

ECE5984 – Applications of Machine Learning

Lecture 14 – Bayesian Prediction

Creed Jones, PhD

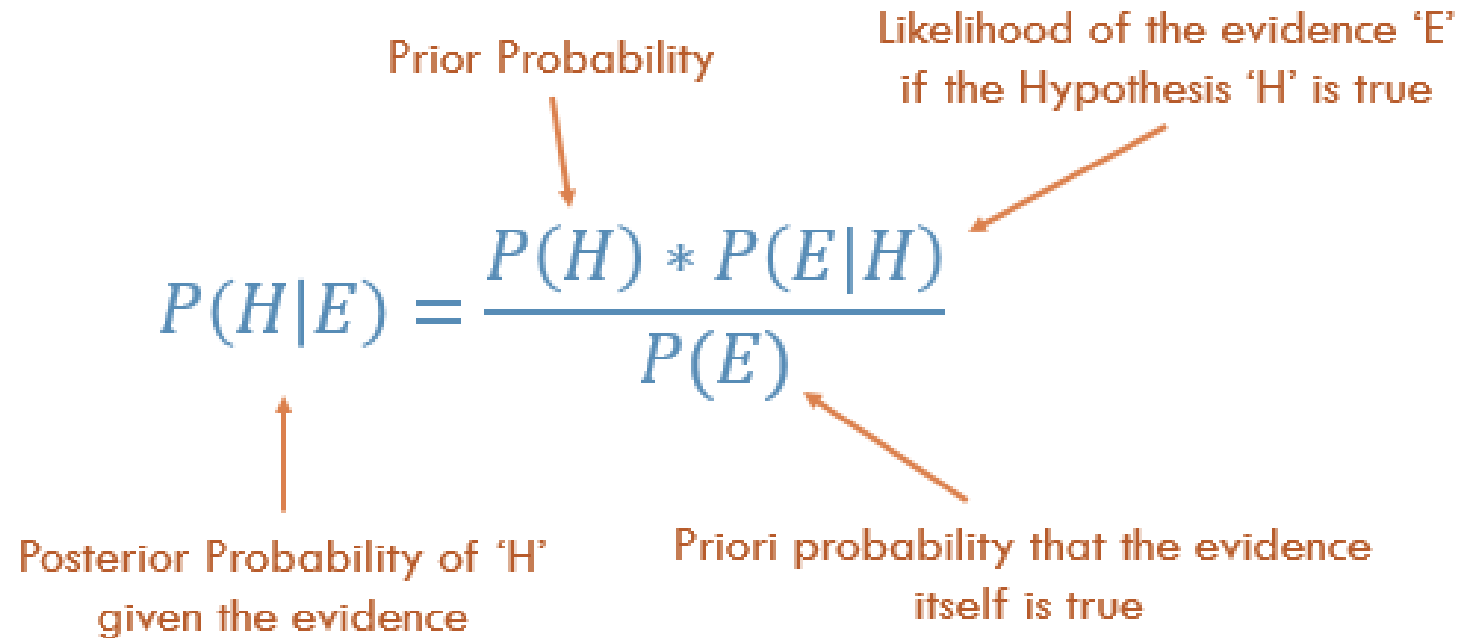
Course update

- Quiz 4 TODAY
- Project I due Tuesday, March 22
 - Hope you're working on it!
- Spring break next week

Today's Objectives

- Conditional Independence
- Bayesian Prediction
- Smoothing
- Continuous Features: Use of Probability Density Functions
- Bayesian Networks

Bayes' theorem relates *prior* (before we have additional knowledge) and *posterior* (after we know more) probabilities



The diagram shows the Bayes' theorem formula $P(H|E) = \frac{P(H) * P(E|H)}{P(E)}$ with four orange arrows pointing to its components:

- An arrow from "Prior Probability" points to $P(H)$.
- An arrow from "Likelihood of the evidence 'E' if the Hypothesis 'H' is true" points to $P(E|H)$.
- An arrow from "Posterior Probability of 'H' given the evidence" points to $P(H|E)$.
- An arrow from "Prior probability that the evidence itself is true" points to $P(E)$.

$$P(H|E) = \frac{P(H) * P(E|H)}{P(E)}$$

The divisor is the prior probability of the evidence
This division functions as a normalization constant

The generalized form of Bayes' theorem relates to cases with several prior events

- If many events form the prior, then we use the joint distribution to calculate the probability of all of them occurring
- The form below relates the likelihood of the target variable t taking on the value l , given that events $q[1], q[2] \cdots q[m]$ have all occurred

Generalized Bayes' Theorem

$$P(t = l | \mathbf{q}[1], \dots, \mathbf{q}[m]) = \frac{P(\mathbf{q}[1], \dots, \mathbf{q}[m] | t = l) P(t = l)}{P(\mathbf{q}[1], \dots, \mathbf{q}[m])}$$

Chain Rule

$$P(\mathbf{q}[1], \dots, \mathbf{q}[m]) = \\ P(\mathbf{q}[1]) \times P(\mathbf{q}[2]|\mathbf{q}[1]) \times \\ \dots \times P(\mathbf{q}[m]|\mathbf{q}[m-1], \dots, \mathbf{q}[2], \mathbf{q}[1])$$

To apply the chain rule to a conditional probability we just add the conditioning term to each term in the expression:

$$P(q[1], \dots, q[m]|t = l) = \\ P(q[1]|t = l) \times P(q[2]|q[1], t = l) \times \dots \\ \dots \times P(q[m]|q[m-1], \dots, q[3], q[2], q[1], t = l)$$

$$P(A, B, C|t = F) = P(A|t = F) \times P(B|A, t = F) \times P(C|A, B, t = F)$$

Higher dimensions means fewer instances of each combination of feature values

Curse of Dimensionality

As the number of descriptive features grows the number of potential conditioning events grows. Consequently, an exponential increase is required in the size of the dataset as each new descriptive feature is added to ensure that for any conditional probability there are enough instances in the training dataset matching the conditions so that the resulting probability is reasonable.

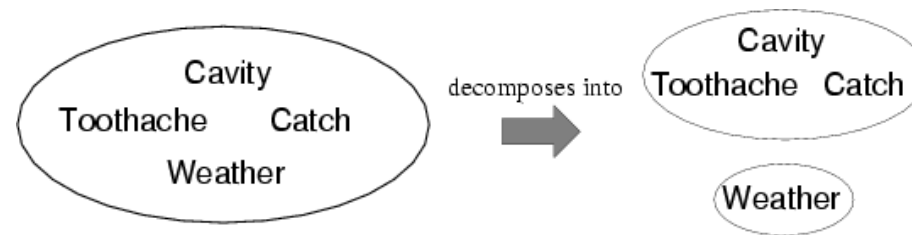
The probability of a patient who has a headache and a fever having meningitis should be greater than zero!

Our dataset is not large enough → our model is over-fitting to the training data. The concepts of conditional independence and factorization can help us overcome this flaw of our current approach.

CONDITIONAL INDEPENDENCE

Independence means that two or more events don't influence the probability of each other

- A and B are independent if and only if
 $P(A|B) = P(A)$ or $P(B|A) = P(B)$ or $P(A, B) = P(A)P(B)$



$$P(\text{Toothache}, \text{Catch}, \text{Cavity}, \text{Weather}) = P(\text{Toothache}, \text{Catch}, \text{Cavity}) P(\text{Weather})$$

-
- Absolute independence is powerful but rare
- Dentistry is a large field with hundreds of variables, none of which are independent. What to do?

Conditional independence means that A is independent of B – *if C is true*

- $P(\text{Toothache}, \text{Cavity}, \text{Catch})$ has $2^3 - 1 = 7$ independent entries
- If I have a cavity, the probability that the probe catches in it doesn't depend on whether I have a toothache:
 - (1) $P(\text{catch} | \text{toothache}, \text{cavity}) = P(\text{catch} | \text{cavity})$
- The same independence holds if I haven't got a cavity:
 - (2) $P(\text{catch} | \text{toothache}, \neg \text{cavity}) = P(\text{catch} | \neg \text{cavity})$
- *Catch* is **conditionally independent** of *Toothache* given *Cavity*:
 - $P(\text{Catch} | \text{Toothache}, \text{Cavity}) = P(\text{Catch} | \text{Cavity})$
- Equivalent statements:
 - $P(\text{Toothache} | \text{Catch}, \text{Cavity}) = P(\text{Toothache} | \text{Cavity})$
 - $P(\text{Toothache}, \text{Catch} | \text{Cavity}) = P(\text{Toothache} | \text{Cavity}) P(\text{Catch} | \text{Cavity})$

Conditional independence continued

- Write out full joint distribution using chain rule:

$$\begin{aligned} P(\textit{Toothache}, \textit{Catch}, \textit{Cavity}) &= P(\textit{Toothache} \mid \textit{Catch}, \textit{Cavity}) P(\textit{Catch}, \textit{Cavity}) \\ &= P(\textit{Toothache} \mid \textit{Catch}, \textit{Cavity}) P(\textit{Catch} \mid \textit{Cavity}) P(\textit{Cavity}) \\ &= P(\textit{Toothache} \mid \textit{Cavity}) P(\textit{Catch} \mid \textit{Cavity}) P(\textit{Cavity}) \end{aligned}$$

In most cases, the use of conditional independence reduces the size of the representation of the joint distribution from exponential in n to linear in n .
Conditional independence is our most basic and robust form of knowledge about uncertain environments

Assuming conditional independence simplifies the calculation of the prior probability

Without conditional independence

$$P(X, Y, Z | W) = P(X | W) \times P(Y | X, W) \times P(Z | Y, X, W) \times P(W)$$

With conditional independence

$$P(X, Y, Z | W) = \underbrace{P(X | W)}_{\text{Factor 1}} \times \underbrace{P(Y, W)}_{\text{Factor 2}} \times \underbrace{P(Z, W)}_{\text{Factor 3}} \times \underbrace{P(W)}_{\text{Factor 4}}$$

Conditional Independence

For two events, X and Y , that are conditionally independent given knowledge of a third events, here Z , the definition of the probability of a joint event and conditional probability are:

$$P(X|Y, Z) = P(X|Z)$$

$$P(X, Y|Z) = P(X|Z) \times P(Y|Z)$$

$$P(X|Y) = \frac{P(X, Y)}{P(Y)}$$

$$\begin{aligned} P(X, Y) &= P(X|Y) \times P(Y) \\ &= P(Y|X) \times P(X) \end{aligned}$$

X and Y are **dependent**

$$P(X|Y) = P(X)$$

$$P(X, Y) = P(X) \times P(Y)$$

X and Y are **independent**

Let's assume that headache, nausea and fever are independent of each other for patients with meningitis

$$P(q[1], \dots, q[m] | t = l) = P(q[1] | t = l) \times P(q[2] | q[1], t = l) \dots$$

$$P(q[1], \dots, q[m]) = P(q[1]) \times P(q[2] | q[1]) \times P(q[3] | q[2], q[1]) \dots$$

$$p(h, f, v, M) = p(M) \cdot p(h|M) \cdot p(f|h, M) \cdot p(v|h, f, M)$$

Assuming the descriptive features are conditionally independent of each other given MENINGITIS ($p(f|h, M) = p(f|h, \bar{M})$) we only need to store four factors:

- Factor1 : $\langle p(M) \rangle$ Factor2 : $\langle p(h|M), p(h|\bar{M}) \rangle$
- Factor3 : $\langle p(f|M), p(f|\bar{M}) \rangle$ Factor4 : $\langle p(v|M), p(v|\bar{M}) \rangle$

$$p(h, f, v, M) = p(M) \cdot p(h|M) \cdot p(f|M) \cdot p(v|M)$$

Calculate the factors from the data:

- Factor1 : $p(M) = \frac{3}{10} = 0.3$
- Factor2 : $p(h|M) = \frac{2}{3} = 0.6667$, $p(h|\bar{M}) = \frac{5}{7} = 0.7143$
- Factor3 : $p(f|M) = \frac{1}{3} = 0.3333$, $p(f|\bar{M}) = \frac{3}{7} = 0.4286$
- Factor4 : $p(v|M) = \frac{2}{3} = 0.6667$, $p(v|\bar{M}) = \frac{4}{7} = 0.5714$
- Let's calculate the probability of meningitis for a patient with headache and fever but no vomiting

$$p(M|\{h, f, \bar{v}\}) = \frac{p(h|M) \cdot p(f|M) \cdot p(\bar{v}|M) \cdot p(M)}{\sum_i p(h|M_i) \cdot p(f|M_i) \cdot p(\bar{v}|M_i) \cdot p(M_i)}$$

$$= \frac{0.6667 \cdot 0.3333 \cdot 0.3333 \cdot 0.3}{0.6667 \cdot 0.3333 \cdot 0.3333 \cdot 0.3 + 0.7143 \cdot 0.4286 \cdot 0.4286 \cdot 0.7}$$

$$= 0.1948$$

ID	HEADACHE	FEVER	VOMITING	MENINGITIS
1	true	true	false	false
2	false	true	false	false
3	true	false	true	false
4	true	false	true	false
5	false	true	false	true
6	true	false	true	false
7	true	false	true	false
8	true	false	true	true
9	false	true	false	false
10	true	false	true	true

This is an example of a *naïve Bayes model* – we assume conditional independence and form the products of the conditional probabilities

A standard way to write this is

$$p(C_k | x_1, x_2, \dots, x_n) = \frac{p(C_k) \prod_{i=1}^n p(x_i | C_k)}{\sum_k p(C_k) p(\mathbf{x} | C_k)}$$

where $\mathbf{x} = (x_1, x_2, \dots, x_n)$ are the specific values of the feature vector for which we want to predict the likelihood of outcome C_k

This is called a *naïve Bayes* model because it's a bit naïve to assume conditional independence – but it allows us to predict in cases where the joint distribution is sparse

SMOOTHING

Consider a joint distribution for a modestly sized dataset...

$P(\text{fr}) = 0.3$	$P(!\text{fr}) = 0.7$
$P(\text{CH} = \text{'none'} \mid \text{fr}) = 0.1666$	$P(\text{CH} = \text{'none'} \mid !\text{fr}) = 0$
$P(\text{CH} = \text{'paid'} \mid \text{fr}) = 0.1666$	$P(\text{CH} = \text{'paid'} \mid !\text{fr}) = 0.2857$
$P(\text{CH} = \text{'current'} \mid \text{fr}) = 0.5$	$P(\text{CH} = \text{'current'} \mid !\text{fr}) = 0.2857$
$P(\text{CH} = \text{'arrear'} \mid \text{fr}) = 0.1666$	$P(\text{CH} = \text{'arrear'} \mid !\text{fr}) = 0.4286$
$P(\text{GC} = \text{'none'} \mid \text{fr}) = 0.8334$	$P(\text{GC} = \text{'none'} \mid !\text{fr}) = 0.8571$
$P(\text{GC} = \text{'guarantor'} \mid \text{fr}) = 0.1666$	$P(\text{GC} = \text{'guarantor'} \mid !\text{fr}) = 0$
$P(\text{GC} = \text{'coapplicant'} \mid \text{fr}) = 0$	$P(\text{GC} = \text{'coapplicant'} \mid !\text{fr}) = 0.1429$
$P(\text{ACC} = \text{'own'} \mid \text{fr}) = 0.6666$	$P(\text{ACC} = \text{'own'} \mid !\text{fr}) = 0.7857$
$P(\text{ACC} = \text{'rent'} \mid \text{fr}) = 0.3333$	$P(\text{ACC} = \text{'rent'} \mid !\text{fr}) = 0.1429$
$P(\text{ACC} = \text{'free'} \mid \text{fr}) = 0$	$P(\text{ACC} = \text{'free'} \mid !\text{fr}) = 0.0714$

Let's calculate the probability of fraud for the case: CREDIT CH=paid, GC=guarantor, ACC=free

P(fr) = 0.3	P(!fr) = 0.7
P(CH = 'none' fr) = 0.1666	P(CH = 'none' !fr) = 0
P(CH = 'paid' fr) = 0.1666	P(CH = 'paid' !fr) = 0.2857
P(CH = 'current' fr) = 0.5	P(CH = 'current' !fr) = 0.2857
P(CH = 'arrear' fr) = 0.1666	P(CH = 'arrear' !fr) = 0.4286
P(GC = 'none' fr) = 0.8334	P(GC = 'none' !fr) = 0.8571
P(GC = 'guarantor' fr) = 0.1666	P(GC = 'guarantor' !fr) = 0
P(GC = 'coapplicant' fr) = 0	P(GC = 'coapplicant' !fr) = 0.1429
P(ACC = 'own' fr) = 0.6666	P(ACC = 'own' !fr) = 0.7857
P(ACC = 'rent' fr) = 0.3333	P(ACC = 'rent' !fr) = 0.1429
P(ACC = 'free' fr) = 0	P(ACC = 'free' !fr) = 0.0714

The naïve Bayes equation is

$$p(fr|\{pd, ga, fe\}) = \frac{p(pd|fr) \cdot p(ga|fr) \cdot p(fe|fr) \cdot p(fr)}{\sum_i p(pd|fr_i) \cdot p(ga|fr_i) \cdot p(fe|fr_i) \cdot p(fr_i)}$$

$$= \frac{(\prod_{k=1}^m p(q_k|fd))p(fd)}{\sum_{i=1}^I (\prod_{k=1}^m p(q_k|fd_i))p(fd_i)}$$

But:

$$(\prod_{k=1}^m p(q_k|fd))p(fd) = 0.0$$

$$(\prod_{k=1}^m p(q_k|!fd))p(!fd) = 0.0$$

So what do we do?

The problem is that the training set did not contain samples that represent all of the important combinations

- The (probably incorrect) conclusion is that these combinations have zero probability of occurring
- More likely, the training set is just too small (or incomplete)
- The standard way to avoid this issue is to use *smoothing*
- Smoothing takes some of the probability from the events with lots of the probability share and apportions it to the other probabilities in the set
- There are several different ways to smooth probabilities, we will show the use of *Laplacian smoothing*

Laplacian Smoothing (conditional probabilities)

$$p(f = v|t) = \frac{\text{count}(f = v|t) + k}{\text{count}(f|t) + (k \cdot |\text{Domain}(f)|)}$$

Raw	$P(GC = none !fr)$	=	0.8571
Probabilities	$P(GC = guarantor !fr)$	=	0
	$P(GC = coapplicant !fr)$	=	0.1429
Smoothing	k	=	3
Parameters	$count(GC !fr)$	=	14
	$count(GC = none !fr)$	=	12
	$count(GC = guarantor !fr)$	=	0
	$count(GC = coapplicant !fr)$	=	2
	$ Domain(GC) $	=	3
Smoothed	$P(GC = none !fr) = \frac{12+3}{14+(3 \times 3)}$	=	0.6522
Probabilities	$P(GC = guarantor !fr) = \frac{0+3}{14+(3 \times 3)}$	=	0.1304
	$P(GC = coapplicant !fr) = \frac{2+3}{14+(3 \times 3)}$	=	0.2174

Smoothing the posterior probabilities for the GUARANTOR/COAPPLICANT feature conditioned on FRAUDULENT being False.

$P(fr) = 0.3$	$P(\neg fr) = 0.7$
$P(CH = none fr) = 0.2222$	$P(CH = none \neg fr) = 0.1154$
$P(CH = paid fr) = 0.2222$	$P(CH = paid \neg fr) = 0.2692$
$P(CH = current fr) = 0.3333$	$P(CH = current \neg fr) = 0.2692$
$P(CH = arrears fr) = 0.2222$	$P(CH = arrears \neg fr) = 0.3462$
$P(GC = none fr) = 0.5333$	$P(GC = none \neg fr) = 0.6522$
$P(GC = guarantor fr) = 0.2667$	$P(GC = guarantor \neg fr) = 0.1304$
$P(GC = coapplicant fr) = 0.2$	$P(GC = coapplicant \neg fr) = 0.2174$
$P(ACC = own fr) = 0.4667$	$P(ACC = own \neg fr) = 0.6087$
$P(ACC = rent fr) = 0.3333$	$P(ACC = rent \neg fr) = 0.2174$
$P(ACC = Free fr) = 0.2$	$P(ACC = Free \neg fr) = 0.1739$

The Laplacian smoothed, with $k = 3$, probabilities needed by a Naive Bayes prediction model calculated from the fraud detection dataset. Notation key: FR=FRAUDULENT, CH=CREDIT HISTORY, GC = GUARANTOR/COAPPLICANT, ACC = ACCOMODATION, T='True', F='False'.

Compare the unsmoothed and smoothed probabilities;

$$p(f = v|t) = \frac{\text{count}(f = v|t) + k}{\text{count}(f|t) + (k \cdot |\text{Domain}(f)|)}; \quad k = 3$$

	unsmoothed	smoothed			unsmoothed	smoothed
$P(\text{fr}) =$	0.3	0.3	$P(!\text{fr}) =$		0.7	0.7
$P(\text{CH} = \text{'none'} \text{fr}) =$	0.1666	0.2222	$P(\text{CH} = \text{'none'} !\text{fr}) =$		0	0.1154
$P(\text{CH} = \text{'paid'} \text{fr}) =$	0.1666	0.2222	$P(\text{CH} = \text{'paid'} !\text{fr}) =$		0.2857	0.2692
$P(\text{CH} = \text{'current'} \text{fr}) =$	0.5	0.3333	$P(\text{CH} = \text{'current'} !\text{fr}) =$		0.2857	0.2692
$P(\text{CH} = \text{'arrear'} \text{fr}) =$	0.1666	0.2222	$P(\text{CH} = \text{'arrear'} !\text{fr}) =$		0.4286	0.3462
$P(\text{GC} = \text{'none'} \text{fr}) =$	0.8334	0.5333	$P(\text{GC} = \text{'none'} !\text{fr}) =$		0.8571	0.6522
$P(\text{GC} = \text{'guarantor'} \text{fr}) =$	0.1666	0.2667	$P(\text{GC} = \text{'guarantor'} !\text{fr}) =$		0	0.1304
$P(\text{GC} = \text{'coapplicant'} \text{fr}) =$	0	0.2	$P(\text{GC} = \text{'coapplicant'} !\text{fr}) =$		0.1429	0.2174
$P(\text{ACC} = \text{'own'} \text{fr}) =$	0.6666	0.4667	$P(\text{ACC} = \text{'own'} !\text{fr}) =$		0.7857	0.6087
$P(\text{ACC} = \text{'rent'} \text{fr}) =$	0.3333	0.3333	$P(\text{ACC} = \text{'rent'} !\text{fr}) =$		0.1429	0.2174
$P(\text{ACC} = \text{'free'} \text{fr}) =$	0	0.2	$P(\text{ACC} = \text{'free'} !\text{fr}) =$		0.0714	0.1739

Using smoothing, calculate the probability of fraud for the case: CREDIT CH=paid, GC=guarantor, ACC=free

P(fr) =	0.3	P(¬fr) =	0.7
P(CH = paid fr) =	0.2222	P(CH = paid ¬fr) =	0.2692
P(GC = guarantor fr) =	0.2667	P(GC = guarantor ¬fr) =	0.1304
P(ACC = Free fr) =	0.2	P(ACC = Free ¬fr) =	0.1739

$$\prod_{k=1}^m p(q_k | fr) \cdot p(fr) = 0.0036$$

$$\prod_{k=1}^m p(q_k | \neg fr) \cdot p(\neg fr) = 0.0043$$

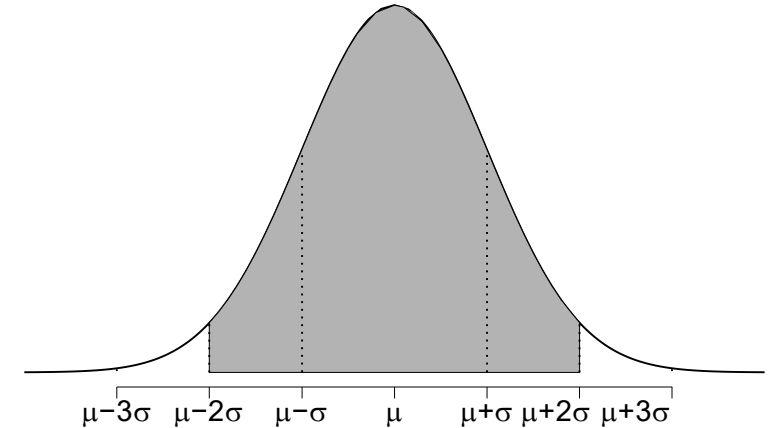
$$p(fr | \{pd, ga, fe\}) = \frac{(\prod_{k=1}^m p(q_k | fd))p(fd)}{\sum_{i=1}^I (\prod_{k=1}^m p(q_k | fd_i))p(fd_i)} = \frac{0.0036}{0.0036 + 0.0043} = 0.4557$$

CONTINUOUS FEATURES: USE OF PDFS

A probability density function (PDF) represents the probability distribution of a continuous feature as a mathematical function

One example is the normal (Gaussian) distribution:

$$N(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



A PDF defines a density curve; the shape of the curve is determined by: the statistical distribution that is used to define the PDF - the values of the statistical distribution parameters.

Many PDFs have been defined that are useful because they resemble real-world probabilities

Normal
 $x \in \mathbb{R}$
 $\mu \in \mathbb{R}$
 $\sigma \in \mathbb{R}_{>0}$

$$N(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x - \mu)^2}{2\sigma^2}}$$

Student-t
 $x \in \mathbb{R}$
 $\phi \in \mathbb{R}$
 $\rho \in \mathbb{R}_{>0}$
 $\kappa \in \mathbb{R}_{>0}$
 $z = \frac{x - \phi}{\rho}$

$$\tau(x, \phi, \rho, \kappa) = \frac{\Gamma(\frac{\kappa+1}{2})}{\Gamma(\frac{\kappa}{2}) \times \sqrt{\pi\kappa} \times \rho} \times \left(1 + \left(\frac{1}{\kappa} \times z^2\right)\right)^{-\frac{\kappa+1}{2}}$$

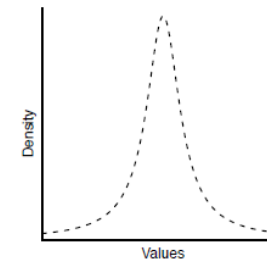
Exponential
 $x \in \mathbb{R}$
 $\lambda \in \mathbb{R}_{>0}$

$$E(x, \lambda) = \begin{cases} \lambda e^{-\lambda x} & \text{for } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

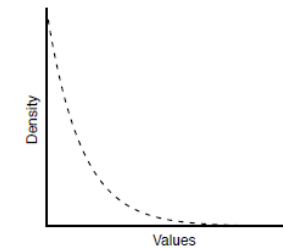
Mixture of n Gaussians
 $x \in \mathbb{R}$

$\{\mu_1, \dots, \mu_n | \mu_i \in \mathbb{R}\}$
 $\{\sigma_1, \dots, \sigma_n | \sigma_i \in \mathbb{R}_{>0}\}$
 $\{\omega_1, \dots, \omega_n | \omega_i \in \mathbb{R}_{>0}\}$
 $\sum_{i=1}^n \omega_i = 1$

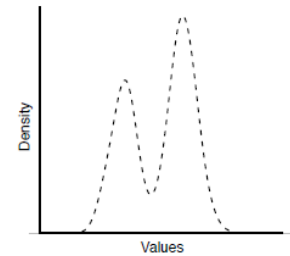
$$N(x, \mu_1, \sigma_1, \omega_1, \dots, \mu_n, \sigma_n, \omega_n) = \sum_{i=1}^n \frac{\omega_i}{\sigma_i\sqrt{2\pi}} e^{-\frac{(x - \mu_i)^2}{2\sigma_i^2}}$$



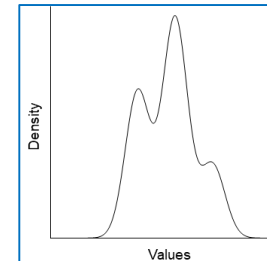
(a) Normal/Student-t



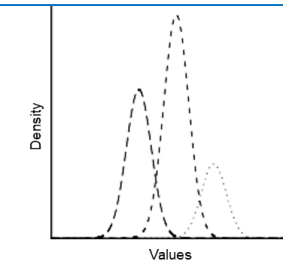
(b) Exponential



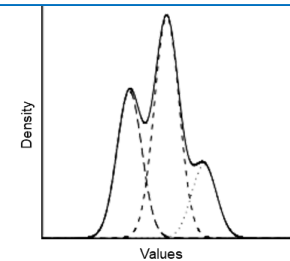
(c) Mixture of Gaussians



(a)



(b)



(c)

Illustration of how a mixture of Gaussians model is composed of several normal distributions.

To illustrate how PDFs can be used in Naive Bayes models, extend our loan application fraud detection query to have an ACCOUNT BALANCE feature

ID	CREDIT HISTORY	GUARANTOR/COAPPLICANT	ACCOMMODATION	ACCOUNT BALANCE	FRAUD
1	current	none	own	56.75	true
2	current	none	own	1,800.11	false
3	current	none	own	1,341.03	false
4	paid	guarantor	rent	749.50	true
5	arrears	none	own	1,150.00	false
6	arrears	none	own	928.30	true
7	current	none	own	250.90	false
8	arrears	none	own	806.15	false
9	current	none	rent	1,209.02	false
10	none	none	own	405.72	true
11	current	coapplicant	own	550.00	false
12	current	none	free	223.89	true
13	current	none	rent	103.23	true
14	paid	none	own	758.22	false
15	arrears	none	own	430.79	false
16	current	none	own	675.11	false
17	arrears	coapplicant	rent	1,657.20	false
18	arrears	none	free	1,405.18	false
19	arrears	none	own	760.51	false
20	current	none	own	985.41	false

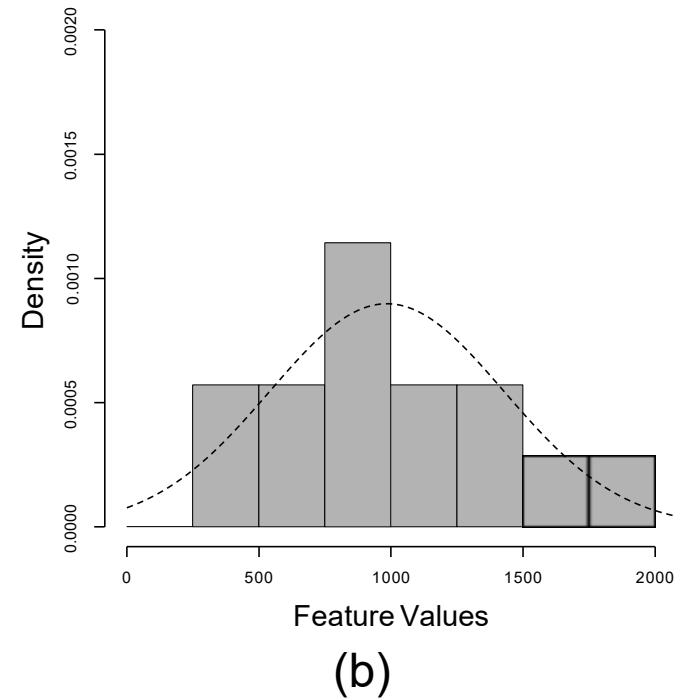
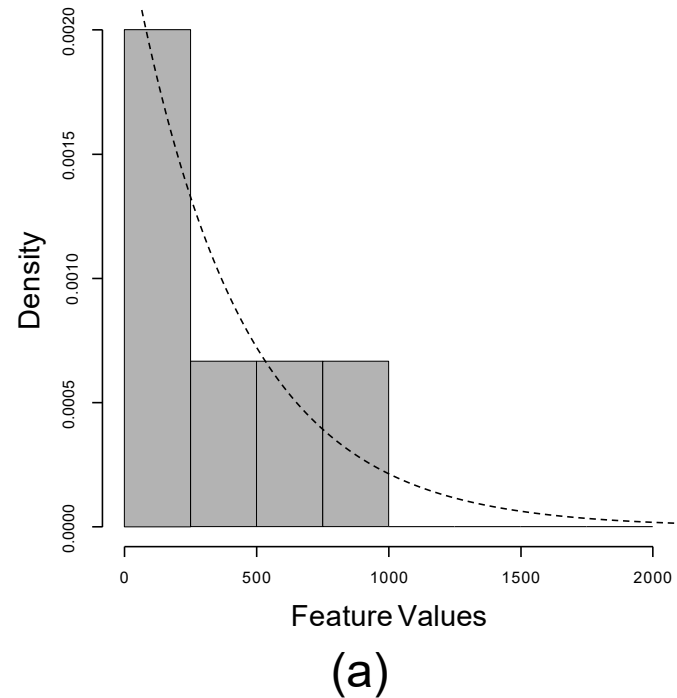
We need to define two PDFs for the new ACCOUNT BALANCE (AB) feature with each PDF conditioned on a different value in the domain or the target:

$$p(AB = X|fr) = \text{pdf}_1$$

$$p(AB = X|fr) = \text{pdf}_2$$

Note that these two PDFs do not have to be defined using the same statistical distribution.

Next, we find PDFs that fit well to the observed distributions (choice of 250-wide bins is heuristic)

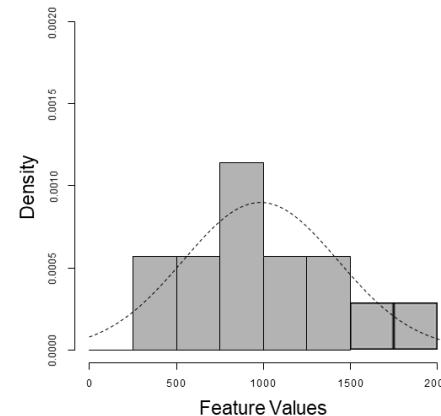
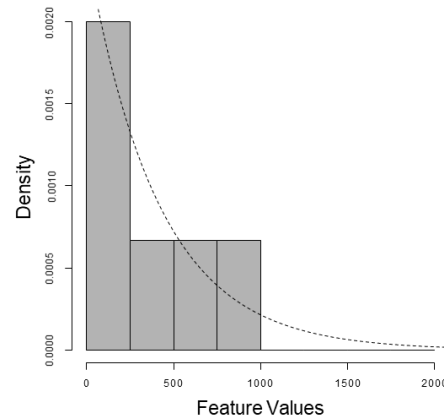


Histograms, using a bin size of 250 units, and density curves for the ACCOUNT BALANCE feature: (a) the fraudulent instances overlaid with a fitted exponential distribution; (b) the non-fraudulent instances overlaid with a fitted normal distribution.

Once we conclude that the PDFs are exponential and normal, we estimate the distribution parameters

To fit the exponential distribution we simply compute the sample mean \bar{x} of the ACCOUNT BALANCE feature in the set of instances where FRAUDULENT='True' and set $\lambda = \frac{1}{\bar{x}}$.

To fit the normal distribution to the set of instances where FRAUDULENT='False' we simply compute the sample mean and sample standard deviation, s , for the ACCOUNT BALANCE feature for this set of instances and set the parameters of the normal distribution to these values.



From these PDFs we can calculate the modeled
ACCOUNT BALANCE feature across the dataset

ACCOUNT			
ID	...	BALANCE	FRAUD
1		56.75	true
4		749.50	true
6		928.30	true
10	...	405.72	true
12		223.89	true
13		103.23	true
\overline{AB}		411.22	
$\lambda = 1! / \overline{AB}$		0.0024	

ACCOUNT			
ID	...	BALANCE	FRAUD
2		1 800.11	false
3		1 341.03	false
5		1 150.00	false
7		250.90	false
8		806.15	false
9		1 209.02	false
11		550.00	false
14		758.22	false
15		430.79	false
16		675.11	false
17		1 657.20	false
18		1 405.18	false
19		760.51	false
20		985.41	false
\overline{AB}		984.26	
$sd(AB)$		460.94	

We extend the conditional probabilities (Laplace smoothing, $k=3$) by adding the PDF fits to the ACCOUNT BALANCE feature

$P(fr)$	=	0.3	$P(\neg fr)$	=	0.7
$P(CH = none fr)$	=	0.2222	$P(CH = none \neg fr)$	=	0.1154
$P(CH = paid fr)$	=	0.2222	$P(CH = paid \neg fr)$	=	0.2692
$P(CH = current fr)$	=	0.3333	$P(CH = current \neg fr)$	=	0.2692
$P(CH = arrears fr)$	=	0.2222	$P(CH = arrears \neg fr)$	=	0.3462
$P(GC = none fr)$	=	0.5333	$P(GC = none \neg fr)$	=	0.6522
$P(GC = guarantor fr)$	=	0.2667	$P(GC = guarantor \neg fr)$	=	0.1304
$P(GC = coapplicant fr)$	=	0.2	$P(GC = coapplicant \neg fr)$	=	0.2174
$P(ACC = own fr)$	=	0.4667	$P(ACC = own \neg fr)$	=	0.6087
$P(ACC = rent fr)$	=	0.3333	$P(ACC = rent \neg fr)$	=	0.2174
$P(ACC = free fr)$	=	0.2	$P(ACC = free \neg fr)$	=	0.1739
$P(AB = x fr)$	$\approx E \left(\begin{matrix} x, \\ \lambda = 0.0024 \end{matrix} \right)$		$P(AB = x \neg fr)$	$\approx N \left(\begin{matrix} x, \\ \mu = 984.26, \\ \sigma = 460.94 \end{matrix} \right)$	

Calculate a prediction for the query
CH=paid, Ga/Ca=G, ACC=free, ACCTBAL=759.07

P(fr)	=	0.3	P(¬fr)	=	0.7
P(CH = paid fr)	=	0.2222	P(CH = paid ¬fr)	=	0.2692
P(GC = guarantor fr)	=	0.2667	P(GC = guarantor ¬fr)	=	0.1304
P(ACC = free fr)	=	0.2	P(ACC = free ¬fr)	=	0.1739
P(AB = x fr)	$\approx E \left(\begin{matrix} 759.07, \\ \lambda = 0.0024 \end{matrix} \right)$ $= 0.00039$		P(AB = x ¬fr)	$\approx N \left(\begin{matrix} 759.07, \\ \mu = 984.26, \\ \sigma = 460.94 \end{matrix} \right)$ $= 0.00077$	

- The product terms now include the information from the estimated PDFs

$$\prod_{k=1}^m p(q_k | fr) \cdot p(fr) = 0.0000014$$

$$\prod_{k=1}^m p(q_k | \neg fr) \cdot p(\neg fr) = 0.0000033$$

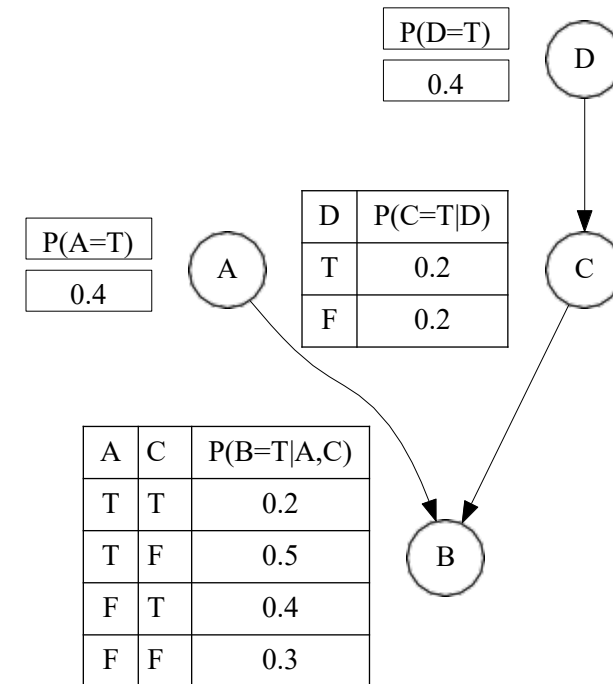
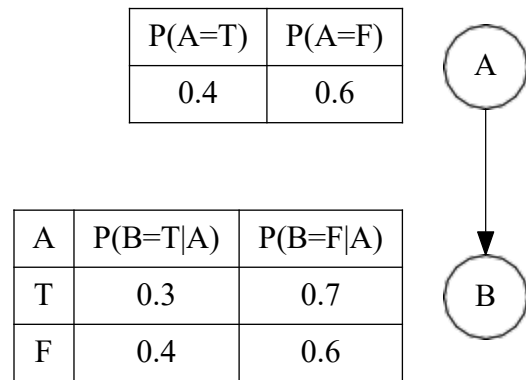
$$p(fr | \{pd, ga, fe, AB = 759.07\}) = \frac{(\prod_{k=1}^m p(q_k | fr))p(fr)}{\sum_{i=1}^I (\prod_{k=1}^m p(q_k | fd_i))p(fd_i)} = \frac{0.0000014}{0.0000014 + 0.0000033} = 0.2987$$

BAYESIAN NETWORKS

Bayesian networks use a graph-based representation to encode the structural relationships between subsets of features in a domain

- A Bayesian network representation is generally more compact than a full joint distribution
- Global conditional independence between all descriptive features is not required
- A Bayesian Network is a directed acyclical graph that is composed of three basic elements:
 - Nodes
 - Edges
 - Conditional probability tables (CPT)

Two simple Bayesian networks



- (a) A Bayesian network for a domain consisting of two binary features. The structure of the network states that the value of feature A directly influences the value of feature B.
- (b) A Bayesian network consisting of 4 binary features with a path containing 3 generations of nodes: D, C, and B.

P(A=T)	P(A=F)
0.4	0.6



A	P(B=T A)	P(B=F A)
T	0.3	0.7
F	0.4	0.6

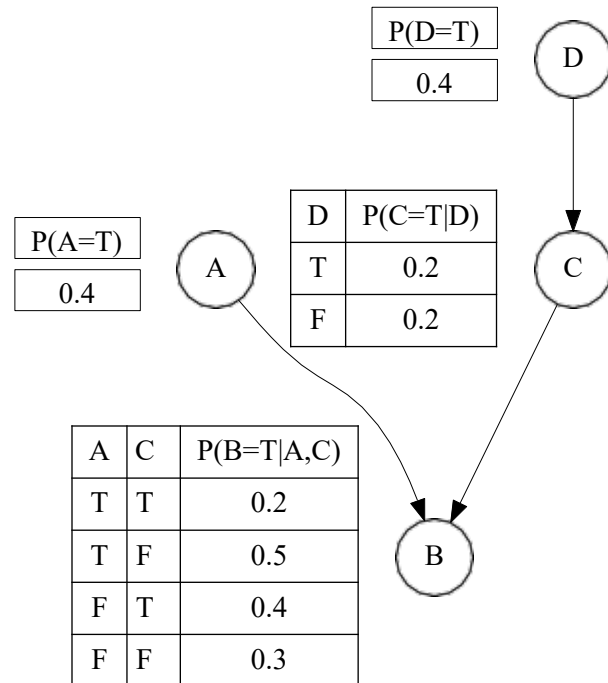
In probability terms the directed edge from A to B in this Bayesian network states that $p(A, B) = p(B|A) \cdot p(A)$

For example, the probability of the event a and $!b$ is

$$p(a, !b) = p(!b|a) \cdot p(a) = 0.7 \cdot 0.4 = 0.28$$

This can be generalized to show that for any network with N nodes, the probability of an event x_1, \dots, x_n can be computed using the following formula:

$$p(x_1, \dots, x_n) = \prod_{i=1}^n p(x_i | \text{Parents}(x_i))$$



For example, using the more complex Bayesian network here, we can calculate the probability of the joint event $p(a, !b, !c, d)$ as follows:

$$\begin{aligned}
 p(a, !b, !c, d) &= p(!b|[a, c]) \cdot p(!c|d) \cdot p(a) \cdot p(d) \\
 &= 0.5 \cdot 0.8 \cdot 0.4 \cdot 0.4 = 0.064
 \end{aligned}$$

We can use Bayes' Theorem to invert the dependencies between nodes in a network.

In the network here we can calculate $P(a|\neg b)$ as follows:

$$p(a|\neg b) = \frac{p(\neg b|a) \cdot p(a)}{p(\neg b)} = \frac{p(\neg b|a) \cdot p(a)}{\sum_i p(\neg b|A_i)}$$

$$= \frac{p(\neg b|a) \cdot p(a)}{p(\neg b|a) \cdot p(a) + p(\neg b|\neg a) \cdot p(\neg a)}$$

$$= \frac{0.7 \cdot 0.4}{(0.7 \cdot 0.4) + (0.6 \cdot 0.6)} = 0.4375$$

P(A=T)	P(A=F)
0.4	0.6

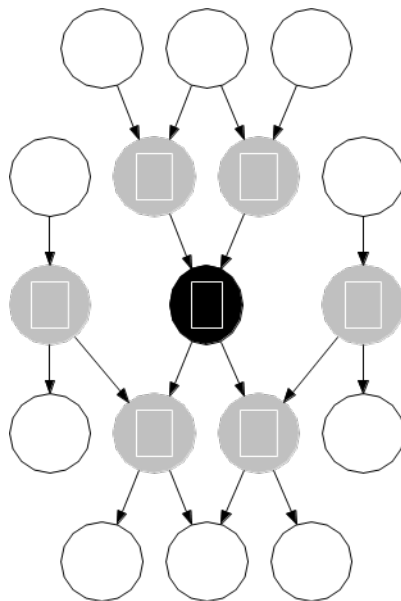
A	P(B=T A)	P(B=F A)
T	0.3	0.7
F	0.4	0.6



A Markov blanket for a given node is the set of other nodes whose state being known will produce conditional independence

For conditional independence we need to take into account not only the parents of a node but also the state of its children and their parents.

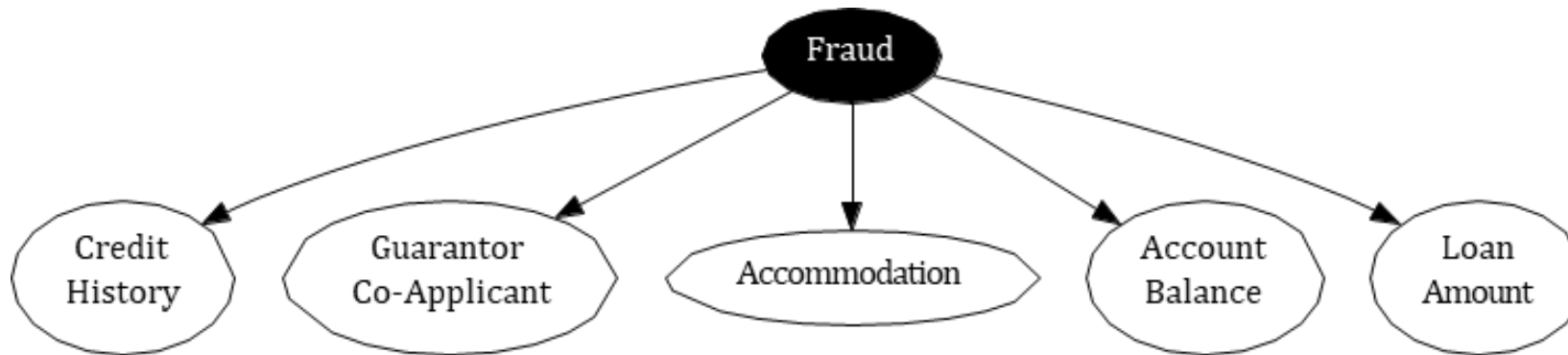
The set of nodes in a graph that make a node independent of the rest of the graph are known as the **Markov blanket** of a node.



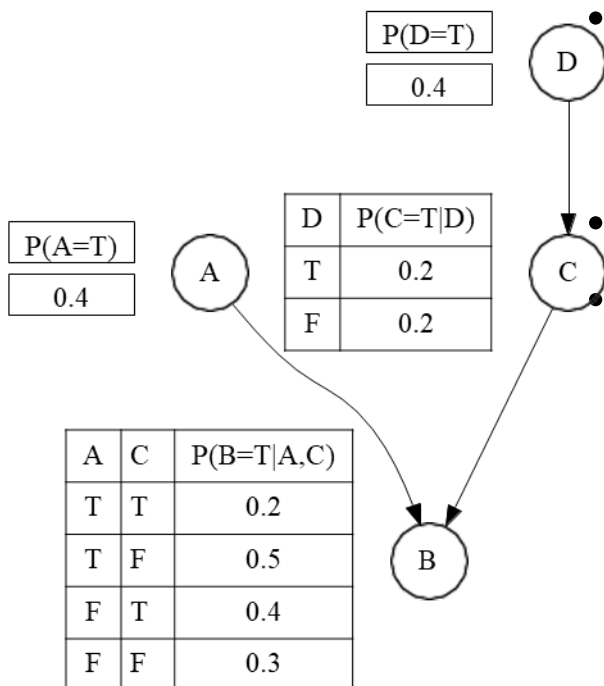
The gray nodes define the Markov blanket of the black node. The black node is conditionally independent of the white nodes given the state of the gray nodes.

A naïve Bayes classifier is a Bayesian network with a specific topological structure, showing the assumption of conditional independence

- The Bayesian network for the fraud dataset – assuming conditional independence
 - looks like this
 - The complete network would have the conditional probability tables



Computing a conditional probability for a node becomes more complex if the value of one or more of the parent nodes is unknown



For example, in this network, to compute $p(b|[a, D])$ where the status of node C is unknown we would do the following calculations:

Compute the distribution for C given D: $p(c|d) = 0.2$, $p(!c|d) = 0.8$

Compute $p(b|[a, C])$ by summing out C:

$$\begin{aligned}
 p(b|[a, C]) &= \sum_i \frac{p(b|[a, C_i])}{p(a, C_i)} \\
 &= \frac{(p(b|[a, c]) \cdot p(a) \cdot p(c)) + (p(b|[a, !c]) \cdot p(a) \cdot p(!c))}{(p(a) \cdot p(c)) + (p(a) \cdot p(!c))} \\
 &= \frac{(0.2 \cdot 0.4 \cdot 0.2) + (0.5 \cdot 0.4 \cdot 0.8)}{(0.4 \cdot 0.2) + (0.4 \cdot 0.8)} = 0.44
 \end{aligned}$$

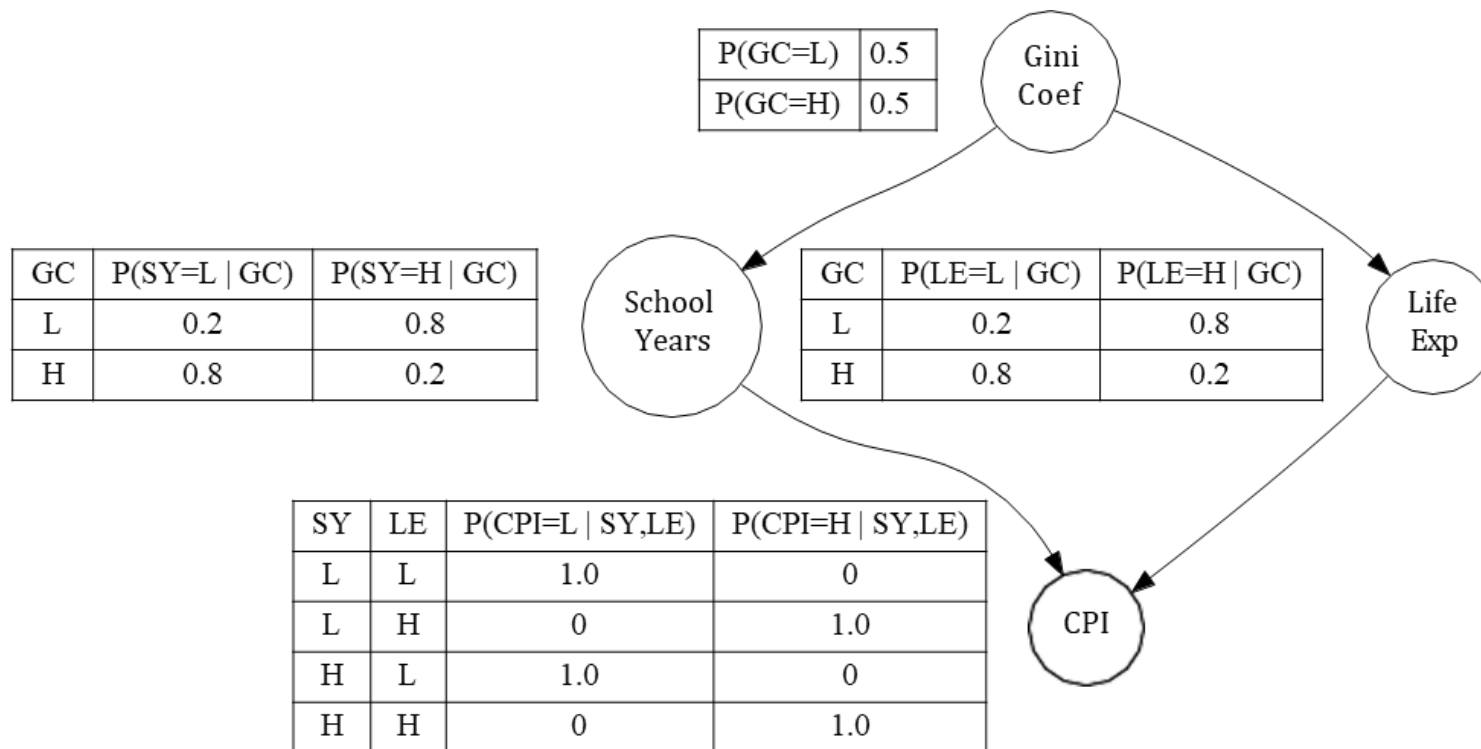
This example illustrates the power of Bayesian networks

- When complete knowledge of the state of all the nodes in the network is not available, we clamp the values of nodes that we do have knowledge of and sum out the unknown nodes.
- The simplest way to construct a Bayesian network is to use a hybrid approach where:
 - the topology of the network is given to the learning algorithm,
 - and the learning task involves inducing the CPT from the data.

Here is some socio-economic data for a set of countries, and a *binned* version of the same data (CPI, the target, is the *corruption perception index*)

COUNTRY ID	ORIGINAL DATA				BINNED DATA			
	GINI COEF	SCHOOL YEARS	LIFE EXP	CPI	GINI COEF	SCHOOL YEARS	LIFE EXP	CPI
Afghanistan	27.82	0.40	59.61	1.52	low	low	low	low
Argentina	44.49	10.10	75.77	3.00	high	low	low	low
Australia	35.19	11.50	82.09	8.84	low	high	high	high
Brazil	54.69	7.20	73.12	3.77	high	low	low	low
Canada	32.56	14.20	80.99	8.67	low	high	high	high
China	42.06	6.40	74.87	3.64	high	low	low	low
Egypt	30.77	5.30	70.48	2.86	low	low	low	low
Germany	28.31	12.00	80.24	8.05	low	high	high	high
Haiti	59.21	3.40	45.00	1.80	high	low	low	low
Ireland	34.28	11.50	80.15	7.54	low	high	high	high
Israel	39.2	12.50	81.30	5.81	low	high	high	high
New Zealand	36.17	12.30	80.67	9.46	low	high	high	high
Nigeria	48.83	4.10	51.30	2.45	high	low	low	low
Russia	40.11	12.90	67.62	2.45	high	high	low	low
Singapore	42.48	6.10	81.788	9.17	high	low	high	high
South Africa	63.14	8.50	54.547	4.08	high	low	low	low
Sweden	25.00	12.80	81.43	9.30	low	high	high	high
U.K.	35.97	13.00	80.09	7.78	low	high	high	high
U.S.A	40.81	13.70	78.51	7.14	high	high	high	high
Zimbabwe	50.10	6.7	53.684	2.23	high	low	low	low

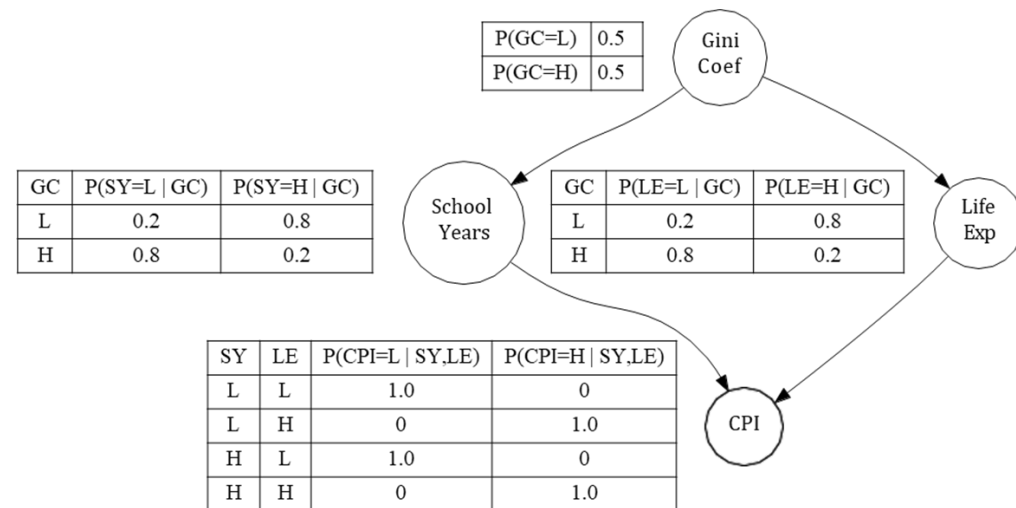
Represent this data in a Bayesian network



A Bayesian network that encodes the causal relationships between the features in the corruption domain. The CPT entries have been calculated using the data from the previous table (Table 16 in the book).

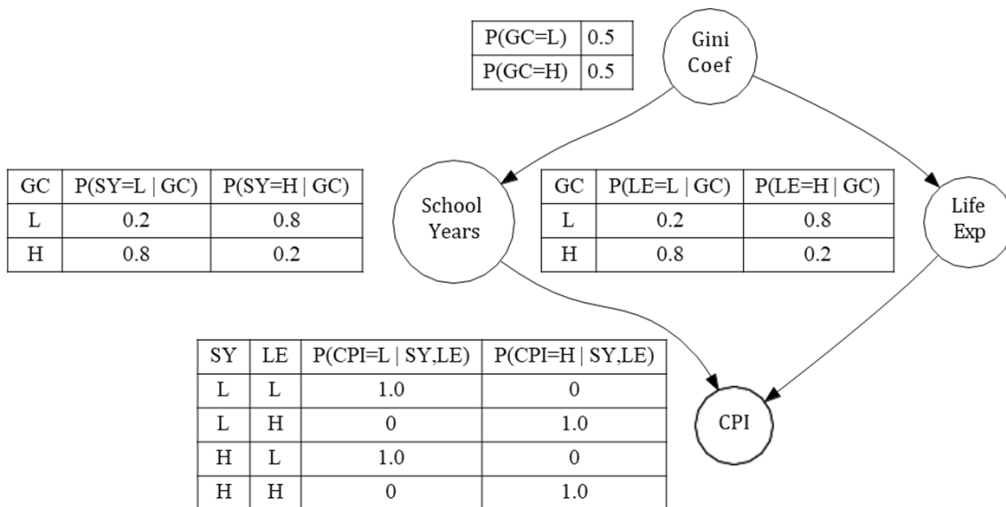
We wish to predict the CPI for a country with this profile:

GINI COEF = 'high', SCHOOL YEARS = 'high'



$$\begin{aligned}
 p(cpi = h | sy = h, gc = h) &= \frac{p(cpi = h, sy = h, gc = h)}{p(sy = h, gc = h)} \\
 &= \frac{\sum_{i \in H, L} p(cpi = h, sy = h, gc = h, le = i)}{p(sy = h, gc = h)} \\
 &= \sum_{i \in H, L} p(cpi = h, sy = h, gc = h, le = i) \\
 &= \sum_{i \in H, L} p(cpi = h | sy = h, le = i) \cdot p(le = i | gc = h) \cdot p(gc = h) \\
 &= (1.0 \cdot 0.2 \cdot 0.2 \cdot 0.5) + (0 \cdot 0.2 \cdot 0.8 \cdot 0.5) = 0.02
 \end{aligned}$$

Continuing the calculation for GINI COEF = 'high', SCHOOL YEARS = 'high'



$$\sum_{i \in H, L} p(cpi = h, sy = h, gc = h, le = i) = 0.02$$

$$p(sy = h, gc = h) = p(sy = h | gc = h) \cdot p(gc = h) = 0.2 \cdot 0.5 = 0.1$$

$$p(cpi = h | sy = h, gc = h) = \frac{p(cpi = h, sy = h, gc = h)}{p(sy = h, gc = h)} = \frac{0.02}{0.1} = 0.2$$

If GC=H and SY=H, the probability of CPI being high is 0.2

Practical considerations for Bayesian Networks

- Because of the calculation complexity that can arise when using Bayesian networks to do exact inference a popular approach is to approximate the required probability distribution using Markov Chain Monte Carlo algorithms.
- ***Gibbs sampling*** is one of the best known MCMC algorithms.
 1. Clamp the values of the evidence variables and randomly assign the values of the non-evidence variables.
 2. Generate samples by changing the value of one of the non-evidence variables using the distribution for the node conditioned on the state of the rest of the network.

Scikit-learn has a Naïve Bayes classifier object that assumes Gaussian distributions

- `class sklearn.naive_bayes.GaussianNB(*, priors=None, var_smoothing=1e-09)`
- `priors`: the prior probabilities can either be entered as arrays of data, or (default) calculated from the data itself
- `var_smoothing`: "Portion of the largest variance of all features that is added to variances for calculation stability." This is comparable to smoothing as we discussed (to handle sparse combinations of values).
- ```
>>> clf_pf = GaussianNB()
```
- ```
>>> clf_pf.partial_fit(X, Y, np.unique(Y))
```
- ```
GaussianNB()
```
- ```
>>> print(clf_pf.predict([[-0.8, -1]]))
```
- ```
[1]
```

# Here's an extremely simple example of a naïve Bayesian classifier in Python/scikit

```
-*- coding: utf-8 -*-
"""
```

```
Bayesian network
```

```
Created on Mon Mar 23 16:32:24 2020
```

```
@author: crjones4
```

```
"""
```

```
import pandas as pd
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.naive_bayes import GaussianNB
```

```
pathName = "C:\\\\Data\\"
```

```
dataFrame = pd.read_excel(pathName + 'iris.xlsx', sheet_name='data')
```

```
X = dataFrame.drop(["species"], axis=1)
```

```
Y = dataFrame.species
```

```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.5, random_state=0)
```

```
gnb = GaussianNB()
```

```
y_pred = gnb.fit(X_train, y_train).predict(X_test)
```

```
print("Number of mislabeled points out of a total %d points : %d"
```

```
 % (X_test.shape[0], (y_test != y_pred).sum()))
```

Number of mislabeled points out of a total 75 points : 3

## There is also a Naïve Bayes classifier object for use with categorical features

- `class sklearn.naive_bayes.CategoricalNB(*, *, alpha=1.0, fit_prior=True, class_prior=None, min_categories=None)[source]`
- *alpha*: an additive smoothing parameter
- *fit\_prior*: a Boolean indicating whether we assume uniform priors or priors calculated from the data
- *class\_prior*: the prior probabilities can either be entered as arrays of data, or (default) calculated from the data itself
- *min\_categories*: the minimum number of categories for each feature (int or array)

# Today's Objectives

- Bayesian Prediction
- Smoothing
- Continuous Features: Use of Probability Density Functions
- Bayesian Networks