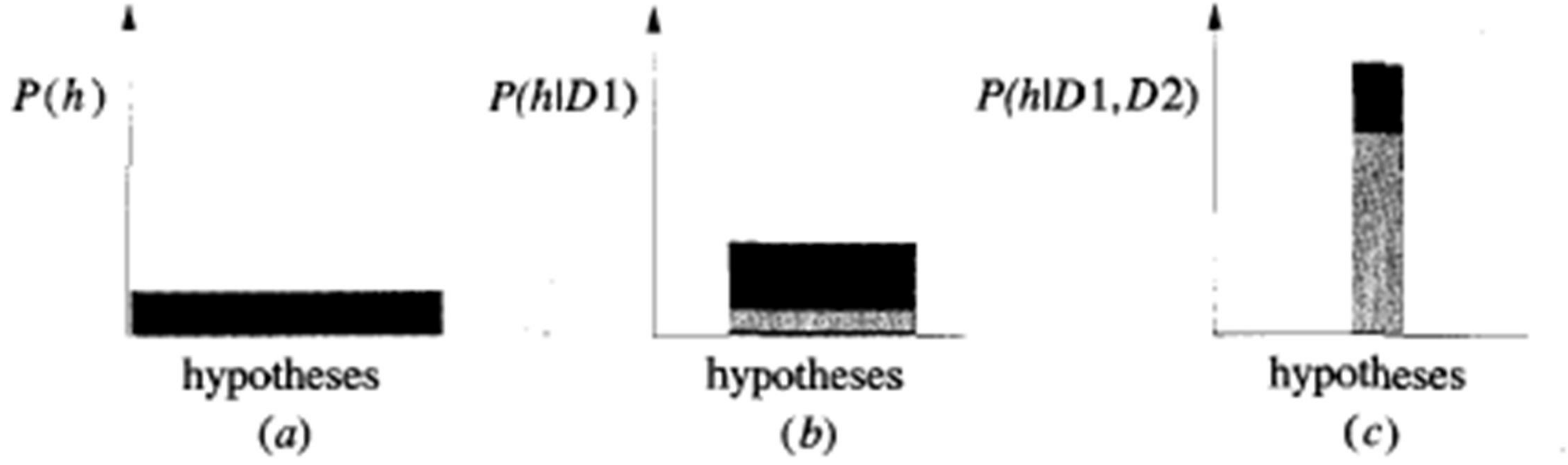
where $VS_{H,D}$ is the version space of H with respect to D.

$$P(h|D) = \frac{1 \cdot \frac{1}{|H|}}{P(D)} = \frac{1 \cdot \frac{1}{|H|}}{\frac{|VS_{H,D}|}{|H|}} = \frac{1}{|VS_{H,D}|} \text{ if } h \text{ is consistent with } D$$

- $P(D) = |VS_{H,D}| / |H|$ because
  - the sum over all hypotheses of P(h|D) must be one and the number of hypotheses from H consistent with D is $|VS_{H,D}|$, or
  - we can derive P(D) from *the theorem of total probability* and the fact that the hypotheses are mutually exclusive (i.e., $(\forall i \neq j)(P(h_i \wedge h_j) = 0)$

$$P(D) = \sum_{h_i \in H} P(D|h_i)P(h_i) = \sum_{h_i \in VS_{H,D}} 1 \cdot \frac{1}{|H|} + \sum_{h_i \notin VS_{H,D}} 0 \cdot \frac{1}{|H|} = \sum_{h_i \in VS_{H,D}} 1 \cdot \frac{1}{|H|} = \frac{|VS_{H,D}|}{|H|}$$

Evolution of posterior probabilities P(h|D) with increasing training



(a) Uniform priors assign equal probability to each hypothesis.
As training data increases first to Dl (b), then to Dl $\wedge$ D2 (c),

the posterior probability of inconsistent hypotheses becomes zero, while posterior probabilities increase for hypotheses remaining in the version space.

MAP Hypotheses and Consistent Learners

- A learning algorithm is a ***consistent learner*** if it outputs a hypothesis that commits zero errors over the training examples.
- Every consistent learner outputs a MAP hypothesis, if we assume
  - a uniform prior probability distribution over H (i.e., $P(h_i) = P(h_j)$ for all i, j), and
  - deterministic, noise free training data (i.e., $P(D|h) = 1$ if D and h are consistent, and 0 otherwise).
- Because FIND-S outputs a consistent hypothesis, it will output a MAP hypothesis under the probability distributions P(h) and P(D|h) defined above.
- Are there other probability distributions for P(h) and P(D|h) under which FIND-S outputs MAP hypotheses? Yes.
  - Because FIND-S outputs a maximally specific hypothesis from the version space, its output hypothesis will be a MAP hypothesis relative to any prior probability distribution that favors more specific hypotheses.
  - More precisely, suppose we have a probability distribution P(h) over H that assigns $P(h_1) \geq P(h_2)$ if h_l is more specific than h_2.

Bayes Optimal Classifier

- Normally we consider:
  - What is the most probable ***hypothesis*** given the training data?
- We can also consider:
  - what is the most probable ***classification*** of the new instance given the training data?

- Consider a hypothesis space containing three hypotheses, hl, h2, and h3.
  - Suppose that the posterior probabilities of these hypotheses given the training data are .4, .3, and .3 respectively.
  - Thus, hl is the MAP hypothesis.
  - Suppose a new instance x is encountered, which is classified positive by *hl*, but negative by *h2* and h3.
  - Taking all hypotheses into account, the probability that x is positive is .4 (the probability associated with *h1),* and the probability that it is negative is therefore .6.
  - The most probable classification (negative) in this case is different from the classification generated by the MAP hypothesis.

Bayes Optimal Classifier

- The most probable classification of the new instance is obtained by combining the predictions of all hypotheses, weighted by their posterior probabilities.
- If the possible classification of the new example can take on any value $\boldsymbol{v_j}$ from some set V, then the probability $P(v_j | D)$ that the correct classification for the new instance is $\boldsymbol{v_j}$:

$$P(v_j|D) = \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

- **Bayes optimal classification:**

$$\underset{v_j \in V}{\operatorname{argmax}} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

Bayes Optimal Classifier - Ex

$$P(h_1|D) = .4,\ P(\ominus|h_1) = 0,\ P(\oplus|h_1) = 1$$

$$P(h_2|D) = .3,\ P(\ominus|h_2) = 1,\ P(\oplus|h_2) = 0$$

$$P(h_3|D) = .3,\ P(\ominus|h_3) = 1,\ P(\oplus|h_3) = 0$$

Probabilities:

$$\sum_{h_i \in H} P(\oplus|h_i)P(h_i|D) = .4$$

$$\sum_{h_i \in H} P(\ominus|h_i)P(h_i|D) = .6$$

Result:

$$\underset{v_j \in \{\oplus,\ominus\}}{\text{argmax}} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D) = \ominus$$

Bayes Optimal Classifier

- Although the Bayes optimal classifier obtains the best performance that can be achieved from the given training data, it can be quite costly to apply.
  - The expense is due to the fact that it computes the posterior probability for every hypothesis in H and then combines the predictions of each hypothesis to classify each new instance.

- An alternative, less optimal method is the Gibbs algorithm:

1. Choose a hypothesis ***h*** from H at random, according to the posterior probability distribution over H.
2. Use ***h*** to predict the classification of the next instance ***x*.**

Naive Bayes Classifier

- One highly practical Bayesian learning method is Naive Bayes Learner (***Naive Bayes Classifier*).**
- The naive Bayes classifier applies to learning tasks where each instance ***x*** is described by a conjunction of attribute values and where the target function f (x) can take on any value from some finite set V.
- A set of training examples is provided, and a new instance is presented, described by the tuple of attribute values ***(a_l, a_2 ...a_n).***
- The learner is asked to predict the target value (classification), for this new instance.

Naive Bayes Classifier

- The Bayesian approach to classifying the new instance is to assign the most probable target value v_{MAP}, given the attribute values (a_l, a_2 ... a_n) that describe the instance.

$$v_{MAP} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j | a_1, a_2 \ldots a_n)$$

- By Bayes theorem:

$$v_{MAP} = \underset{v_j \in V}{\operatorname{argmax}} \frac{P(a_1, a_2 \ldots a_n | v_j) P(v_j)}{P(a_1, a_2 \ldots a_n)}$$

$$= \underset{v_j \in V}{\operatorname{argmax}} P(a_1, a_2 \ldots a_n | v_j) P(v_j)$$

Naive Bayes Classifier

- It is easy to estimate each of the **$P(v_j)$** simply by counting the frequency with which each target value **v_j** occurs in the training data.
- However, estimating the different **$P(a_1,a_2 \ldots a_n | v_j)$** terms is not feasible unless we have a very, very large set of training data.
  - The problem is that the number of these terms is equal to the number of possible instances times the number of possible target values.
  - Therefore, we need to see every instance in the instance space many times in order to obtain reliable estimates.
- The naive Bayes classifier is based on the simplifying assumption that the attribute values are conditionally independent given the target value.
- For a given the target value of the instance, the probability of observing conjunction ***$a_1,a_2 ... a_n$***, is just the product of the probabilities for the individual attributes:

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

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

Naive Bayes Classifier - Ex

Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Naive Bayes Classifier -Ex

- New instance to classify:

  (Outlook=sunny, Temperature=cool, Humidity=high, Wind=strong)
- Our task is to predict the target value (yes or no) of the target concept *PlayTennis* for this new instance.

$$v_{NB} = \underset{v_j \in \{yes, no\}}{\operatorname{argmax}} P(v_j) \prod_i P(a_i|v_j)$$

$$= \underset{v_j \in \{yes, no\}}{\operatorname{argmax}} P(v_j) \ \mathbf{P(Outlook=sunny|v_j)\ P(Temperature=cool|v_j)\ P(Humidity=high|v_j)\ P(Wind=strong|v_j)}$$

Naive Bayes Classifier -Ex

- P(P1ayTennis = yes) = 9/14 = .64
- P(P1ayTennis = no) = 5/14 = .36

$$P(yes)\ P(sunny|yes)\ P(cool|yes)\ P(high|yes)\ P(strong|yes) = .0053$$

$$P(no)\ P(sunny|no)\ P(cool|no)\ P(high|no)\ P(strong|no) = .0206$$

□ Thus, the naive Bayes classifier assigns the target value ***PlayTennis*** = ***no*** to this new instance, based on the probability estimates learned from the training data.

- Furthermore, by normalizing the above quantities to sum to one we can calculate the conditional probability that the target value is ***no,*** given the observed attribute values.

.0206 / (.0206 + .0053) = .795

Estimating Probabilities

- **P(Wind=strong | PlayTennis=no)** by the fraction **n_c/n** where ***n*** = 5 is the total number of training examples for which **PlayTennis=no,** and ***n_c*** = 3 is the number of these for which **Wind=strong.**
- When ***n_c*** is zero
  - **n_c/n** will be zero too
  - this probability term will dominate
- To avoid this difficulty we can adopt a Bayesian approach to estimating the probability, using the m-estimate defined as follows.

  m-estimate of probability: (n_c+ m*p) / (n + m)
- if an attribute has ***k*** possible values we set p = 1/k .
  - p=0.5 because Wind has two possible values.
- m is called the equivalent sample size
  - augmenting the ***n*** actual observations by an additional m virtual samples distributed according to p.

Bayesian Belief Networks

- The naive Bayes classifier makes significant use of the assumption that the values of the attributes a1 . . .a, are conditionally independent given the target value v.
    - This assumption dramatically reduces the complexity of learning the target function. When it is met, the naive Bayes classifier outputs the optimal Bayes classification.
- However, in many cases this conditional independence assumption is clearly overly restrictive.

- A Bayesian belief network describes the probability distribution governing a set of variables by specifying a set of conditional independence assumptions along with a set of conditional probabilities.
- In contrast to the naive Bayes classifier, which assumes that all the variables are conditionally independent given the value of the target variable, Bayesian belief networks allow stating conditional independence assumptions that apply to subsets of the variables.
- It provide an intermediate approach that is less constraining than the global assumption of conditional independence made by the naive Bayes classifier, but more tractable than avoiding conditional independence assumptions altogether

- In general, a Bayesian belief network describes the probability distribution over a set of variables.
- Consider an arbitrary set of random variables Yl . . . Y,, where each variable Yi can take on the set of possible values V(Yi).
- We define the joint space of the set of variables Y to be the cross product V(Yl) x V(Y2) x. . . V(Y,).
  - In other words, each item in the joint space corresponds to one of the possible assignments of values to the tuple of variables (Yl . . . Y,).
- The probability distribution over this joint' space is called the joint probability distribution.
- The joint probability distribution specifies the probability for each of the possible variable bindings for the tuple (Yl . . . Y,).

Conditional Independence

- Let us understand the idea of 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; that is, if

$$(\forall x_i, y_j, z_k)\ P(X = x_i | Y = y_j, Z = z_k) = P(X = x_i | Z = z_k)$$

- where $x_i \in V(X)$, $y_j \in V(Y)$, and $z_k \in V(Z)$.
- We commonly write the above expression in abbreviated form as P(XIY, Z) = P(X|Z).

- This definition of conditional independence can be extended to sets of variables as well.
- We say that the set of variables X1 . . . Xi is conditionally independent of the set of variables Yl . . . Ym given the set of variables Z1 . . . Zn, if
  - P(X1 ... Xi I Y1 ... Ym, Z1 ... Zn) = P(Xl ... X1|Z1 ... Zn)
- There is a relation between this definition and conditional independence in the definition of naïve bayes classifier.
- The naive Bayes classifier assumes that the instance attribute A1 is conditionally independent of instance attribute A2 given the target value V

- This allows the naive Bayes classifier to calculate P(Al, A21V) in equation

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

as follows:

$$\begin{aligned} P(A_1, A_2|V) &= P(A_1|A_2, V)P(A_2|V) \\ &= P(A_1|V)P(A_2|V) \end{aligned}$$

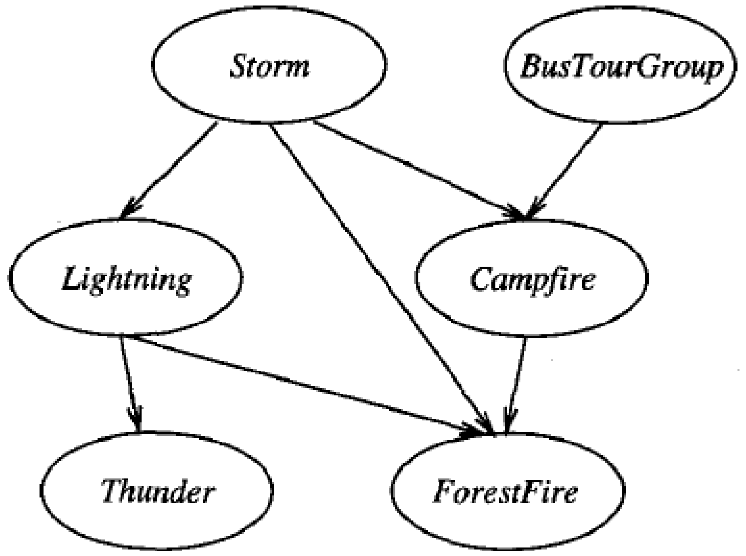
Representation

- In general, a Bayesian network represents the joint probability distribution by specifying a set of conditional independence assumptions represented by a directed acyclic graph, together with sets of local conditional probabilities.
- Each variable in the joint space is represented by a node in the Bayesian network.
- For each variable two types of information are specified.
  - First, the network arcs represent the assertion that the variable is conditionally independent of its nondescendants in the network given its immediate predecessors in the network
  - Second, a conditional probability table is given for each variable, describing the probability distribution for that variable given the values of its immediate predecessors

- The joint probability for any desired assignment of values (yl, . . . , yn) to the tuple of network variables (YI . . . Yn) can be computed by the formula:

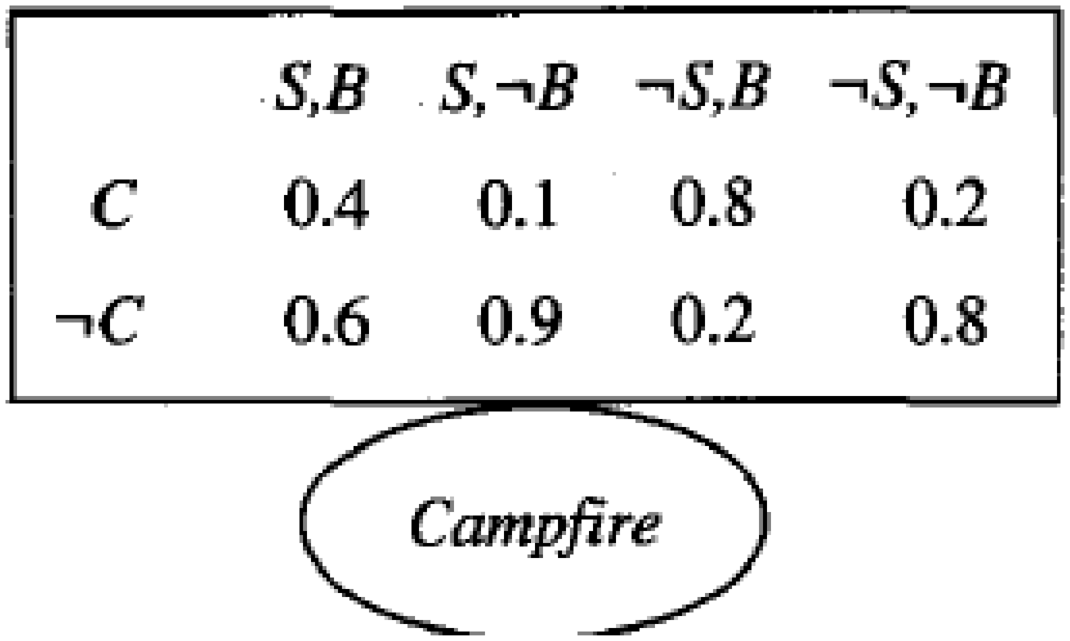
$$P(y_1, \ldots, y_n) = \prod_{i=1}^{n} P(y_i | Parents(Y_i))$$

- where Parents(Yi) denotes the set of immediate predecessors of Yi in the network.
- Note the values of P(yi|Parents(Yi)) are precisely the values stored in the conditional probability table associated with node Yi



- The figure represents the joint probability distribution over the boolean variables Storm, Lightning, Thunder, Forestfire, Campfire, andBusTourGroup.
- Consider the node Campfire. The network nodes and arcs represent the assertion that Campfire is conditionally independent of its nondescendants Lightning and Thunder, given its immediate parents Storm and BusTourGroup.
- This means that once we know the value of the variables Storm and BusTourGroup, the variables Lightning and Thunder provide no additional information about Campfire

- The table shows the conditional probability table associated with the variable Campfire.
- The top left entry in this table, for example, expresses the assertion that
  - P(Campfire = TruelStorm = True, BusTourGroup = True) = 0.4
- Note this table provides only the conditional probabilities of Campfire given its parent variables Storm and BusTourGroup.
- The set of local conditional probability tables for all the variables, together with the set of conditional independence assumptions described by the network, describe the full joint probability distribution for the network.
- Bayesian belief networks allows a convenient way to represent causal knowledge such as the fact that Lightning causes Thunder.
- In the terminology of conditional independence, we express this by stating that Thunder is conditionally independent of other variables in the network, given the value of Lightning.



	S,B	$S,\neg B$	$\neg S,B$	$\neg S,\neg B$
C	0.4	0.1	0.8	0.2
$\neg C$	0.6	0.9	0.2	0.8

Inference

- We might wish to use a Bayesian network to infer the value of some target variable (e.g., ForestFire) given the observed values of the other variables
- Given that we are dealing with random variables it will not generally be correct to assign the target variable a single determined value.
- What we really wish to infer is the probability distribution for the target variable, which specifies the probability that it will take on each of its possible values given the observed values of the other variables.
- This inference step can be straightforward if values for all of the other variables in the network are known exactly.
- In the more general case we may wish to infer the probability distribution for some variable (e.g., ForestFire) given observed values for only a subset of the other variables (e.g., Thunder and BusTourGroup may be the only observed values available).
- In general, 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.

Expectation –Maximization Algorithm

- A common modelling problem involves how to estimate a joint probability distribution for a dataset.
- Density estimation involves selecting a probability distribution function and the parameters of that distribution that best explain the joint probability distribution of the observed data.
- There are many techniques for solving this problem, although a common approach is called maximum likelihood estimation, or simply "maximum likelihood."
- Maximum Likelihood Estimation involves treating the problem as an optimization or search problem, where we seek a set of parameters that results in the best fit for the joint probability of the data sample.
- A limitation of maximum likelihood estimation is that it assumes that the dataset is complete, or fully observed. This does not mean that the model has access to all data; instead, it assumes that all variables that are relevant to the problem are present.
- This is not always the case. There may be datasets where only some of the relevant variables can be observed, and some cannot, and although they influence other random variables in the dataset, they remain hidden.
- More generally, these unobserved or hidden variables are referred to as latent variables.

- The EM algorithm is the technique that can be deployed in order to determine the local maximum likelihood estimates/ parameters (MLE) or maximum a posteriori (MAP) estimates/parameters for latent variables (unobservable variables that are inferred from observable variables) in statistical models.
- simply, the EM algorithm in machine learning uses observable instances of latent variables in order to predict values in instances, unobservable for learning, and continues till the convergence of the values takes place.

- Being an iterative approach, the EM algorithm revolves amid two modes, the first mode estimates the missing or latent variables, called E-step, and the second step optimizes the parameters of the model that explains data more clearly, called M-step. i.e.
  - E-step: Estimates the missing values in the dataset,
  - M-step: Maximize the model parameters while the data is present.
- The algorithm is used for predicting these values or in computing missing or incomplete data, given the generalized form of probability distribution that is connected with these latent variables.

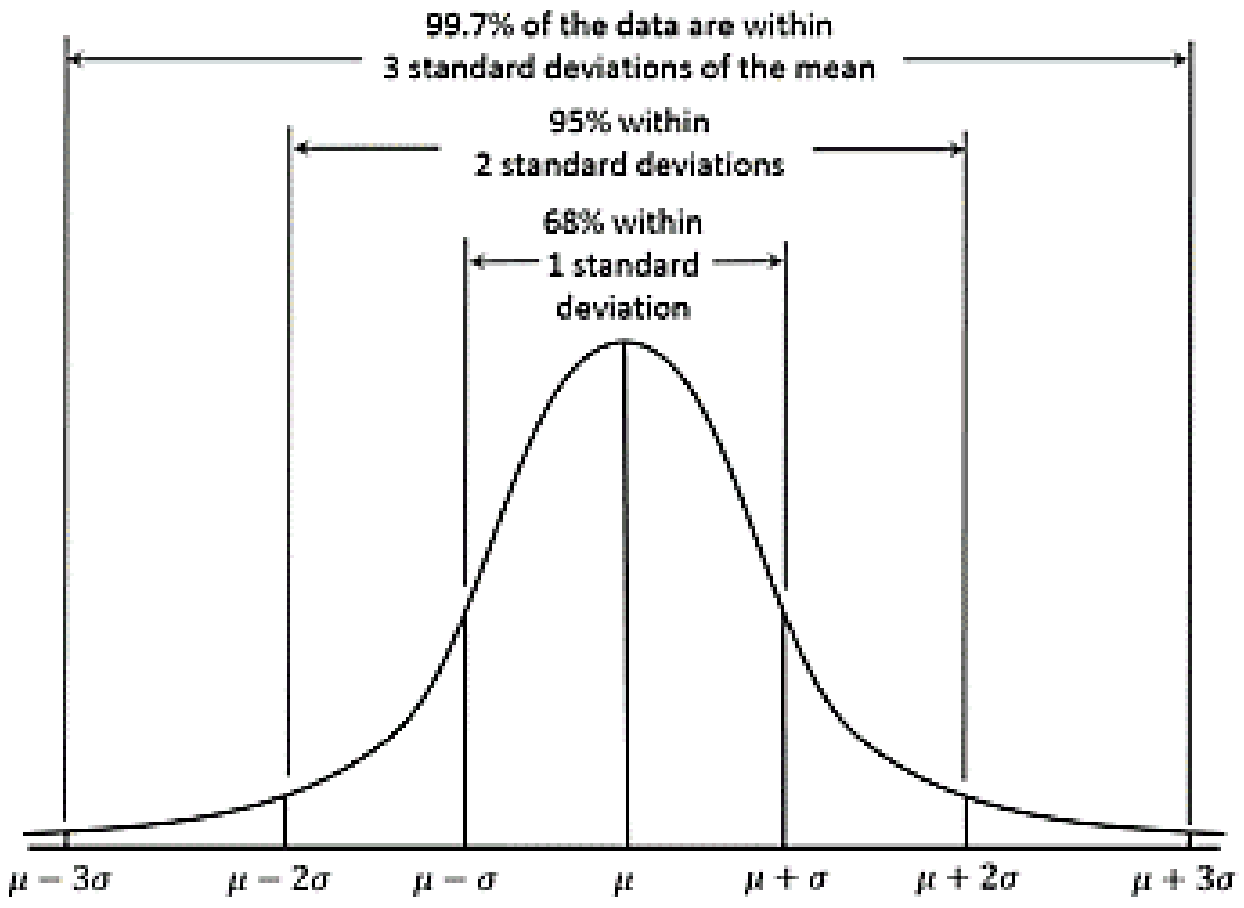
How Algorithm Works?

- **Step 1**: As having one set of missing or incomplete data and another set of starting parameters, we assume that observed data or initial values of the parameters are produced from the specific model.
  - Therefore, an entire set of incomplete observed data is provided to the system, assuming that an observed data comes from a specific model.
- **Step 2**: Depending on the observable value of the observable instances of the available data, next the values of unobservable instances, or missing data are predicted or estimated. This step is known as Expectation step, or E-step.
  - In this step, the observed data is used for estimating or guessing missing or incomplete data values that are used to update the variables.

- Step 3: By the produced data from E-step, next we update the parameters and complete the data set. This step is known as Maximization step, or M-step. And we update the hypothesis.
- As the last step, we check whether the values are converging or not, if yes, stop the process.
- If not, then step 2 and step 3 will be imitated until the state of convergence is achieved, or simply, we will repeat E-step and M-step if the values are not converging.

- A Gaussian is a type of distribution, and it is a popular and mathematically convenient type of distribution. A distribution is a listing of outcomes of an experiment and the probability associated with each outcome.
- It forms a bell curve the frequencies go up as the speed goes up and then it has a peak value and then it goes down again, and we can represent this using a bell curve otherwise known as a Gaussian distribution.

- A Gaussian distribution is a type of distribution where half of the data falls on the left of it, and the other half of the data falls on the right of it. It’s an even distribution, and one can notice just by the thought of it intuitively that it is very mathematically convenient.
- So, what do we need to define a Gaussian or Normal Distribution? We need a mean which is the average of all the data points. That is going to define the centre of the curve, and the standard deviation which describes how to spread out the data is. Gaussian distribution would be a great distribution to model the data in those cases where the data reaches a peak and then decreases.



- Normal distributions has the same variance a2, and where a2 is known. The learning task is to output a hypothesis h = (µ1 …µk) that describes the means of each of the k distributions.
- Next we would like to find a maximum likelihood hypothesis for these means; that is, a hypothesis h that maximizes p(D lh).
- Note it is easy to calculate the maximum likelihood hypothesis for the mean of a single Normal distribution given the observed data instances XI, x2, . . . , xm drawn from this single distribution

- Similarly, in Multi Gaussian Distribution, we will have multiple peaks with multiple means and multiple standard deviations.
- Sometimes our data has multiple distributions or it has multiple peaks. It does not always have one peak, and one can notice that by looking at the data set. It will look like there are multiple peaks happening here and there.

Speed (Km/h)	**Frequency**
1	4
2	9
3	6
4	7
5	3
6	2

- There are two peak points and the data seems to be going up and down twice or maybe three times or four times. But if there are Multiple Gaussian distributions that can represent this data, then we can build what we called a Gaussian Mixture Model.
- In other words we can say that, if we have three Gaussian Distribution as GD1, GD2, GD3 having mean as µ1, µ2,µ3 and variance 1,2,3 than for a given set of data points GMM will identify the probability of each data point belonging to each of these distributions.

- If we were to plot multiple Gaussian distributions, it would be multiple bell curves. What we really want is a single continuous curve that consists of multiple bell curves.
- Once we have that huge continuous curve then for the given data points, it can tell us the probability that it is going to belong to a specific class.
- Now, we would like to find the maximum likelihood estimate of X (the data point we want to predict the probability) i.e. we want to maximize the likelihood that X belongs to a particular class or we want to find a class that this data point X is most likely to be part of.