Naïve Bayes

Slides are adapted from Sebastian Thrun (Udacity),

Ke Chen

Jonathan Huang and

H. Witten's and E. Frank's "Data Mining"
and Jeremy Wyatt,

Background

- There are three methods to establish a classifier

  a) Model a classification rule directly

  Examples: k-NN, decision trees, perceptron, SVM

  b) Model the probability of class memberships given input data

  Example: perceptron with the cross-entropy cost

  c) Make a probabilistic model of data within each class

  Examples: naive Bayes, model based classifiers

- *a*) and *b*) are examples of discriminative classification
- *c*) is an example of generative classification
- *b*) and *c*) are both examples of probabilistic classification

Probability Basics

- Prior, conditional and joint probability for random variables
  - Prior probability: $P(x)$
  - Conditional probability: $P(x_1 \mid x_2), P(x_2 \mid x_1)$
  - Joint probability: $\mathbf{x} = (x_1, x_2), P(\mathbf{x}) = P(x_1, x_2)$
  - Relationship: $P(x_1, x_2) = P(x_2 \mid x_1)P(x_1) = P(x_1 \mid x_2)P(x_2)$
  - Independence:

$$P(x_2 \mid x_1) = P(x_2), P(x_1 \mid x_2) = P(x_1), P(x_1, x_2) = P(x_1)P(x_2)$$

- Bayesian Rule

$$P(c \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid c)P(c)}{P(\mathbf{x})}$$

$$Posterior = \frac{Likelihood \times Prior}{Evidence}$$

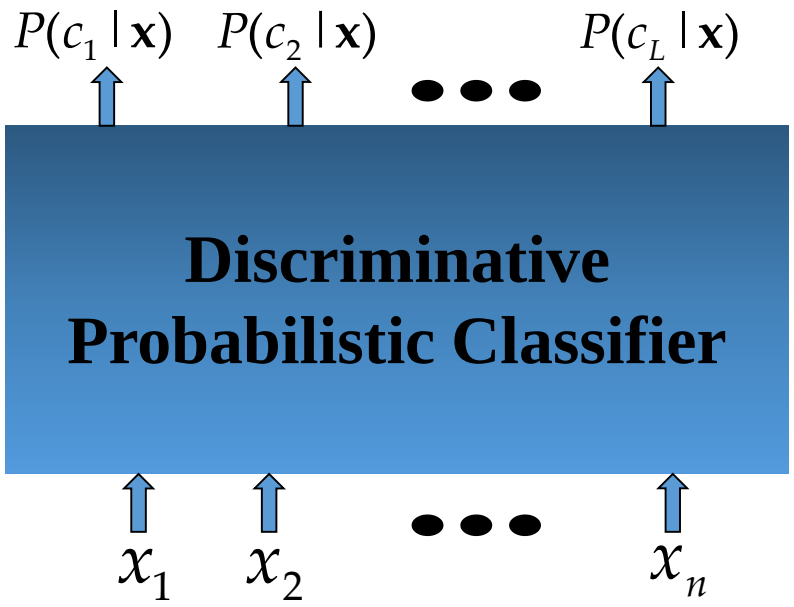
Discriminative

Generative

Probabilistic Classification

- Establishing a probabilistic model for classification
  - **Discriminative model**

$$P(c \mid \mathbf{x}) \quad c = c_1, \cdots, c_L, \mathbf{x} = (x_1, \cdots, x_n)$$



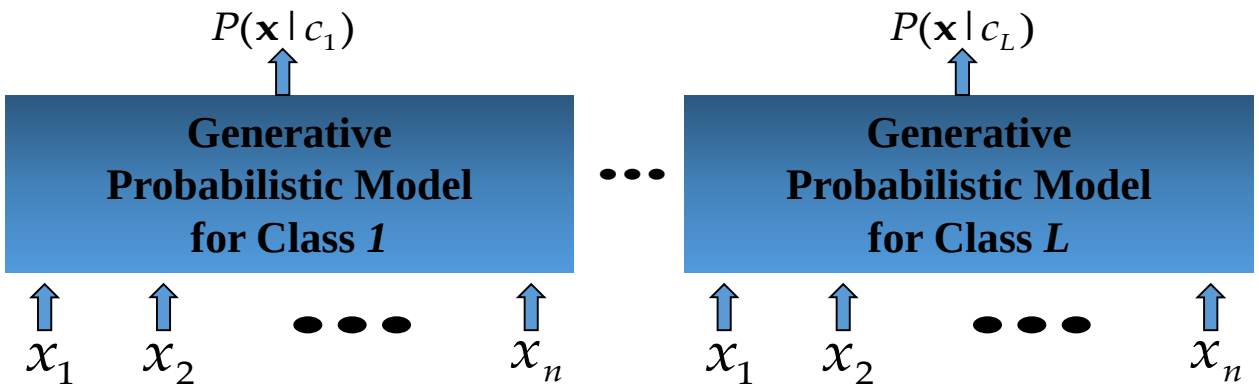
$$\mathbf{x} = (x_1, x_2, \cdots, x_n)$$

- To train a discriminative classifier regardless its probabilistic or non-probabilistic nature, all training examples of different classes must be jointly used to build up a single discriminative classifier.
- Output L probabilities for L class labels in a probabilistic classifier while a single label is achieved by a non-probabilistic classifier .

Probabilistic Classification

- Establishing a probabilistic model for classification (cont.)
  - **Generative model (must be probabilistic)**

$$P(\mathbf{x} \mid c) \quad c = c_1, \cdots, c_L, \mathbf{x} = (x_1, \cdots, x_n)$$



$$\mathbf{x} = (x_1, x_2, \cdots, x_n)$$

- L probabilistic models have to be trained independently
- Each is trained on only the examples of the same label
- Output L probabilities for a given input with L models
- "Generative" means that such a model produces data subject to the distribution via sampling.

Probabilistic Classification

- **M**aximum **A P**osterior (**MAP**) classification rule
  - For an input $\boldsymbol{x}$, find the largest one from L probabilities output by a discriminative probabilistic classifier $P(c_1 \mid \mathbf{x}), \ldots, P(c_L \mid \mathbf{x})$.
  - Assign $\boldsymbol{x}$ to label c^* if $P(c^* \mid \mathbf{x})$ is the largest.
- Generative classification with the MAP rule
  - Apply Bayesian rule to convert them into posterior probabilities

$$P(c_i \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid c_i)P(c_i)}{P(\mathbf{x})} \propto P(\mathbf{x} \mid c_i)P(c_i)$$

Common factor for all L probabilities

$$\text{for } i = 1, 2, \cdots, L$$

  - Then apply the MAP rule to assign a label

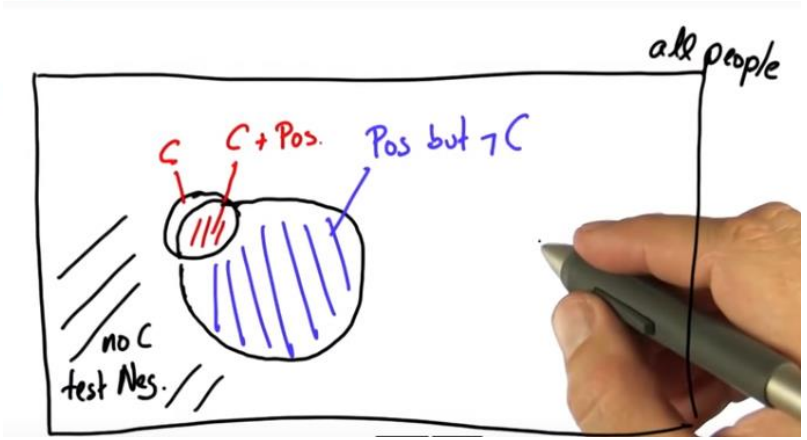
EXAMPLE $P(C) = 0.01$

TEST: 90% it is positive if you have C. SENSITIVITY

90% it is negative if you don't have C. SPECITIVITY

QUESTION: TEST = POSITIVE

PROBABILITY OF HAVING CANCER



prior: $P(C) = 0.01 = 1\%$ $\quad P(\neg C) = 0.99$

$P(Pos|C) = 0.9 = 90\%$ $\quad P(Pos|\neg C) = 0.1$

$P(Neg|\neg C) = 0.9$

joint: $P(C, Pos) = P(C) \cdot P(Pos|C) = 0.009$

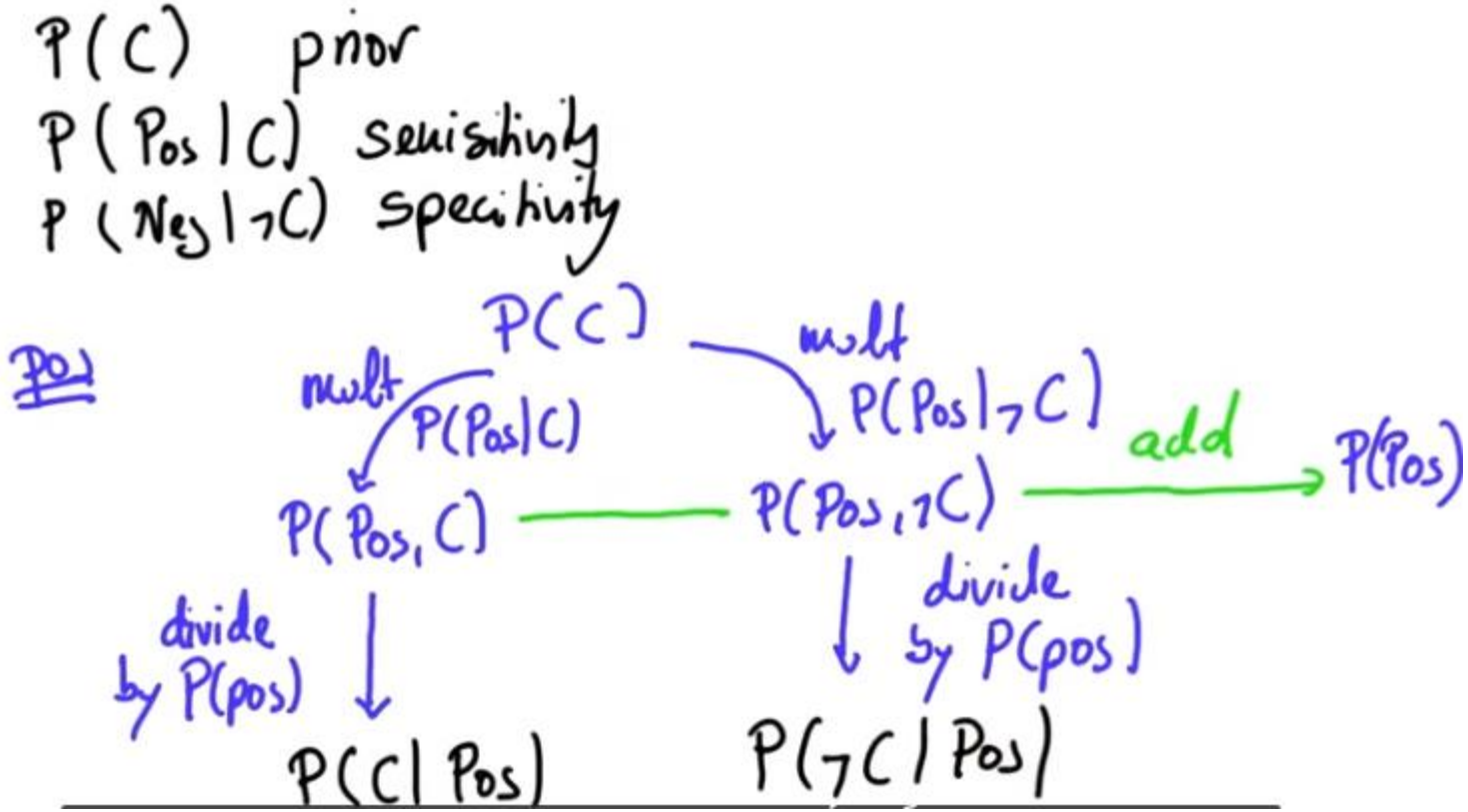
$P(\neg C, Pos) = P(\neg C) \cdot P(Pos|\neg C) = 0.099$

normalizer $P(Pos) = P(C, Pos) + P(\neg C, Pos) = 0.108$

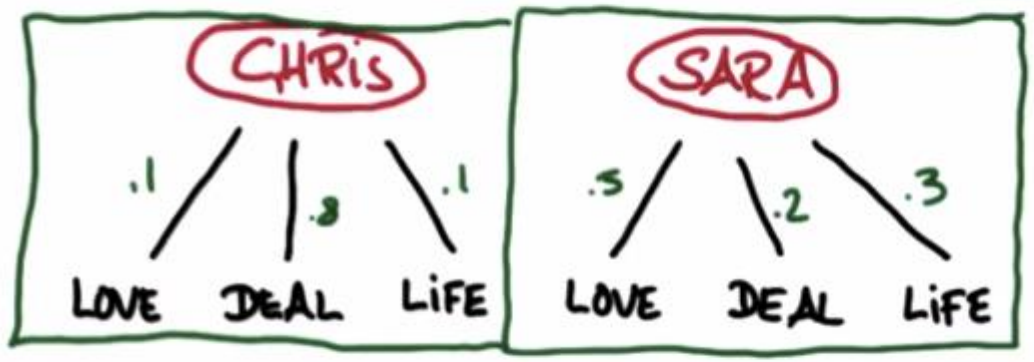
posterior: $P(C|Pos) = 0.0833$

$P(\neg C|Pos) = 0.9167$

$\Big\} = \boxed{1}$


P(C) prior
P(Pos|C) sensitivity
P(¬Neg|¬C) specificity
P(C)
mult
mult
P(Pos|C)
P(Pos|¬C)
add
P(Pos)
P(Pos, C)
P(Pos, ¬C)
divide by P(pos)
divide by P(pos)
P(C|Pos)
P(¬C|Pos)

TEXT LEARNING — NAIVE BAYES



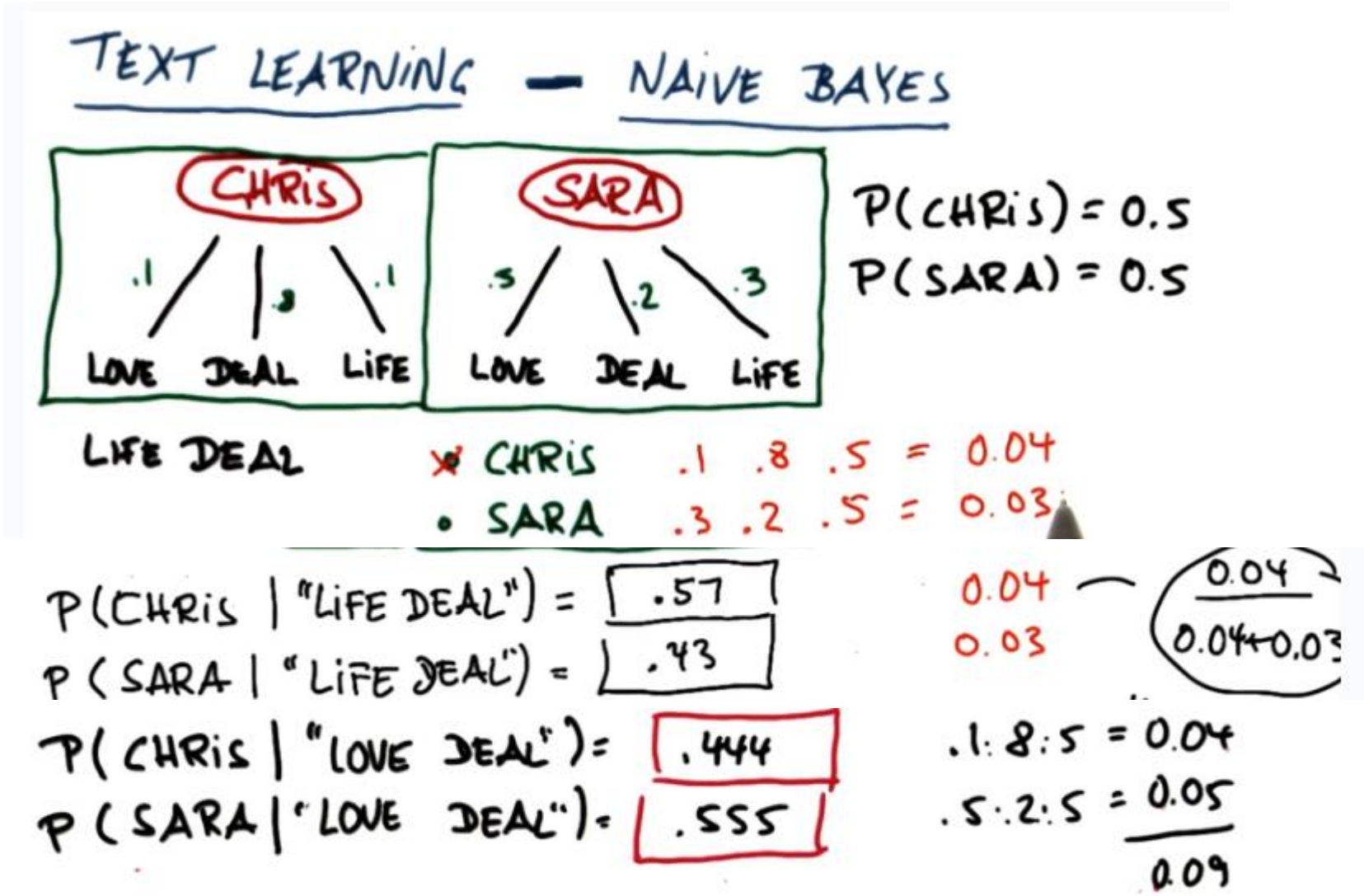
P(CHRIS) = 0.5

P(SARA) = 0.5

LOVE LIFE !

- CHRIS
- SARA

Naïve Bayes


TEXT LEARNING — NAIVE BAYES
CHRIS
.1
.8
.1
LOVE
DEAL
LIFE
SARA
.5
.2
.3
LOVE
DEAL
LIFE
P(CHRIS) = 0.5
P(SARA) = 0.5
LIFE DEAL
CHRIS .1 .8 .5 = 0.04
SARA .3 .2 .5 = 0.03
P(CHRIS | "LIFE DEAL") = .57
P(SARA | "LIFE DEAL") = .43
P(CHRIS | "LOVE DEAL") = .444
P(SARA | "LOVE DEAL") = .555
0.04
0.03
0.04
0.04+0.03
.1 .8 .5 = 0.04
.5 .2 .5 = 0.05
0.09

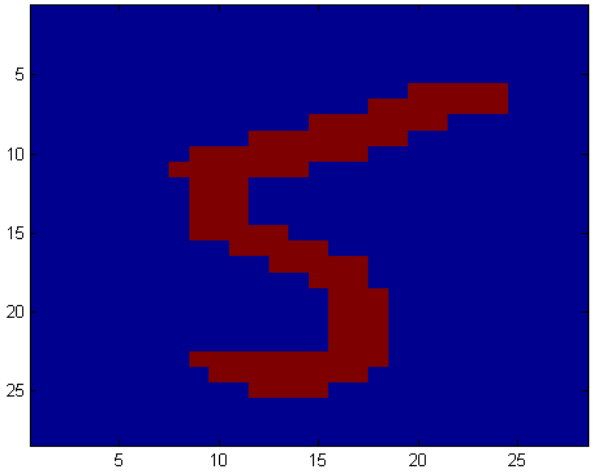
Things We'd Like to Do

- Spam Classification
  - Given an email, predict whether it is spam or not
- Medical Diagnosis
  - Given a list of symptoms, predict whether a patient has disease X or not
- Weather
  - Based on temperature, humidity, etc… predict if it will rain tomorrow

Bayesian Classification

Problem statement:
Given features $X_1, X_2, \ldots, X_n$
Predict a label Y

- **Digit Recognition**



5

- **$X_1, \ldots, X_n \in \{0,1\}$ (Black vs. White pixels)**
- **$Y \in \{5,6\}$ (predict whether a digit is a 5 or a 6)**

The Bayes Classifier

- we saw that a good strategy is to predict:

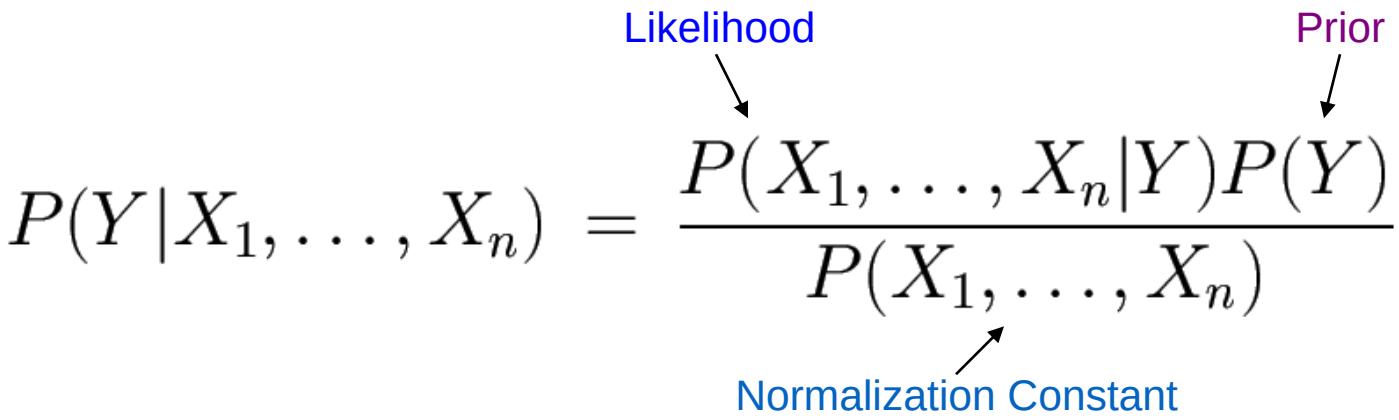
$$\arg\max_Y P(Y|X_1,\ldots,X_n)$$

  - (for example: what is the probability that the image represents a 5 given its pixels?)

- So … How do we compute that?

The Bayes Classifier

- Use Bayes Rule!



- Why did this help? Well, we think that we might be able to specify how features are "generated" by the class label

The Bayes Classifier

- Let's expand this for our digit recognition task:

$$P(Y=5|X_1,\ldots,X_n) = \frac{P(X_1,\ldots,X_n|Y=5)P(Y=5)}{P(X_1,\ldots,X_n|Y=5)P(Y=5)+P(X_1,\ldots,X_n|Y=6)P(Y=6)}$$

$$P(Y=6|X_1,\ldots,X_n) = \frac{P(X_1,\ldots,X_n|Y=6)P(Y=6)}{P(X_1,\ldots,X_n|Y=5)P(Y=5)+P(X_1,\ldots,X_n|Y=6)P(Y=6)}$$

- To classify, we'll simply compute these two probabilities and predict based on which one is greater

Model Parameters

- For the Bayes classifier, we need to "learn" two functions, the likelihood and the prior

- How many parameters are required to specify the prior for our digit recognition example?
- (Supposing that each image is 30x30 pixels)

- The problem with explicitly modeling $P(X_1,\ldots,X_n|Y)$ is that there are usually way too many parameters:
  - We'll run out of space
  - We'll run out of time
  - And we'll need tons of training data (which is usually not available)

The Naïve Bayes Model

- The *Naïve Bayes Assumption*: Assume that all features are independent **given the class label Y**
- Equationally speaking:

$$P(X_1, \ldots, X_n|Y) = \prod_{i=1}^{n} P(X_i|Y)$$

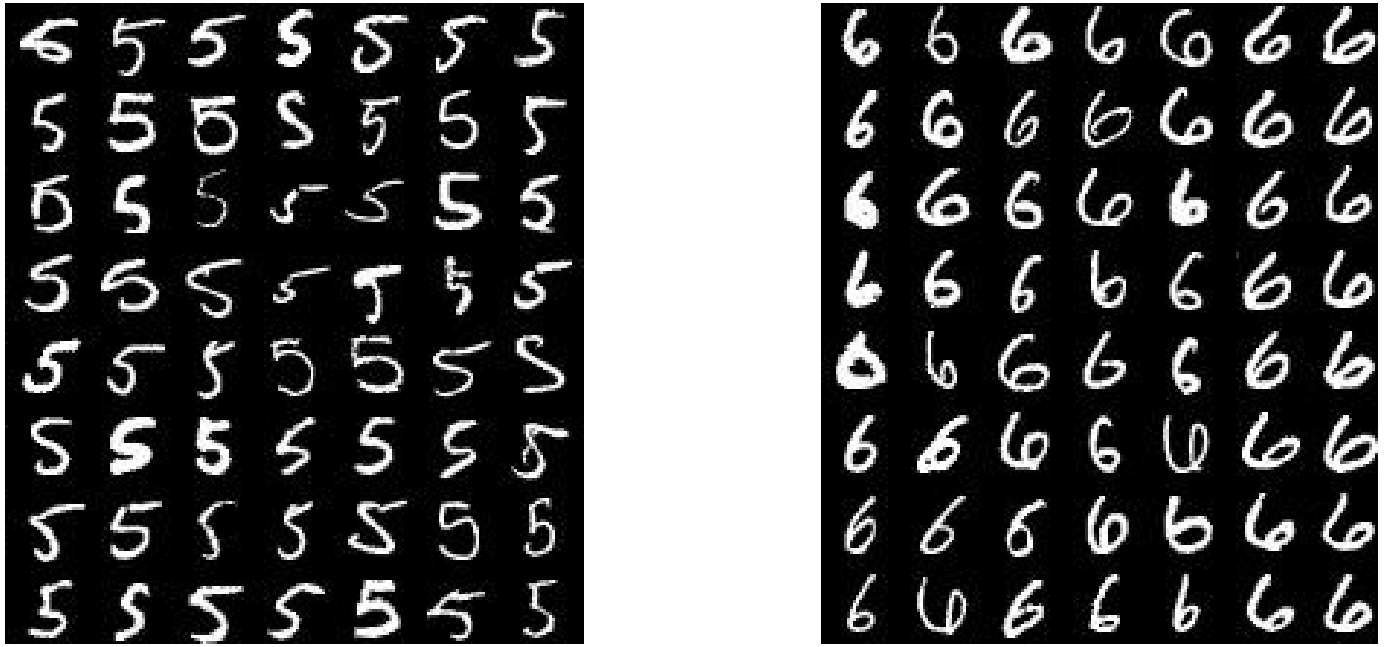
- (We will discuss the validity of this assumption later)

Why is this useful?

- # of parameters for modeling $P(X_1,\ldots,X_n|Y)$:
  - $2(2^n-1)$
- # of parameters for modeling $P(X_1|Y),\ldots,P(X_n|Y)$
  - $2n$

Naïve Bayes Training

- Now that we've decided to use a Naïve Bayes classifier, we need to train it with some data:



MNIST Training Data

Naïve Bayes Training

- Training in Naïve Bayes is **easy**:
  - Estimate P(Y=v) as the fraction of records with Y=v

$$P(Y = v) = \frac{Count(Y = v)}{\#\ records}$$

  - Estimate P(X_i=u|Y=v) as the fraction of records with Y=v for which X_i=u

$$P(X_i = u|Y = v) = \frac{Count(X_i = u \wedge Y = v)}{Count(Y = v)}$$

- (This corresponds to Maximum Likelihood estimation of model parameters)

Naïve Bayes Training

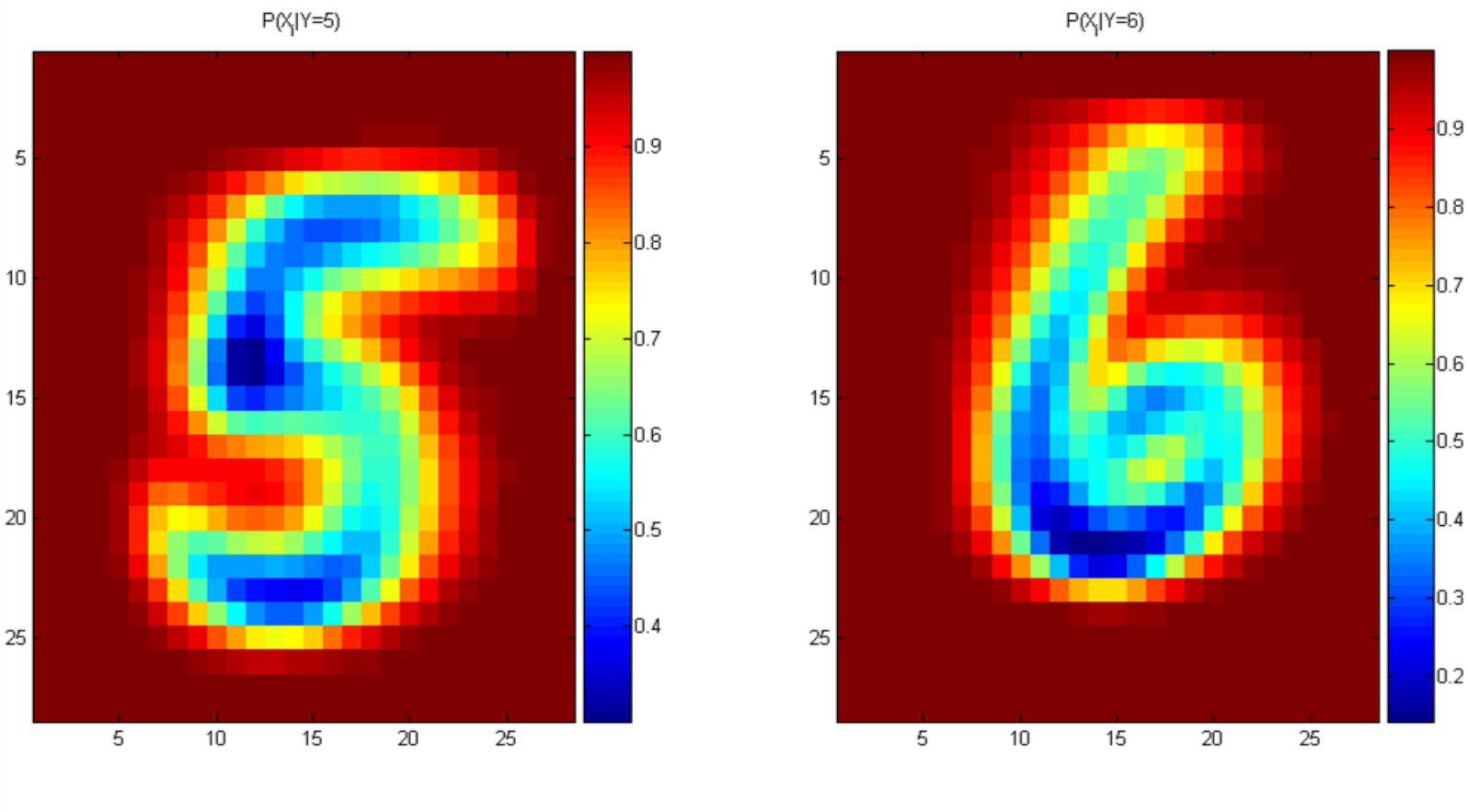
- In practice, some of these counts can be zero
- Fix this by adding “virtual” counts:

$$P(X_i = u | Y = v) = \frac{Count(X_i = u \wedge Y = v) + 1}{Count(Y = v) + 2}$$

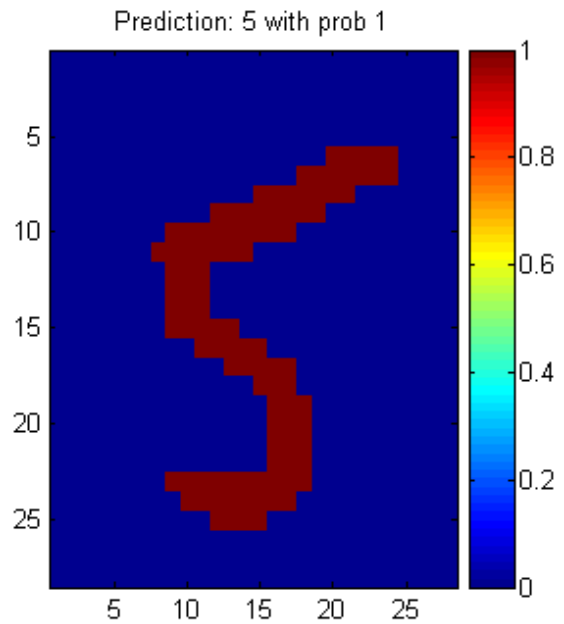
  - (This is like putting a prior on parameters and doing MAP estimation instead of MLE)
  - This is called *Smoothing*

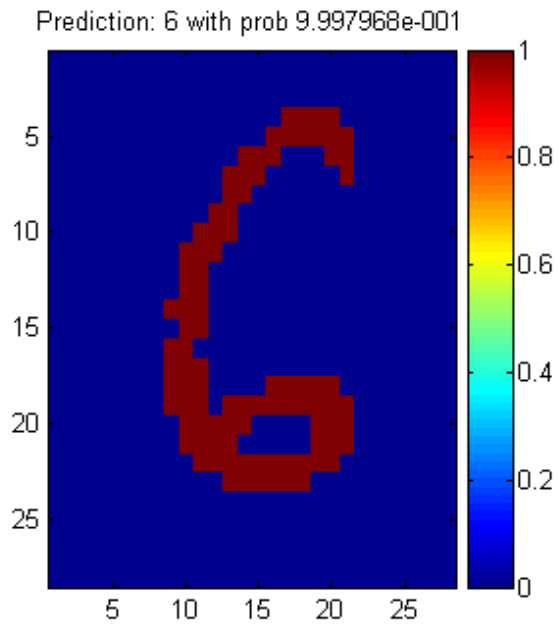
Naïve Bayes Training

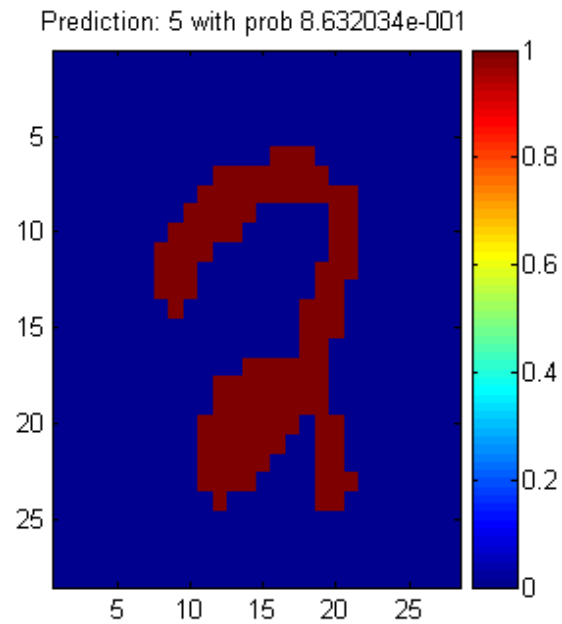
- For binary digits, training amounts to averaging all of the training fives together and all of the training sixes together.



Naïve Bayes Classification







Naïve Bayes Assumption

- Recall the Naïve Bayes assumption:
  - that all features are independent **given the class label Y**
- For an example where conditional independence fails:
  - Y=XOR(X_1,X_2)

X_1	X_2	P(Y=0| X_1,X_2)	P(Y=1| X_1,X_2)
0	0	1	0
0	1	0	1
1	0	0	1
1	1	1	0

Actually, the Naïve Bayes assumption is almost never true

Still… Naïve Bayes often performs surprisingly well even when its assumptions do not hold

Naïve Bayes

- Bayes classification

$$P(c \mid \mathbf{x}) \propto P(\mathbf{x} \mid c)P(c) = P(x_1, \cdots, x_n \mid c)P(c) \text{ for } c = c_1, ..., c_L.$$

Difficulty: learning the joint probability $P(x_1, \cdots, x_n \mid c)$ is infeasible!

- Naïve Bayes classification
  - Assume all input features are class conditionally independent!

$$P(x_1, x_2, \cdots, x_n \mid c) = P(x_1 \mid x_2, \cdots, x_n, c)P(x_2, \cdots, x_n \mid c)$$

Applying the independence assumption

$$= P(x_1 \mid c)P(x_2, \cdots, x_n \mid c)$$

$$= P(x_1 \mid c)P(x_2 \mid c) \cdots P(x_n \mid c)$$

  - Apply the MAP classification rule: assign $\mathbf{x}' = (a_1, a_2, \cdots, a_n)$ to c^* if

$$[P(a_1 \mid c^*) \cdots P(a_n \mid c^*)]P(c^*) > [P(a_1 \mid c) \cdots P(a_n \mid c)]P(c), \quad c \neq c^*, c = c_1, \cdots, c_L$$

estimate of $P(a_1, \cdots, a_n \mid c^*)$

esitmate of $P(a_1, \cdots, a_n \mid c)$

Naïve Bayes

- Algorithm: Discrete-Valued Features
  - Learning Phase: Given a training set S of F features and L classes,

For each target value of c_i $(c_i = c_1, \cdots, c_L)$

$\hat{P}(c_i) \leftarrow$ estimate $P(c_i)$ with examples in S;

For every feature value x_{jk} of each feature x_j $(j = 1, \cdots, F; k = 1, \cdots, N_j)$

$\hat{P}(x_j = x_{jk} \mid c_i) \leftarrow$ estimate $P(x_{jk} \mid c_i)$ with examples in S;

Output: $F * L$ conditional probabilistic (generative) models

  - Test Phase: Given an unknown instance $\mathbf{x}' = (a_1', \cdots, a_n')$

"Look up tables" to assign the label c^* to X' if

$$[\hat{P}(a_1' \mid c^*) \cdots \hat{P}(a_n' \mid c^*)]\hat{P}(c^*) > [\hat{P}(a_1' \mid c_i) \cdots \hat{P}(a_n' \mid c_i)]\hat{P}(c_i), \quad c_i \neq c^*, c_i = c_1, \cdots, c_L$$

Example

- Example: Play Tennis

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
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	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	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

Example

- Learning Phase

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=*Yes*	Play=*No*
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

$P(\text{Play}=Yes) = 9/14$ $P(\text{Play}=No) = 5/14$

Example

- Test Phase
  - Given a new instance, predict its label

    $\mathbf{x}'$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)
  - Look up tables achieved in the learning phrase

P(Outlook=*Sunny*|Play=*Yes*) = 2/9

P(Temperature=*Cool*|Play=*Yes*) = 3/9

P(Huminity=*High*|Play=*Yes*) = 3/9

P(Wind=*Strong*|Play=*Yes*) = 3/9

P(Play=*Yes*) = 9/14

P(Outlook=S*unny*|Play=*No*) = 3/5

P(Temperature=*Cool*|Play==*No*) = 1/5

P(Huminity=*High*|Play=*No*) = 4/5

P(Wind=*Strong*|Play=*No*) = 3/5

P(Play=*No*) = 5/14

  - Decision making with the MAP rule

$P(Yes|\mathbf{x}') \approx [P(Sunny|Yes)P(Cool|Yes)P(High|Yes)P(Strong|Yes)]P(Play=Yes) = 0.0053$

$P(No|\mathbf{x}') \approx [P(Sunny|No)\ P(Cool|No)P(High|No)P(Strong|No)]P(Play=No) = 0.0206$

Given the fact $P(Yes|\mathbf{x}') < P(No|\mathbf{x}')$, we label $\mathbf{x}'$ to be "*No*".

Naïve Bayes

- Algorithm: Continuous-valued Features
  - Numberless values taken by a continuous-valued feature
  - Conditional probability often modeled with the normal distribution

$$\hat{P}(x_j \mid c_i) = \frac{1}{\sqrt{2\pi}\sigma_{ji}} \exp\left(-\frac{(x_j - \mu_{ji})^2}{2\sigma_{ji}^2}\right)$$

μ_{ji} : mean (avearage) of feature values x_j of examples for which $c = c_i$

σ_{ji} : standard deviation of feature values x_j of examples for which $c = c_i$

  - Learning Phase: for $\mathbf{X} = (X_1, \cdots, X_n)$, $C = c_1, \cdots, c_L$

    Output: $n \times L$ normal distributions and $P(C = c_i)$ $i = 1, \cdots, L$
  - Test Phase: Given an unknown instance $\mathbf{X}' = (a_1', \cdots, a_n')$
    - Instead of looking-up tables, calculate conditional probabilities with all the normal distributions achieved in the learning phrase
    - Apply the MAP rule to assign a label (the same as done for the discrete case)

Naïve Bayes

- Example: Continuous-valued Features
  - Temperature is naturally of continuous value.

    Yes: 25.2, 19.3, 18.5, 21.7, 20.1, 24.3, 22.8, 23.1, 19.8

    No: 27.3, 30.1, 17.4, 29.5, 15.1
  - Estimate mean and variance for each class

$$\mu = \frac{1}{N}\sum_{n=1}^{N} x_n, \quad \sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2$$

$$\mu_{Yes} = 21.64, \ \sigma_{Yes} = 2.35$$
$$\mu_{No} = 23.88, \ \sigma_{No} = 7.09$$

  - **Learning Phase**: output two Gaussian models for P(temp|C)

$$\hat{P}(x \mid Yes) = \frac{1}{2.35\sqrt{2\pi}} \exp\left(-\frac{(x-21.64)^2}{2\times 2.35^2}\right) = \frac{1}{2.35\sqrt{2\pi}} \exp\left(-\frac{(x-21.64)^2}{11.09}\right)$$

$$\hat{P}(x \mid No) = \frac{1}{7.09\sqrt{2\pi}} \exp\left(-\frac{(x-23.88)^2}{2\times 7.09^2}\right) = \frac{1}{7.09\sqrt{2\pi}} \exp\left(-\frac{(x-23.88)^2}{50.25}\right)$$

Zero conditional probability

- If no example contains the feature value
  - In this circumstance, we face a zero conditional probability problem during test

$$\hat{P}(x_1 \mid c_i) \cdots \hat{P}(a_{jk} \mid c_i) \cdots \hat{P}(x_n \mid c_i) = 0 \quad \text{for } x_j = a_{jk},\ \hat{P}(a_{jk} \mid c_i) = 0$$

  - For a remedy, class conditional probabilities re-estimated with

$$\hat{P}(a_{jk} \mid c_i) = \frac{n_c + mp}{n + m}$$ **(m-estimate)**

n_c : number of training examples for which $x_j = a_{jk}$ and $c = c_i$

n : number of training examples for which $c = c_i$

p : prior estimate (usually, $p = 1/t$ for t possible values of x_j)

m : weight to prior (number of "virtual" examples, $m \geq 1$)

Zero conditional probability

- Example: $P(\text{outlook=overcast}|\text{no})=0$ in the play-tennis dataset
  - Adding m "virtual" examples (m: up to 1% of #training example)
    - In this dataset, # of training examples for the "no" class is 5.
    - We can only add $m=1$ "virtual" example in our m-esitmate remedy.
  - The "outlook" feature can takes only 3 values. So $p=1/3$.
  - Re-estimate $P(\text{outlook}|\text{no})$ with the m-estimate

$$P(\text{overcast}|\text{no}) = \frac{0+1*\left(\frac{1}{3}\right)}{5+1} = \frac{1}{18}$$

$$P(\text{sunny}|\text{no}) = \frac{3+1*\left(\frac{1}{3}\right)}{5+1} = \frac{5}{9} \qquad P(\text{rain}|\text{no}) = \frac{2+1*\left(\frac{1}{3}\right)}{5+1} = \frac{7}{18}$$

Numerical Stability

- It is often the case that machine learning algorithms need to work with very small numbers
  - Imagine computing the probability of 2000 independent coin flips
  - MATLAB thinks that $(.5)^{2000}=0$

Underflow Prevention

- Multiplying lots of probabilities

➔ floating-point underflow.

- Recall: $log(xy) = log(x) + log(y)$,

➔ better to sum logs of probabilities rather than multiplying probabilities.

Underflow Prevention

- Class with highest final un-normalized log probability score is still the most probable.

$$c_{NB} = \underset{c_j \in C}{\operatorname{argmax}} \log P(c_j) + \sum_{i \in positions} \log P(x_i \mid c_j)$$

Numerical Stability

- Instead of comparing P(Y=5|X_1,…,X_n) with P(Y=6|X_1,…,X_n),
  - Compare their logarithms

$$\begin{aligned}
\log\left(P(Y|X_1,\ldots,X_n)\right) &= \log\left(\frac{P(X_1,\ldots,X_n|Y)\cdot P(Y)}{P(X_1,\ldots,X_n)}\right) \\
&= \text{constant} + \log\left(\prod_{i=1}^{n} P(X_i|Y)\right) + \log P(Y) \\
&= \text{constant} + \sum_{i=1}^{n} \log P(X_i|Y) + \log P(Y)
\end{aligned}$$

Recovering the Probabilities

- What if we want the probabilities though??
- Suppose that for some constant K, we have:

$$\log P(Y = 5|X_1, \ldots, X_n) + K$$

  - And

$$\log P(Y = 6|X_1, \ldots, X_n) + K$$

- How would we recover the original probabilities?

Recovering the Probabilities

- Given:

$$\alpha_i = \log a_i + K$$

- Then for any constant C:

$$\begin{aligned} \frac{a_i}{\sum_i a_i} &= \frac{e^{\alpha_i}}{\sum_i e^{\alpha_i}} \\ &= \frac{e^C \cdot e^{\alpha_i}}{\sum_i e^C \cdot e^{\alpha_i}} \\ &= \frac{e^{\alpha_i + C}}{\sum_i e^{\alpha_i + C}} \end{aligned}$$

$$C = -\max_i \{\alpha_i\}$$

- One suggestion: set C such that the greatest α_i is shifted to zero:

Recap

- We defined a *Bayes classifier* but saw that it's intractable to compute $P(X_1,\ldots,X_n|Y)$
- We then used the *Naïve Bayes assumption* – that everything is independent given the class label Y

- A natural question: is there some happy compromise where we only assume that *some* features are conditionally independent?
  - Stay Tuned…

Conclusions

- Naïve Bayes is:
  - Really easy to implement and often works well
  - Often a good first thing to try
  - Commonly used as a “punching bag” for smarter algorithms

Evaluating classification algorithms

- You have designed a new classifier.

- You give it to me, and I try it on my image dataset

Evaluating classification algorithms

- I tell you that it achieved 95% accuracy on my data.

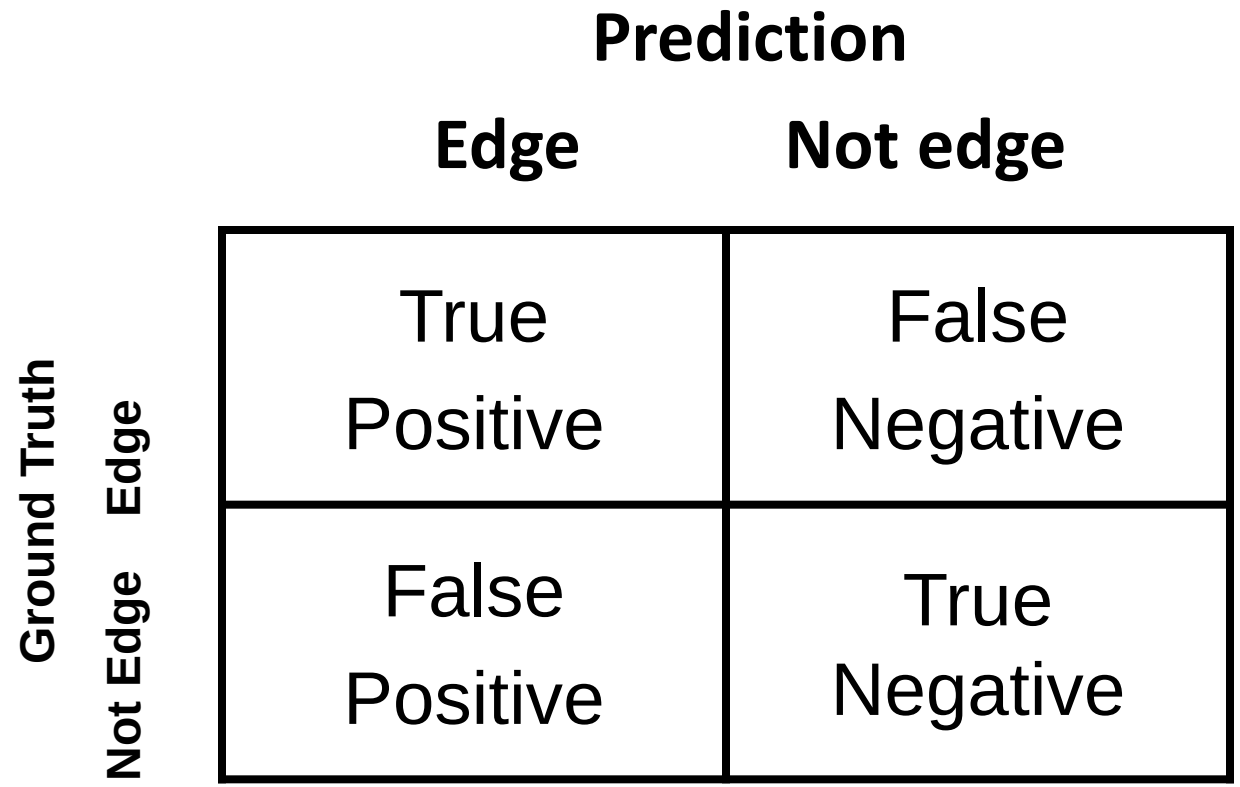
- Is your technique a success?

Types of errors

- But suppose that
  - The 95% is the correctly classified pixels
  - Only 5% of the pixels are actually edges
  - It misses all the edge pixels

- How do we count the effect of different types of error?

Types of errors



Two parts to each: whether you got it correct or not, and what you guessed. For example for a particular pixel, our guess might be labelled…

True Positive

Did we get it correct?
True, we did get it correct.

What did we say?
We said ‘positive’, i.e. edge.

or maybe it was labelled as one of the others, maybe…

False Negative

Did we get it correct?
False, we did not get it correct.

What did we say?
We said ‘negative, i.e. not edge.

Sensitivity and Specificity

Count up the total number of each label (TP, FP, TN, FN) over a large dataset. In ROC analysis, we use two statistics:

$$\textbf{Sensitivity} = \frac{TP}{TP+FN}$$

Can be thought of as the likelihood of spotting a positive case when presented with one.

Or… the proportion of edges we find.

$$\textbf{Specificity} = \frac{TN}{TN+FP}$$

Can be thought of as the likelihood of spotting a negative case when presented with one.

Or… the proportion of non-edges that we find

$$\text{Sensitivity} = \frac{TP}{TP+FN} = ? \qquad \text{Specificity} = \frac{TN}{TN+FP} = ?$$

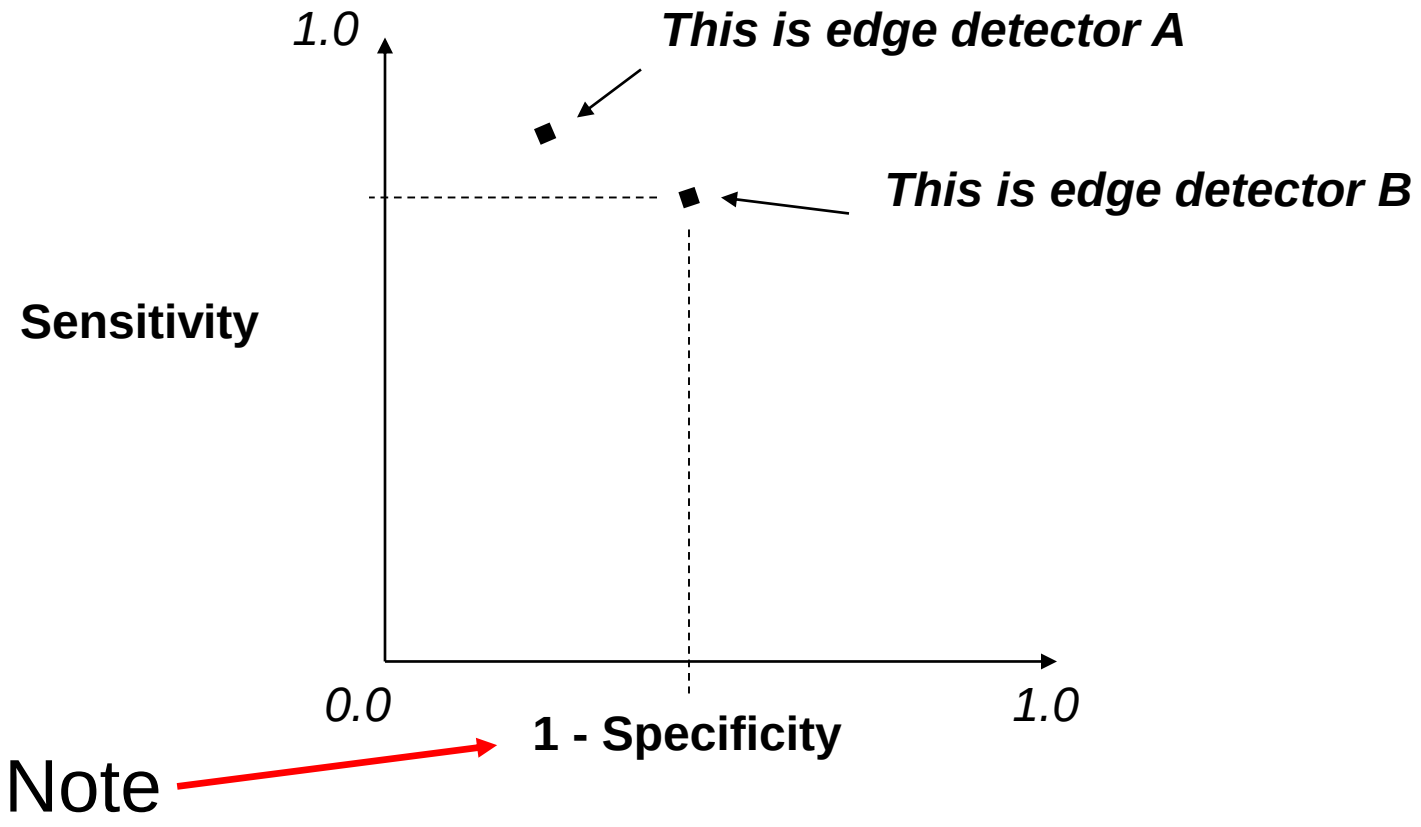
		Prediction 1	**Prediction** 0
Ground Truth	**1**	60	30
	0	80	20

60+30 = 90 cases in the dataset were class 1 (edge)

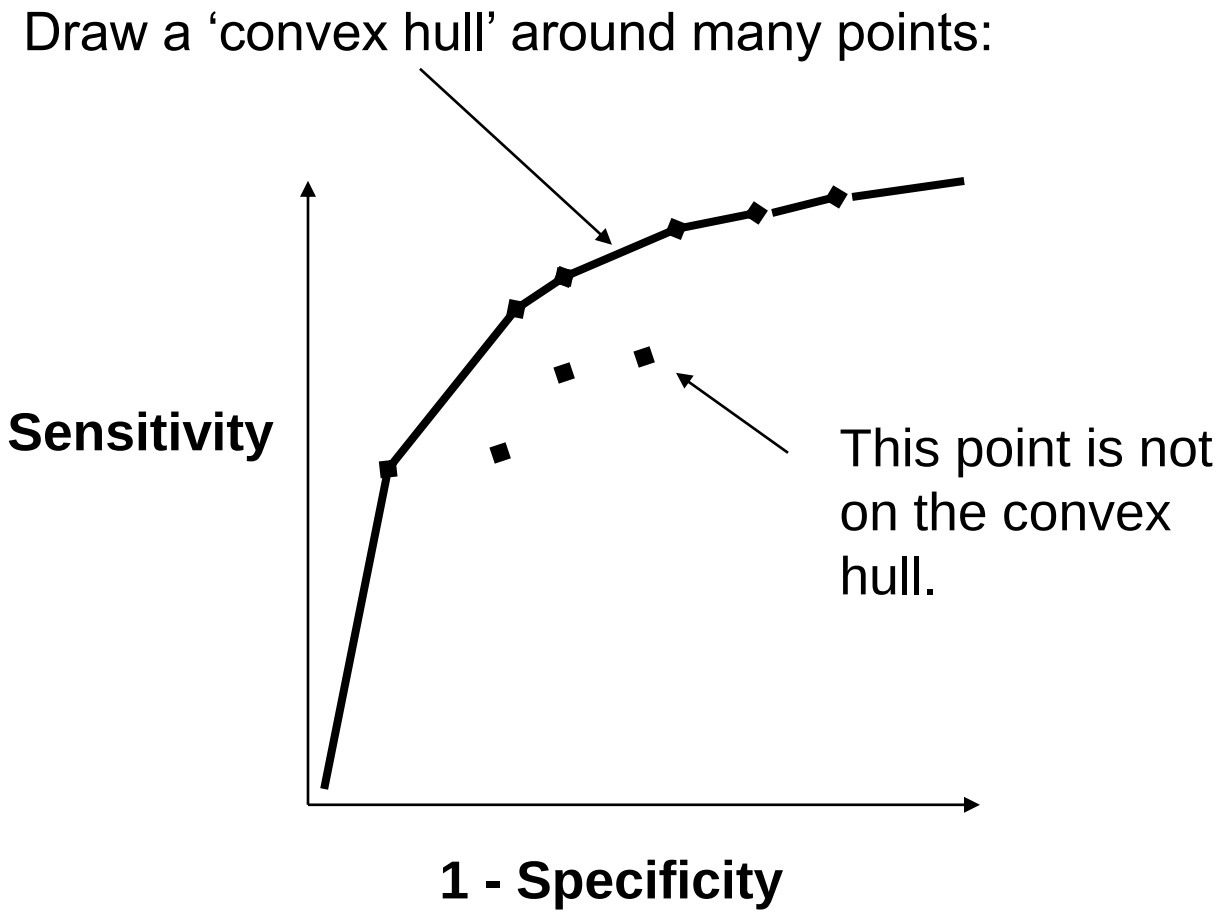
80+20 = 100 cases in the dataset were class 0 (non-edge)

90+100 = 190 examples (pixels) in the data overall

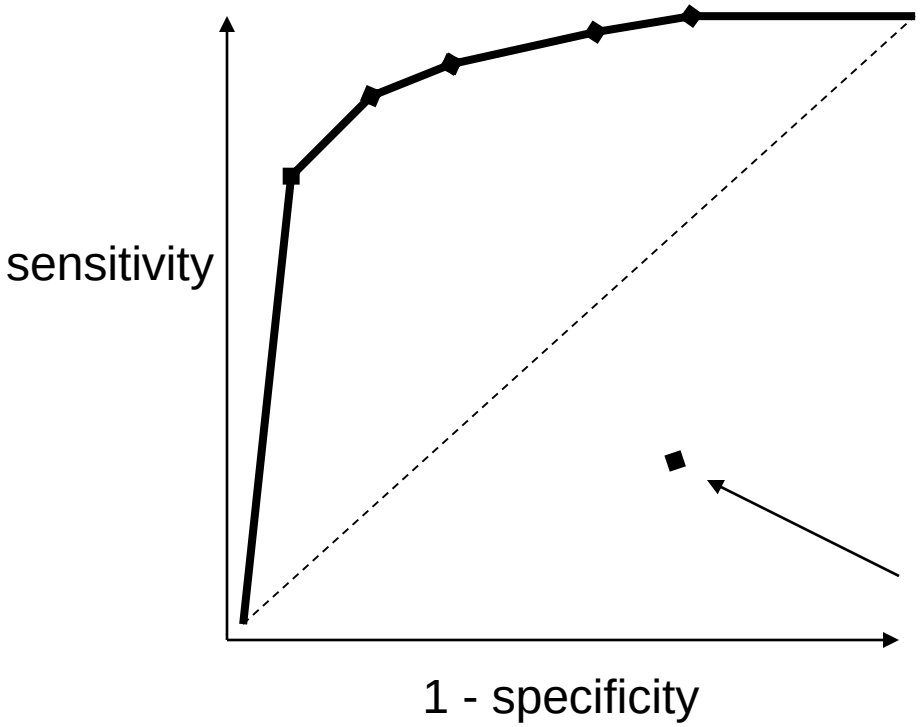
The ROC space



The ROC Curve



ROC Analysis



All the optimal detectors lie on the convex hull.

Which of these is best depends on the ratio of edges to non-edges, and the different cost of misclassification

Any detector on this side can lead to a better detector by flipping its output.

Take-home point : You should always quote sensitivity and specificity for your algorithm, if possible plotting an ROC graph. Remember also though, any statistic you quote should be an average over a suitable range of tests for your algorithm.

Holdout estimation

- What to do if the amount of data is limited?
- The *holdout* method reserves a certain amount for testing and uses the remainder for training

➔Usually: one third for testing, the rest for training

Holdout estimation

- Problem: the samples might not be representative
  - Example: class might be missing in the test data

- Advanced version uses *stratification*
  - Ensures that each class is represented with approximately equal proportions in both subsets

Repeated holdout method

- Repeat process with different subsamples

➔ more reliable

  - In each iteration, a certain proportion is randomly selected for training (possibly with stratificiation)
  - The error rates on the different iterations are averaged to yield an overall error rate

Repeated holdout method

- Still not optimum: the different test sets overlap
  - Can we prevent overlapping?
  - Of course!

Cross-validation

- *Cross-validation* avoids overlapping test sets
  - First step: split data into *k* subsets of equal size
  - Second step: use each subset in turn for testing, the remainder for training

- Called *k-fold cross-validation*

Cross-validation

- Often the subsets are stratified before the cross-validation is performed

- The error estimates are averaged to yield an overall error estimate

More on cross-validation

- Standard method for evaluation: stratified ten-fold cross-validation
- Why ten?
  - Empirical evidence supports this as a good choice to get an accurate estimate
  - There is also some theoretical evidence for this
- Stratification reduces the estimate’s variance
- Even better: repeated stratified cross-validation
  - E.g. ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)

Leave-One-Out cross-validation

- Leave-One-Out:
  a particular form of cross-validation:
  - Set number of folds to number of training instances
  - I.e., for *n* training instances, build classifier *n* times
- Makes best use of the data
- Involves no random subsampling
- Very computationally expensive
  - (exception: NN)

Leave-One-Out-CV and stratification

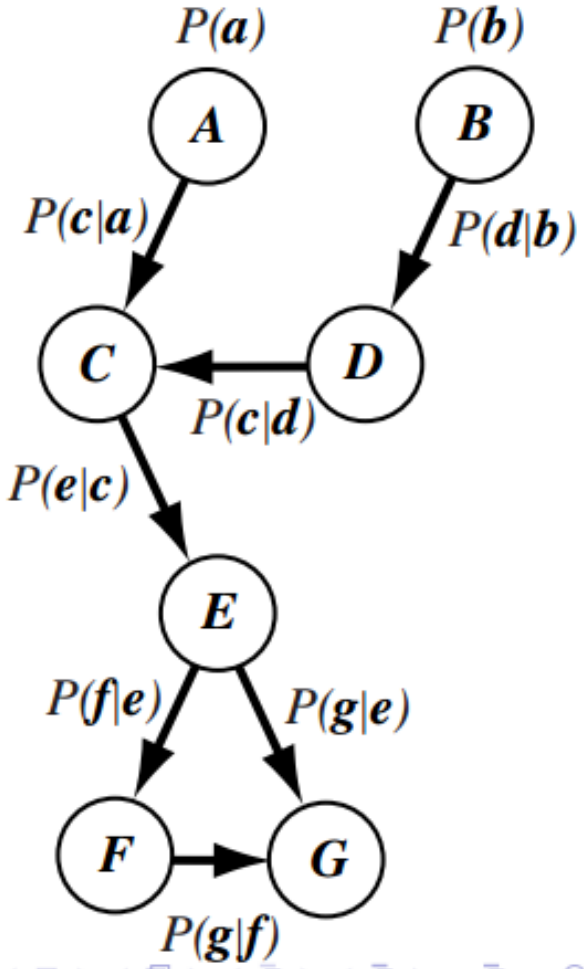
- Disadvantage of Leave-One-Out-CV: stratification is not possible
  - It *guarantees* a non-stratified sample because there is only one instance in the test set!

Bayesian Networks

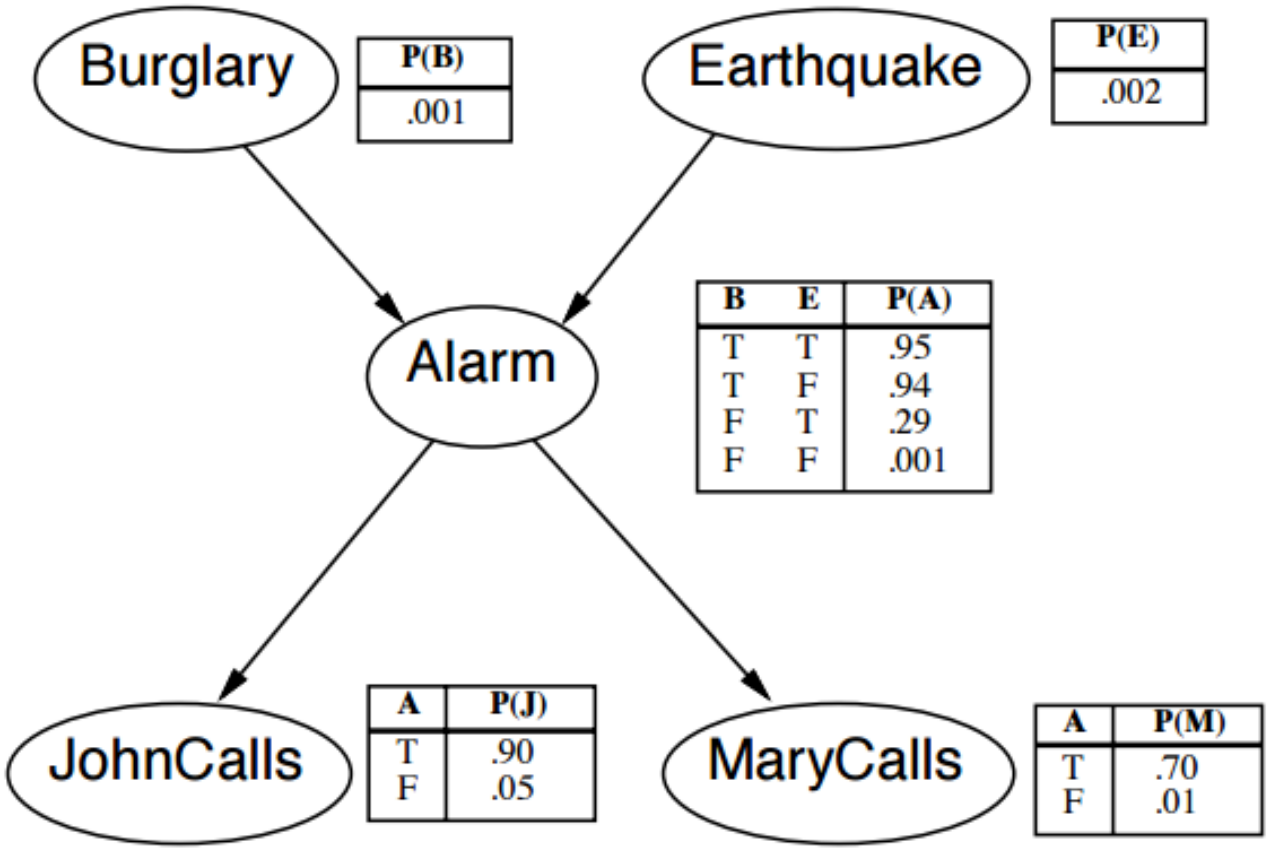
Slides are adapted from Jason Corso

Bayes Nets

- Each **node** represents one variable (assume discrete for simplicity).
- A **link** joining two nodes is directional and it represents **conditional probabilities**.
- The intuitive meaning of a link is that the source has a direct influence on the sink.
- Since we typically work with discrete distributions, we evaluate the conditional probability at each node given its parents and store it in a lookup table called a **conditional probability table**.



Example



- Key: given knowledge of the values of some nodes in the network, we can apply Bayesian inference to determine the maximum posterior values of the unknown variables!

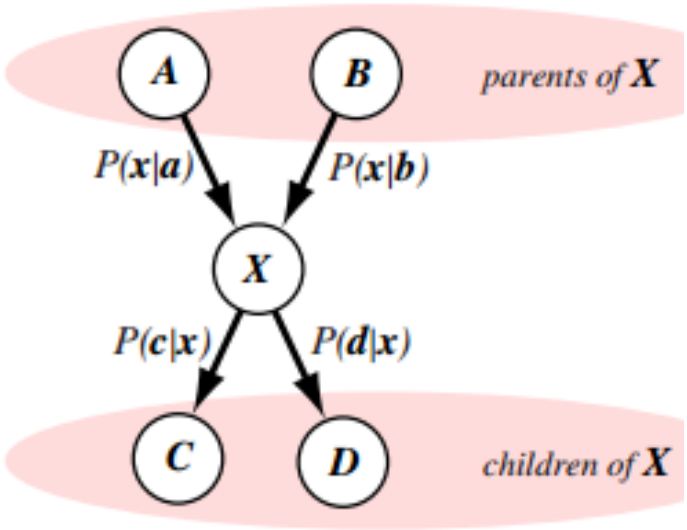
Full Joint distribution

- Consider a Bayes network with n variables $x_1, \ldots, x_n$.
- Denote the parents of a node x_i as $\mathcal{P}(x_i)$.
- Then, we can decompose the joint distribution into the product of conditionals

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P\left(x_i | \mathcal{P}(x_i)\right) \qquad (1)$$

$$\begin{aligned}
\mathbf{P}\,(x_1, \ldots, x_n) &= P(x_n | x_{n-1}, \ldots x_1) P(x_{n-1}, \ldots x_1) \\
&= P(x_n | x_{n-1}, \ldots x_1) P(x_{n-1} | x_{n-2} \ldots x_1) \ldots P(x_2 | x_1) P(x_1) \\
&= \prod_{i=1} \mathbf{P}\,(x_i | x_{i-1}, \ldots x_1) \\
&= \prod_{i=1} \mathbf{P}\,(x_i | \mathit{Parents}(X_i))
\end{aligned}$$

- What is the distribution at a single node, given the rest of the network and the evidence $\mathbf{e}$?
- **Parents** of $\mathbf{X}$, the set $\mathcal{P}$ are the nodes on which $\mathbf{X}$ is conditioned.
- **Children** of $\mathbf{X}$, the set $\mathcal{C}$ are the nodes conditioned on $\mathbf{X}$.

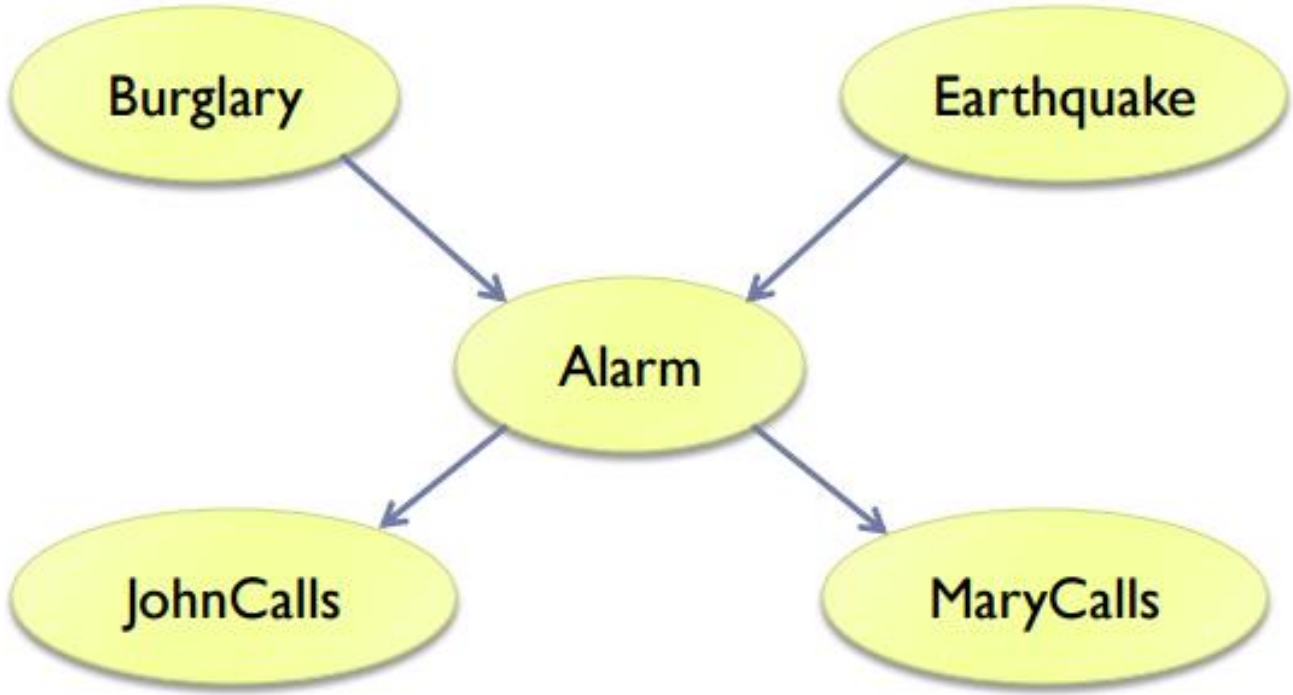


- Use the Bayes Rule, for the case on the right:

$$\begin{aligned} P(a,b,x,c,d) &= P(a,b,x|c,d)P(c,d) \\ &= P(a,b|x)P(x|c,d)P(c,d) \end{aligned}$$

or more generally,

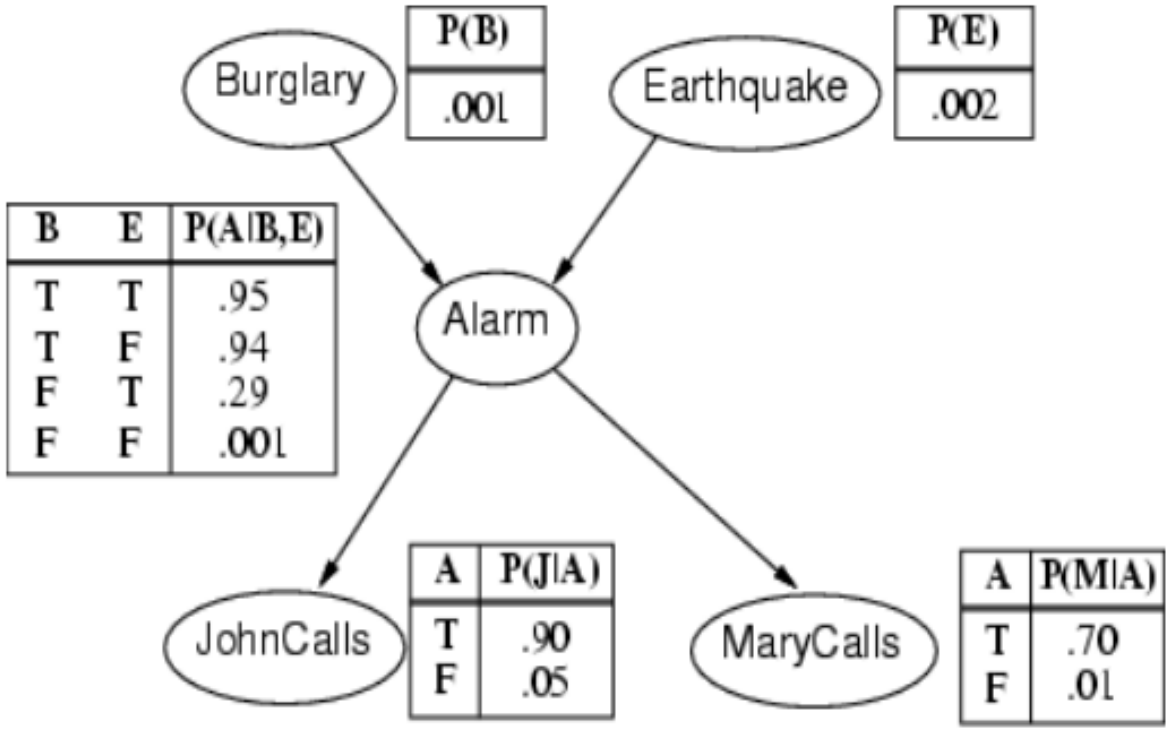
$$P(\mathcal{C}(x),x,\mathcal{P}(x)|\mathbf{e}) = P(\mathcal{C}(x)|x,\mathbf{e})P(x|\mathcal{P}(x),\mathbf{e})P(\mathcal{P}(x)|,\mathbf{e})$$



$$\boldsymbol{P}(x_1, \ldots, x_n) = \prod_{i=1} \boldsymbol{P}(x_i \mid Parents(X_i))$$

P(JohnCalls ^ MaryCalls ^ Alarm ^ Burglary ^ Earthquake)
= P(JohnCalls|Alarm) x P(MaryCalls|Alarm) x P(Alarm|Burglary^Earthquake)
x P(Burglary) x P(Earthquake)

Problem 1



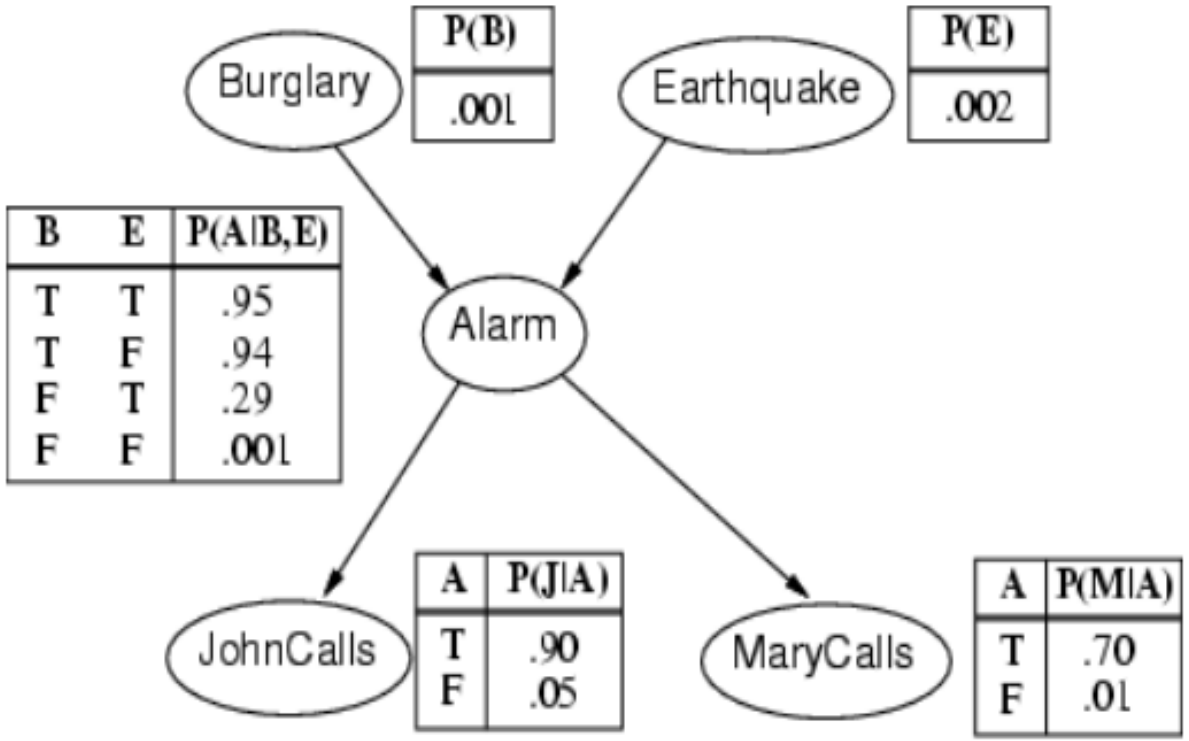
J: *JohnCalls*
M: *MaryCalls*
A: *Alarm*
B: *Burglary*
E: *Earthquake*

What is the probability of the event that the alarm has sounded and no burglary but an earthquake has occurred and both Mary and John call?

$P(J \wedge M \wedge A \wedge \neg B \wedge E) = P(J|A) \times P(M|A) \times P(A|\neg B \wedge E) \times P(\neg B) \times P(E)$

$= 0.90 \times 0.70 \times 0.29 \times 0.999 \times 0.002 = 0.00036$

Problem 2

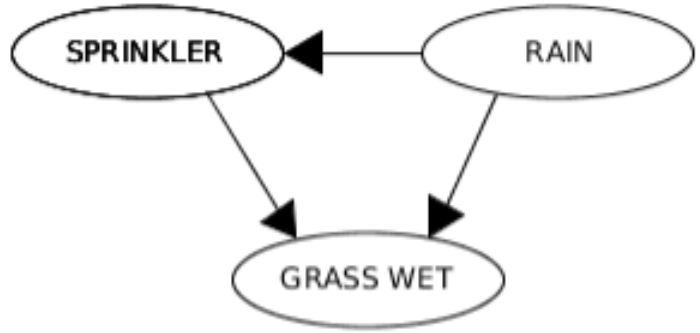


J: *JohnCalls*
M: *MaryCalls*
A: *Alarm*
B: *Burglary*
E: *Earthquake*

What is the probability of the event that the alarm has sounded but neither a burglary nor an earthquake has occurred and John call and Mary didn't call?

$$P(J \wedge \neg M \wedge A \wedge \neg B \wedge \neg E) = P(J|A) \times P(\neg M|A) \times P(A|\neg B \wedge \neg E) \times P(\neg B) \times P(\neg E)$$
$$= 0.90 \times 0.30 \times 0.001 \times 0.999 \times 0.998 = 0.00027$$

RAIN	SPRINKLER T	SPRINKLER F
F	0.4	0.6
T	0.01	0.99



RAIN T	RAIN F
0.2	0.8

What is P(S|G)?

SPRINKLER	RAIN	GRASS WET T	GRASS WET F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

P(S|G) = P(S^G)/P(G)
0.6467

P(S^G) = P(S^G^R) + P(S^G^~R) = 0.00198+0.288 = .28998
P(G) = P(S^G) + P(~S^G) = .28998 + 0.1584 = 0.44838

P(~S^G) =P(~S^G^R) + P(~S^G^~R) = 0.1584 + 0
P(S^G^R)=P(S|R)P(G|S^R)P(R) = (0.01)(0.99)(0.2) = 0.00198
P(S^G^~R) = P(S|~R)P(G|S^~R)P(~R) = (0.4)(0.9)(0.8) = 0.288
P(~S^G^R) = P(~S|R)P(G|~S^R)P(R) = (0.99)(0.8)(0.2) = 0.1584
P(~S^G^~R) = P(~S|~R)P(G|~S^~R)P(~R) = (0.6)(0.0)(0.8) = 0