Unit-3

Bayesian learning –

Bayesian learning and the frequents method can also be considered as two ways of looking at the tasks of estimating values of unknown parameters given some observations caused by those parameters. For certain tasks, either the concept of uncertainty is meaningless or interpreting prior beliefs is too complex

Bayes theorem:

Bayes Theorem is a method to determine conditional probabilities – that is, the **probability** of one event occurring given that another event has already occurred. ... Thus, conditional probabilities are a must in determining accurate predictions and probabilities in **Machine Learning**

Bayes's formula

Below is Bayes's formula.

$P(A|B) = P(B|A)P(A)/P(B)\}$

The formula provides the relationship between P(A|B) and P(B|A). It is mainly derived from conditional probability formula discussed in the previous post.

Consider the below formulas for conditional probabilities P(A|B) and P(B|A)

$P(A|B) = P(A \cap B)/P(B)\}$ —-(1)

$P(B|A) = P(B \cap A)/P(A)\}$ —-(2)

Since $P(B \cap A) = P(A \cap B)$, we can replace $P(A \cap B)$ in the first formula with $P(B|A)P(A)$

After replacing, we get the given formula.

Product Rule

Product rule states that

$P(x \cap y)=p(x|y)*p(y)$(1)

So the joint probability that both X and Y will occur is equal to the product of two terms:

From the product rule :

implies **P(X|Y) = P(X)/P(Y)**

implies **P(X|Y) = 1**

Chain rule

When the above product rule is generalized we lead to chain rule. Let there are E_1, E_2,....E_n. n events . Then, the joint probability is given by

$$P(\bigcap_{i=1,..,n} E_i) = P(E_n|\bigcap_{i=1,..,n-1} E_i) * P(\bigcap_{i=1,..,n-1} E_i) \quad (2)$$

Bayes' Theorem

From the product rule, $P(X \cap Y) = P(X|Y)P(Y)$ and $P(Y \cap X) = P(Y|X)P(X)$. As $P(X \cap Y)$ and $P(Y \cap X)$ are same .

$$P(Y|X) = \frac{P(X|Y) * P(Y)}{P(X)} \quad (3)$$

where $P(X) = P(X \cap Y) + P(X \cap Y^c)$.

Example : Box P has 2 red balls and 3 blue balls and box Q has 3 red balls and 1 blue ball. A ball is selected as follows:

(i) Select a box

(ii) Choose a ball from the selected box such that each ball in

the box is equally likely to be chosen. The probabilities of

selecting boxes P and Q are (1/3) and (2/3), respectively.

Given that a ball selected in the above process is a red ball, the probability that it came from the box P is (GATE CS 2005)

(A) 4/19

(B) 5/19

(C) 2/9

(D) 19/30

Solution:

```
R --> Event that red ball is selected
B --> Event that blue ball is selected
P --> Event that box P is selected
Q --> Event that box Q is selected

We need to calculate P(P|R)?
```

$$P(P|R) = \frac{P(R|P)P(P)}{P(R)}$$

```
P(R|P) = A red ball selected from box P
       = 2/5
P(P) = 1/3
P(R) = P(P)*P(R|P) + P(Q)*P(R|Q)
     = (1/3)*(2/5) + (2/3)*(3/4)
     = 2/15 + 1/2
     = 19/30
```

Putting above values in the Bayes's Formula

P(P|R) = (2/5)*(1/3) / (19/30)

= 4/19

Practice question

Exercise A company buys 70% of its computers from company X and 30% from company Y. Company X produces 1 faulty computer per 5 computers and company Y produces 1 faulty computer per 20 computers. A computer is found faulty what is the probability that it was bought from company X?

Bayes' Theorem

Bayes' Theorem or Bayes' Rule is named after Reverend Thomas Bayes. It describes the probability of an event, based on prior knowledge of conditions that might be related to that event. It can also be considered for conditional probability examples.

For example: There are 3 bags, each containing some white marbles and some black marbles in each bag. If a white marble is drawn at random. With probability to find that this white marble is from the first bag. In cases like such, we use the Bayes' Theorem. It is used where the probability of occurrence of a particular event is calculated based on other conditions which are also called conditional probability. So before jumping into detail let's have a brief discussion on **The theorem of Total Probability**.

Theorem of Total Probability

Let $E_1, E_2, \ldots\ldots\ldots\ldots E_n$ is mutually exclusive and exhaustive events associated with a random experiment and lets E be an event that occurs with some E_i. Then, prove that

$$P(E) = {}^{n}\sum_{i=1} P(E/E_i) \,.\, P(E_j)$$

Proof:

$S = E_1 \cup E_2 \cup E_3 \cup \ldots\ldots\ldots\ldots\ldots\ldots \cup En$ and $E_i \cap E_j = \emptyset$ for $i \neq j$.

Therefore, $E = E \cap S = E \cap (E_1 \cup E_2 \cup E_3 \cup \ldots\ldots\ldots\ldots\ldots\ldots \cup E_n)$

$= (E \cap E_1) \cup (E \cap E_2) \cup \ldots\ldots \cup (E \cap E_n)$

$\Rightarrow P(E) = P\{(E \cap E_1) \cup (E \cap E_2) \cup \ldots\ldots \cup (E \cap E_n)\}$

$= P(E \cap E_1) + P(E \cap E_2) + \ldots\ldots + P(E \cap E_n)$

$= \{$Therefore, $(E \cap E_1), (E \cap E_2), \ldots\ldots\ldots\ldots, (E \cap E_n)\}$ are pairwise disjoint$\}$

$= P(E/E_1) \,.\, P(E_1) + P(E/E_2) \,.\, P(E_2) + \ldots\ldots\ldots\ldots\ldots\ldots + P(E/E_n) \,.\, P(E_n)$ [by multiplication theorem]

$= {}^{n}\sum_{i=1} P(E/E_i) \,.\, P(E_i)$

Examples

Example 1: A person has undertaken a job. The probabilities of completion of the job on time with and without rain are 0.44 and 0.95 respectively. If the probability that it will rain is 0.45, then determine the probability that the job will be completed on time?

Solution:

Let E1 be the event that the mining job will be completed on time and E2 be the event t hat it rains. We have,

P(A) = 0.45,

P(no rain) = P(B) = 1 − P(A) = 1 − 0.45 = 0.55

By multiplication law of probability,

P(E1) = 0.44

P(E2) = 0.95

Since, events A and B form partitions of the sample space S, by total probability theore m, we have

$$P(E) = P(A)\,P(E1) + P(B)\,P(E2)$$

= 0.45 × 0.44 + 0.55 × 0.95

= 0.198 + 0.5225 = 0.7205

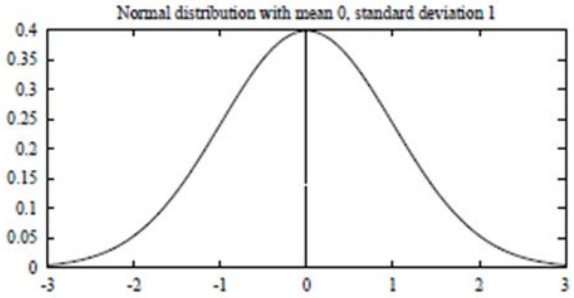
So, the probability that the job will be completed on time is 0.684.

Bayes Theorem & Concept Learning

- Bayes theorem is a principled way to calculate posterior probability of each hypothesis
- Assumptions: The training data is noise free $(\text{i.e., } d_i = c(x_i))$.The target concept is contained in . We have no a priori reason to believe that any hypothesis is more probable than any ot
- $P(h) = \frac{1}{|H|}$ for all h in H
- $$P(D|h) = \begin{cases} 1 & \text{if } d_i = h(x_i) \text{ for all } d_i \text{ in } D \\ 0 & \text{otherwise} \end{cases}$$

- $P(D) = \sum_{h_i \in H} P(D|h_i)P(h_i) = \sum_{h_i \in VS_{H,D}} 1 \cdot \frac{1}{|H|} + \sum_{h_i \notin VS_{H,D}} 0 \cdot \frac{1}{|H|} = \frac{|VS_{H,D}|}{|H|}$
- So if h is inconsistent with D, $P(h|D) = \frac{P(D|h)P(h)}{P(D)} = \frac{0 \cdot P(h)}{P(D)} = 0$
- So if h is consistent with D, $P(h|D) = \frac{1 \cdot \frac{1}{|H|}}{\frac{|VS_{H,D}|}{|H|}} = \frac{1}{|VS_{H,D}|}$
- every consistent hypothesis is a MAP hypothesis

Basic Concepts from Probability Theory



A **Normal Distribution (Gaussian Distribution)** is a bell-shaped distribution defined by the ***probability density function***

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

- A Normal distribution is fully determined by two parameters in the formula: μ and σ .
- If the random variable X follows a normal distribution:
  - The probability that X will fall into the interval (a, b) is $\int_a^b p(x)dx$
  - The expected, or ***mean value of X***, E[X] = μ
  - The ***variance of X***, Var(X) = σ^2
  - The ***standard deviation of X***, $\sigma_x = \sigma$
- The ***Central Limit Theorem*** states that the sum of a large number of independent, identically distributed random variables follows a distribution that is approximately ***Normal***.

Maximum likelihood and least squared error hypotheses

Two commonly used approaches to estimate population parameters from a random sa mple are the **maximum likelihood** estimation method (default) and the **least squares** es timation method. The **likelihood** function indicates how likely the observed sample is as a function of possible parameter values.

Maximum Likelihood

The equation above looks simple but it is notoriously tricky to compute in practice — because of extremely large hypothesis space and complexity in evaluating integrals over complicated probability distribution functions.

However, in the quest of our search for the '*most probable hypothesis given data,*' we can simplify it further.

- We can drop the term in the denominator it does not have any term containing ***h*** i.e. hypothesis. We can imagine it as a normalizer to make total probability sum up to 1.
- Uniform prior assumption — this essentially relaxes any assumption on the nature of $P(h)$ by making it uniform i.e. all hypotheses are probable. Then it is a constant number 1/|Vsd| where |Vsd| is the size of the version space i.e. a set of all hypothesis consistent with the training data. Then it does not actually figure in the determination of the maximally probable hypothesis.

After these two simplifying assumptions, the **maximum likelihood (ML)** hypothesis can be given by,

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

This simply means the most likely hypothesis is the one for which the conditional prob ability of the observed data (given the hypothesis) reaches maximum.

Noise in the data

We generally start using least-square error while learning about simple linear regressio n back in Stats 101 but this simple-looking loss function resides firmly inside pretty m uch every supervised machine learning algorithm viz. linear models, splines, decision t rees, or deep learning networks.

So, what's special about it? Is it related to the Bayesian inference in any way?

It turns out that, the key connection between the least-square error and Bayesian infer ence is through the assumed nature of the error or residuals.

Measured/observed data is never error-free and there is always random noise associa

ted with data, which can be thought of the signal of interest. Task of a machine learnin g algorithm is to estimate/approximate the function which could have generated the d ata by separating the signal from the noise.

But what can we say about the nature of this noise? It turns out that noise can be mod eled as a random variable. Therefore, we can associate a probability distribution of our choice to this random variable.

One of the key assumptions of least-square optimization is that probability distributio n over residuals is our trusted old friend — Gaussian Normal.

This means that every data point (d) in a supervised learning training data set can be written as the sum of the unknown function f(x) (which the learning algorithm is trying to approximate) and an error term which is drawn from a Normal distribution of zero mean (μ) and unknown variance σ². This is what I mean,

$$d_i = f(x_i) + e(i) = f(x_i) + \boldsymbol{N}(\mu = 0, \sigma^2)$$

And from this assertion, we can easily derive that the maximum likely hypothesis is the one which minimizes the least-square error.

Derivation of least-square from Maximum Likelihood hypothesis

We start with the **maximum likelihood hypothesis**:

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

We assume a fixed set of training instances $\langle x_1, x_2, \ldots, x_n \rangle$.
Therefore, we consider the data D to be the corresponding sequence of target values $D = \langle d_1, d_2, \ldots, d_n \rangle$
Here, $d_i = f(x_i) + e_i$ where e_i is the ***Normally distributed*** error.
Assuming the training examples are mutually independent given h, we can write $P(D|h)$ as the product of the various $p(d_i|h)$

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} p(d_i|h)$$

Given that the noise e_i obeys a Normal distribution with zero mean(μ) and unknown variance σ^2, each d_i must also obey a Normal distribution with variance σ^2 centered around the true target value $f(x_i)$ rather than zero. Therefore $p(d_i|h)$ can be written as a Normal distribution with variance σ^2 and mean $\mu = f(x_i)$. Because we are writing the expression for the probability of d_i given that h is the correct description of the target function f, we will also substitute $\mu = f(x_i) = h(x_i)$, yielding,

$$\begin{aligned} h_{ML} &= \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i - \mu)^2} \\ &= \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i - h(x_i))^2} \end{aligned}$$

We now apply **log-likelihood** transformation. This is justified because $ln(p)$ is a monotonic function of p. Therefore maximizing $ln(p)$ also maximizes p. So, the product becomes summation.

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} \left[\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}(d_i - h(x_i))^2 \right]$$

The first term in this expression is a constant independent of h, and can therefore be discarded, yielding

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} -\frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

Maximizing this negative quantity is equivalent to minimizing the corresponding positive quantity.

$$h_{ML} = \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{m} \frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

Finally, we can again discard constants that are independent of h.

$$h_{ML} = \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{m} (d_i - h(x_i))^2$$

MAXIMUM LIKELIHOOD HYPOTHESES FOR PREDICTING PROBABILITIES

In the problem setting of the previous section we determined that the maximum likelihood hypothesis is the one that minimizes the sum of squared errors over the training examples.

• In this section we derive an analogous criterion for a second setting that is common in neural network learning: learning to predict probabilities.

• Consider the setting in which we wish to learn a nondeterministic (probabilistic) function f : X {0, 1}, which has two discrete output values.

• For example, the instance space X might represent medical patients in terms of their symptoms, and the target function f (x) might be 1 if the patient survives the disease and 0 if not.

• Alternatively, X might represent loan applicants in terms of their past credit history, and f (x) might be 1 if the applicant successfully repays their next loan and 0 if not. In both of these cases we might well expect f to be probabilistic.

• For example, among a collection of patients exhibiting the same set of observable symptoms, we might find that 92% survive, and 8% do not. This unpredictability could arise from our inability to observe all the important distinguishing features of the patients, or from some

genuinely probabilistic mechanism in the evolution of the disease.

• Whatever the source of the problem, the effect is that we have a target function f (x) whose output is a probabilistic function of the input.

Example•

Given this problem setting, we might wish to learn a neural network (or other real-valued function approximator) whose output is the probability that f (x) = 1.

• In other words, we seek to learn the target function, f' : X [O, 1], such that f '(x) = P(f (x) = 1).

• In the above medical patient example, if x is one of those indistinguishable patients of which 92% survive, then f'(x) = 0. 92 whereas the probabilistic function f (x) will be equal to 1 in 92% of cases and equal to 0 in the remaining 8%.

• How can we learn f' using, say, a neural network? One obvious, bruteforce way would be to first collect the observed frequencies of 1's and 0's for each possible value of x and to then train the neural network to output the target frequency for each x.

• As we shall see below, we can instead train a neural network directly from the observed training examples of f, yet still derive a maximum likelihood hypothesis for f '.

• What criterion should we optimize in order to find a maximum likelihood hypothesis for f' in this setting? To answer this question we must first obtain an expression for P(D|h).

• Let us assume the training data D is of the form D = {(x 1, d 1). . . (xm, dm)}, where di is the observed 0 or 1 value for f (xi).

• Recall that in the maximum likelihood, least-squared error analysis of the previous section, we made the simplifying assumption that the instances (x 1. . . xm) were fixed.

• This enabled us to characterize the data by considering only the target values di.

- Consider the setting in which we wish to learn a nondeterministic function f: $X \rightarrow \{0,1\}$
- We might expect f to be probabilistics
- For example, for neural network learning, we might output the f(x)=1 with probability 92%
  - $f': X \rightarrow [0,1]$, $f'=P(f(x)=1)$

v

- Brute-Force
  - Collecting d_i is the observed 0 or 1 for $f(x_i)$
- Assume training data D is of the form $D=\{<x_1,d_1>...<x_m,d_m>\}$

$$P(D \mid h) = \prod_{i=1}^{m} P(x_i, d_i \mid h) = \prod_{i=1}^{m} P(d_i \mid h, x_i) P(x_i)$$

$$P(d_i \mid h, x_i) = \begin{cases} h(x_i) & d_i = 1 \\ 1 - h(x_i) & d_i = 0 \end{cases} = h(x_i)^{d_i} (1 - h(x_i))^{1-d_i}$$

$$P(D \mid h) = \prod_{i=1}^{m} h(x_i)^{d_i} (1 - h(x_i))^{1-d_i} P(x_i)$$

$$h_{ML} = \arg\max_{h \in H} \prod_{i=1}^{m} h(x_i)^{d_i} (1 - h(x_i))^{1-d_i} p(x_i)$$

$$= \arg\max_{h \in H} \prod_{i=1}^{m} h(x_i)^{d_i} (1 - h(x_i))^{1-d_i}$$

$$= \arg\max_{h \in H} \sum_{i=1}^{m} d_i \ln h(x_i) + (1 - d_i) \ln(1 - h(x_i))$$

Minimum description length principle

Occam's razor: prefer the shortest hypothesis

MDL: prefer the hypothesis h that minimizes

$$h_{MDL} = \arg\min_{h \in H} L_{C1}(h) + L_{C2}(D \mid h)$$

where $L_C(x)$ is the description length of x under encoding C

Example:

- H = decision trees, D = training data labels
- $L_{C1}(h)$ is # bits to describe tree h
- $L_{C2}(D|h)$ is #bits to describe D given h
  - Note $L_{C2}(D|h) = 0$ if examples classified perfectly by h. Need only describe exceptions
- Hence h_{MDL} trades off tree size for training errors

$$h_{MAP} = \arg\max_{h \in H} P(D \mid h) P(h)$$
$$= \arg\max_{h \in H} \log_2 P(D \mid h) + \log_2 P(h)$$
$$= \arg\min_{h \in H} -\log_2 P(D \mid h) - \log_2 P(h) \quad (1)$$

Interesting fact from information theory:

The optimal (shortest expected length) code for an event with probability p is $\log_2 p$ bits.

So interpret (1):

$-\log_2 P(h)$ is the length of h under optimal code

$-\log_2 P(D|h)$ is length of D given h in optimal code

→ prefer the hypothesis that minimizes

length(h)+length(misclassifications)

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.

- 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 v_j from some set V, then the probability $P(v_j \mid D)$ that the correct classification for the new instance is v_j :

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

- **Bayes optimal classification:**

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

Example:

$$P(h_1|D) = .4, \quad P(\ominus|h_1) = 0, \quad P(\oplus|h_1) = 1$$
$$P(h_2|D) = .3, \quad P(\ominus|h_2) = 1, \quad P(\oplus|h_2) = 0$$
$$P(h_3|D) = .3, \quad P(\ominus|h_3) = 1, \quad P(\oplus|h_3) = 0$$

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:

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

- 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.***

Gibbs Algorithm

·· Bayes Optimal is quite costly to apply. It computes the posterior probabilities for every hypothesis in H and combines the predictions of each hypothesis to classify each new instance

· An alternative (less optimal) method:

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 .

· Under certain conditions the expected misclassification error for Gibbs algorithm is at most twice the expected error of the Bayes optimal classifier.

· Bayes Optimal is quite costly to apply. It computes the posterior probabilities for every hypothesis in H and combines the predictions of each hypothesis to classify each new instance

· An alternative (less optimal) method:

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 .

· Under certain conditions the expected misclassification error for Gibbs algorithm is at most twice the expected error of the Bayes optimal classifier.

Naive Bayes Classifiers

A classifier is a machine learning model that is used to discriminate different objects based on certain features.

A Naive Bayes classifier is a probabilistic machine learning model that's used for classification task. The crux of the classifier is based on the Bayes theorem.

Naive Bayes classifiers are a collection of classification algorithms based on **Bayes' Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is

independent of each other.

Bayes Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Using Bayes theorem, we can find the probability of **A** happening, given that **B** has occurred. Here, **B** is the evidence and **A** is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naive.

Example:

Let us take an example to get some better intuition. Consider the problem of playing golf. The dataset is represented as below.

Consider a fictional dataset that describes the weather conditions for playing a game of golf. Given the weather conditions, each tuple classifies the conditions as fit("Yes") or unfit("No") for plaing golf.

Here is a tabular representation of our dataset.

	OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY GOLF
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

We classify whether the day is suitable for playing golf, given the features of the day. The columns represent these features and the rows represent individual entries. If we take the first row of the dataset, we can observe that is not suitable for playing golf if the outlook is rainy, temperature is hot, humidity is high and it is not windy. We make two assumptions here, one as stated above we consider that these predictors are independent. That is, if the temperature is hot, it does not necessarily mean that the humidity is high. Another assumption made here is that all the predictors have an equal effect on the outcome. That is, the day being windy does not have more importance in deciding to play golf or not.

According to this example, Bayes theorem can be rewritten as:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

The variable **y** is the class variable(play golf), which represents if it is suitable to play golf or not given the conditions. Variable **X** represent the parameters/features.

X is given as,

$$X = (x_1, x_2, x_3, ..., x_n)$$

Here x_1,x_2....x_n represent the features, i.e they can be mapped to outlook, temperature, humidity and windy. By substituting for **X** and expanding using the chain rule we get

$$P(y|x_1, ..., x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

Now, you can obtain the values for each by looking at the dataset and substitute them into the equation. For all entries in the dataset, the denominator does not change, it remain static. Therefore, the denominator can be removed and a proportionality can be introduced.

$$P(y|x_1, ..., x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y)$$

In our case, the class variable(**y**) has only two outcomes, yes or no. There could be cases where the classification could be multivariate. Therefore, we need to find the class **y** with maximum probability.

$$y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y)$$

Types of Naive Bayes Classifier:

Multinomial Naive Bayes:

This is mostly used for document classification problem, i.e whether a document belongs to the category of sports, politics, technology

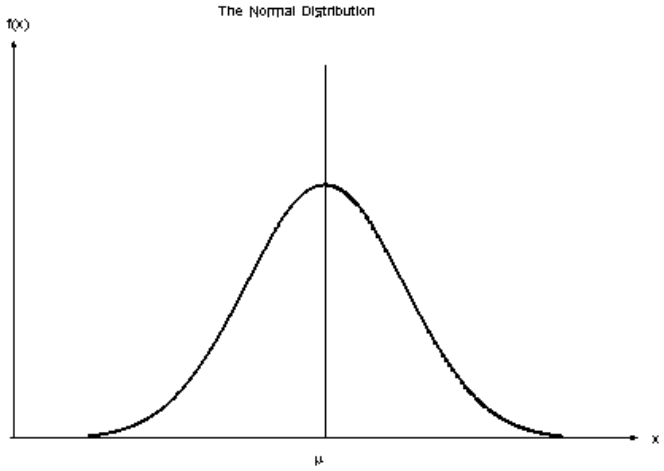
etc. The features/predictors used by the classifier are the frequency of the words present in the document.

Bernoulli Naive Bayes:

This is similar to the multinomial naive bayes but the predictors are boolean variables. The parameters that we use to predict the class variable take up only values yes or no, for example if a word occurs in the text or not.

Gaussian Naive Bayes:

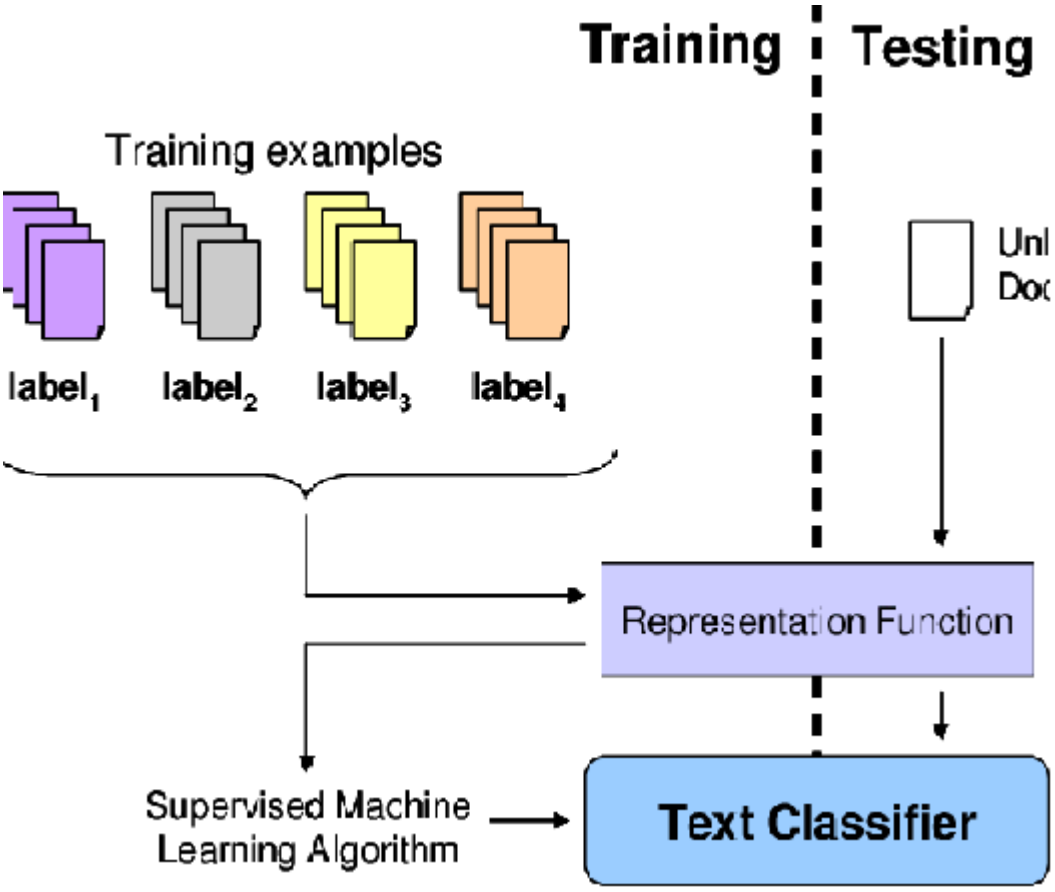
When the predictors take up a continuous value and are not discrete, we assume that these values are sampled from a gaussian distribution.



Since the way the values are present in the dataset changes, the formula for conditional probability changes to,

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

An example learning to classify text



Bayesian belief networks

Bayesian belief network is key computer technology for dealing with probabilistic events and to solve a problem which has uncertainty. We can define a Bayesian network as:

"A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph."

It is also called a **Bayes network, belief network, decision network**, or **Bayesian model**.

Bayesian networks are probabilistic, because these networks are built from a **probability distribution**, and also use probability theory for prediction and anomaly detection.

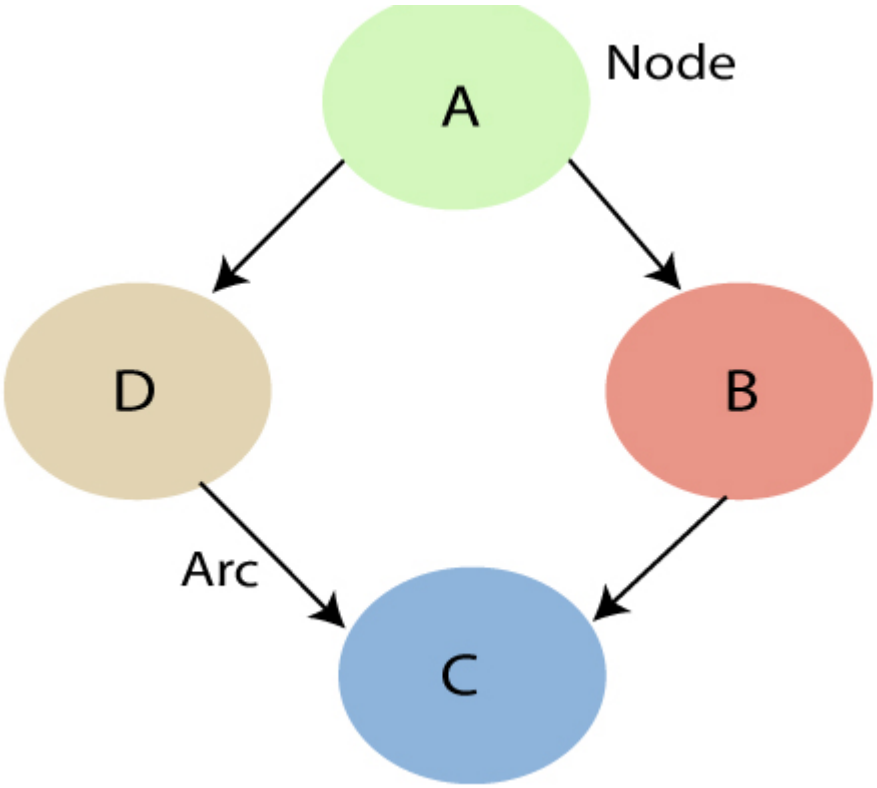
Real world applications are probabilistic in nature, and to represent the relationship between multiple events, we need a Bayesian network. It can also be used in various tasks including **prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction**, and **decision making under uncertainty**.

Bayesian Network can be used for building models from data and experts opinions, and it consists of two parts:

- **Directed Acyclic Graph**
- **Table of conditional probabilities.**

The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as an **Influence diagram**.

A Bayesian network graph is made up of nodes and Arcs (directed links), where:



- Each **node** corresponds to the random variables, and a variable can be **continuous** or **discrete**.
- **Arc or directed arrows** represent the causal relationship or conditional probabilities between random variables. These directed links or arrows connect the pair of nodes in the graph.
  These links represent that one node directly influence the other node, and if there is no directed link that means that nodes are independent with each other
  - **In the above diagram, A, B, C, and D are random variables represented by the nodes of the network graph.**
  - **If we are considering node B, which is connected with node A by a directed arrow, then node A is called the parent of Node B.**
  - **Node C is independent of node A.**

The Bayesian network has mainly two components:

- **Causal Component**
- **Actual numbers**

Each node in the Bayesian network has condition probability distribution $P(X_i \mid Parent(X_i))$, which determines the effect of the parent on that node.

Bayesian network is based on Joint probability distribution and conditional probability. So let's first understand the joint probability distribution:

Joint probability distribution:

If we have variables x1, x2, x3,....., xn, then the probabilities of a different combination of x1, x2, x3.. xn, are known as Joint probability distribution.

$P[x_1, x_2, x_3, ..., x_n]$, it can be written as the following way in terms of the joint probability distribution.

$= P[x_1 \mid x_2, x_3, ..., x_n]P[x_2, x_3, ..., x_n]$

$= P[x_1 \mid x_2, x_3, ..., x_n]P[x_2 \mid x_3, ..., x_n]....P[x_{n-1} \mid x_n]P[x_n].$

In general for each variable Xi, we can write the equation as:

$P(X_i|X_{i-1},\ldots\ldots, X_1) = P(X_i|\text{Parents}(X_i))$

Explanation of Bayesian network:

Let's understand the Bayesian network through an example by creating a directed acyclic graph:

Problem:

Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occurred, and David and Sophia both called the Harry.

Solution:

- The Bayesian network for the above problem is given below. The network structure is showing that burglary and earthquake is the parent node of the alarm and directly affecting the probability of alarm's going off, but David and Sophia's calls depend on alarm probability.
- The network is representing that our assumptions do not directly perceive the burglary and also do not notice the minor earthquake, and they also not confer before calling.
- The conditional distributions for each node are given as conditional probabilities table or CPT.
- Each row in the CPT must be sum to 1 because all the entries in the table represent an exhaustive set of cases for the variable.
- In CPT, a boolean variable with k boolean parents contains 2^K probabilities. Hence, if there are two parents, then CPT will contain 4 probability values

List of all events occurring in this network:

- **Burglary (B)**
- **Earthquake(E)**
- **Alarm(A)**
- **David Calls(D)**
- **Sophia calls(S)**

We can write the events of problem statement in the form of probability: P[D, S, A, B, E], can rewrite the above probability statement using joint probability distribution:

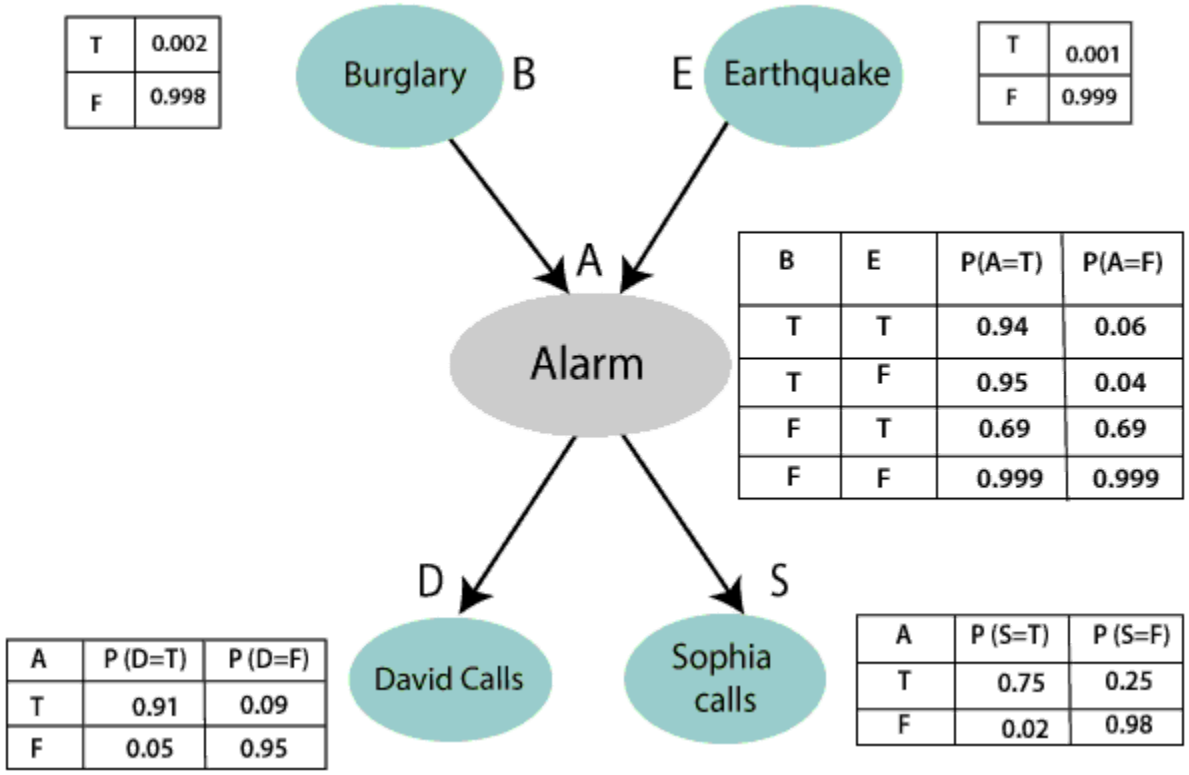
P[D, S, A, B, E]= P[D | S, A, B, E]. P[S, A, B, E]

=P[D | S, A, B, E]. P[S | A, B, E]. P[A, B, E]

= P [D| A]. P [S| A, B, E]. P[A, B, E]

= P[D | A]. P[S | A]. P[A| B, E]. P[B, E]

= P[D | A]. P[S | A]. P[A| B, E]. P[B |E]. P[E]



T	0.002
F	0.998

T	0.001
F	0.999

B	E	P(A=T)	P(A=F)
T	T	0.94	0.06
T	F	0.95	0.04
F	T	0.69	0.69
F	F	0.999	0.999

A	P (D=T)	P (D=F)
T	0.91	0.09
F	0.05	0.95

A	P (S=T)	P (S=F)
T	0.75	0.25
F	0.02	0.98