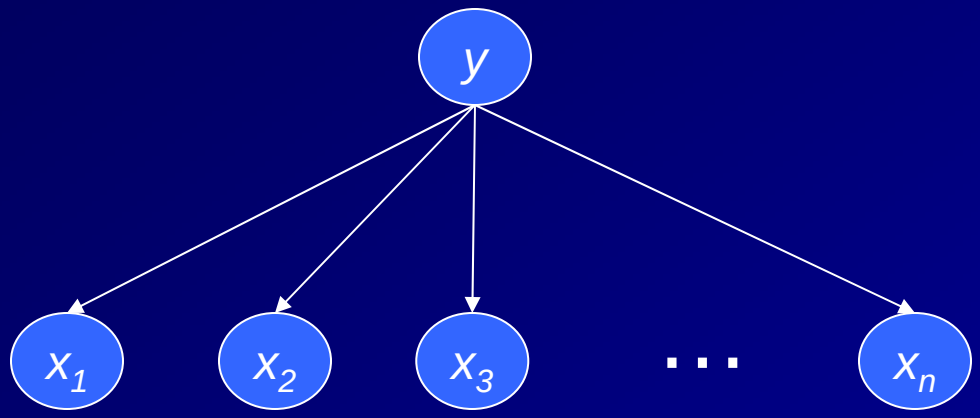
Learning Bayesian Networks: Naïve and non-Naïve Bayes

- Hypothesis Space
  - fixed size
  - stochastic
  - continuous parameters
- Learning Algorithm
  - direct computation
  - eager
  - batch

Multivariate Gaussian Classifier

- The multivariate Gaussian Classifier is equivalent to a simple Bayesian network
- This models the joint distribution P(**x**,y) under the assumption that the class conditional distributions P(**x**|y) are multivariate gaussians
  - P(y): multinomial random variable (K-sided coin)
  - P(**x**|y): multivariate gaussian mean μ_k covariance matrix Σ_k

Naïve Bayes Model



- Each node contains a probability table
  - y: $P(y = k)$
  - x_j: $P(x_j = v \mid y = k)$ "class conditional probability"
- Interpret as a generative model
  - Choose the class k according to $P(y = k)$
  - Generate each feature *independently* according to $P(x_j=v \mid y=k)$
  - The feature values are *conditionally independent*
    $P(x_i,x_j \mid y) = P(x_i \mid y) \cdot P(x_j \mid y)$

Representing P(x_j|y)

- Many representations are possible
  - Univariate Gaussian
    - if x_j is a continuous random variable, then we can use a normal distribution and learn the mean μ and variance σ^2
  - Multinomial
    - if x_j is a discrete random variable, $x_j \in \{v_1, \ldots, v_m\}$, then we construct the conditional probability table

	y = 1	y=2	…	y=K
$x_j=v_1$	P($x_j=v_1$ | y = 1)	P($x_j=v_1$ | y = 2)	…	P($x_j=v_m$ | y = K)
$x_j=v_2$	P($x_j=v_2$ | y = 1)	P($x_j=v_2$ | y = 2)	…	P($x_j=v_m$ | y = K)
…	…	…	…	…
$x_j=v_m$	P($x_j=v_m$ | y = 1)	P($x_j=v_m$ | y = 2)	…	P($x_j=v_m$ | y = K)

  - Discretization
    - convert continuous x_j into a discrete variable
  - Kernel Density Estimates
    - apply a kind of nearest-neighbor algorithm to compute P(x_j | y) in neighborhood of query point

Discretization via Mutual Information

- Many discretization algorithms have been studied. One of the best is mutual information discretization
  - To discretize feature x_j, grow a decision tree considering only splits on x_j. Each leaf of the resulting tree will correspond to a single value of the discretized x_j.
  - Stopping rule (applied at each node). Stop when

$$I(x_j; y) < \frac{\log_2(N-1)}{N} + \frac{\Delta}{N}$$

$$\Delta = \log_2(3^K - 2) - [K \cdot H(S) - K_l \cdot H(S_l) - K_r \cdot H(S_r)]$$

  - where S is the training data in the parent node; S_l and S_r are the examples in the left and right child. K, K_l, and K_r are the corresponding number of classes present in these examples. I is the mutual information, H is the entropy, and N is the number of examples in the node.

Kernel Density Estimators

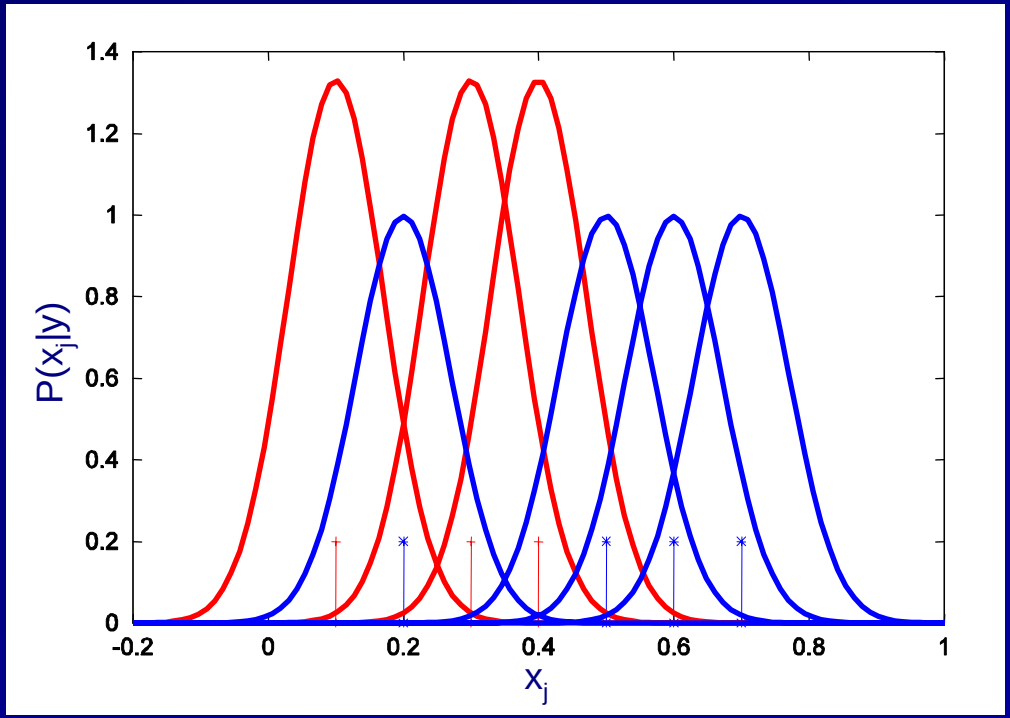
- Define $K(x_j, x_{i,j}) = \frac{1}{\sqrt{2\pi}\sigma} \exp - \left(\frac{x_j - x_{i,j}}{\sigma}\right)^2$ to be the Gaussian Kernel with parameter σ
- Estimate

$$P(x_j|y = k) = \frac{\sum_{\{i|y=k\}} K(x_j, x_{i,j})}{N_k}$$

where N_k is the number of training examples in class *k*.

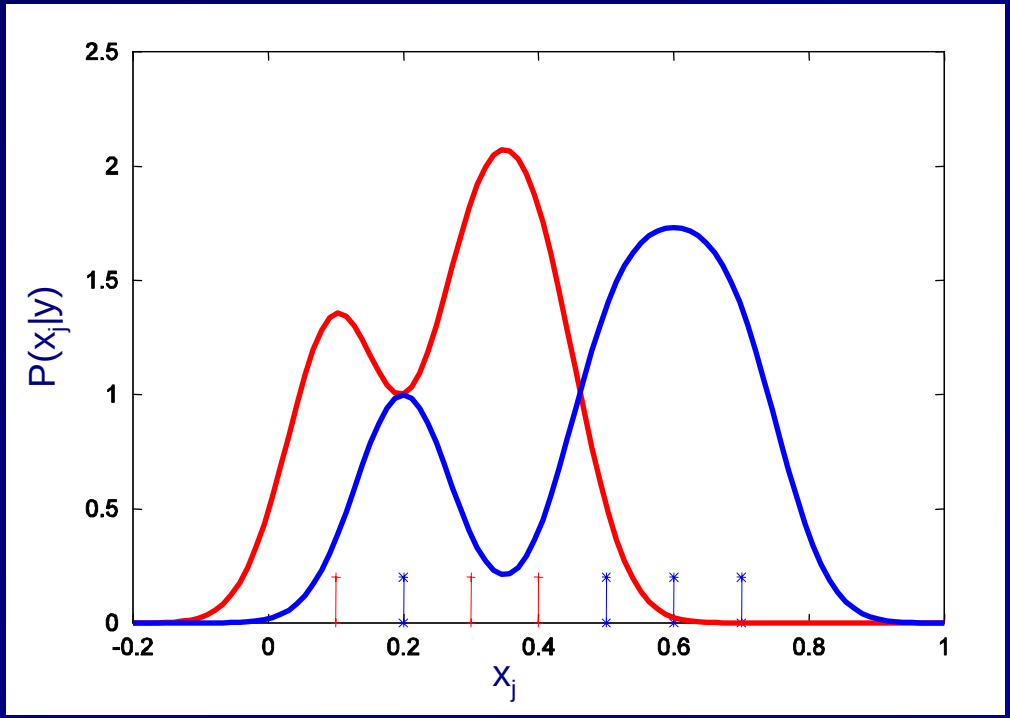
Kernel Density Estimators (2)

- This is equivalent to placing a Gaussian "bump" of height $1/N_k$ on each trianing data point from class *k* and then adding them up

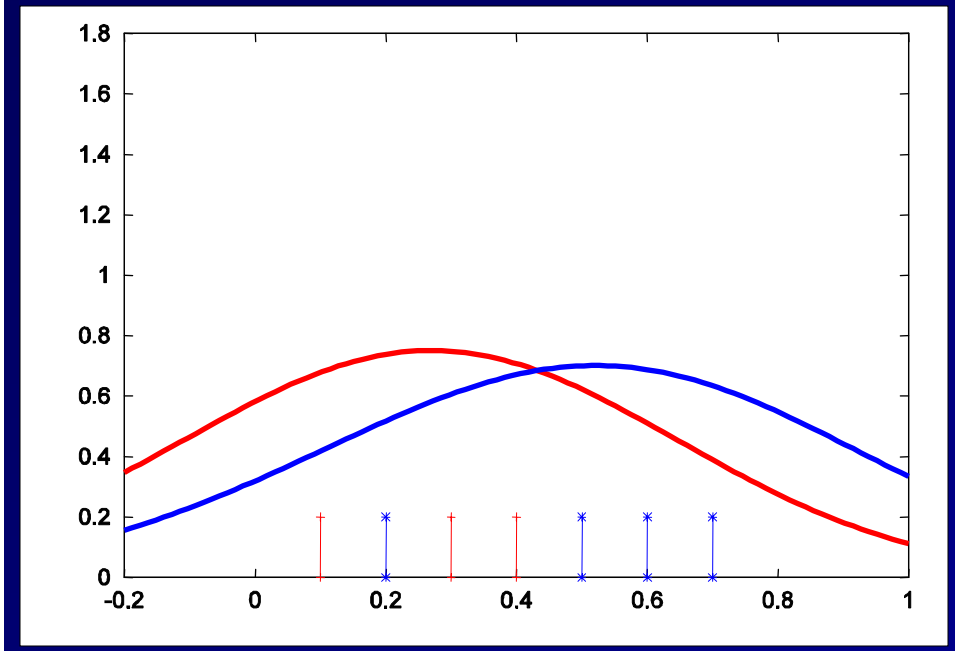


Kernel Density Estimators

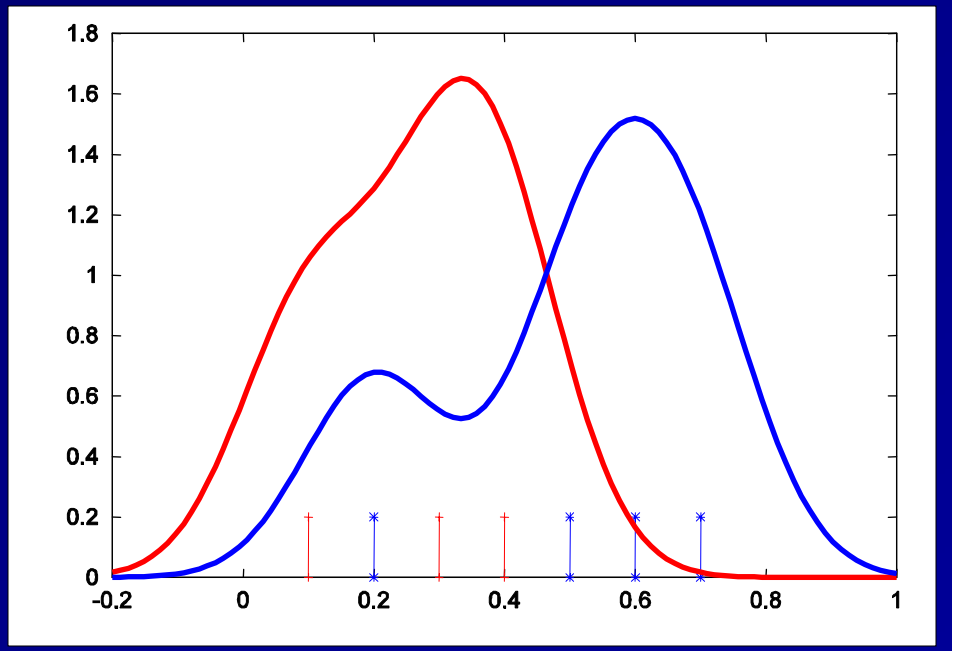
- Resulting probability density



The value chosen for σ is critical



σ=0.15



σ=0.50

Naïve Bayes learns a Linear Threshold Unit

- For multinomial and discretized attributes (but not Gaussian), Naïve Bayes gives a linear decision boundary

$$P(\mathbf{x}|Y=y) = P(x_1=v_1|Y=y) \cdot P(x_2=v_2|Y=y) \cdots P(x_n=v_n|Y=y)$$

- Define a discriminant function for class 1 versus class K

$$h(\mathbf{x}) = \frac{P(Y=1|\mathbf{X})}{P(Y=K|\mathbf{X})} = \frac{P(x_1=v_1|Y=1)}{P(x_1=v_1|Y=K)} \cdots \frac{P(x_n=v_n|Y=1)}{P(x_n=v_n|Y=K)} \cdot \frac{P(Y=1)}{P(Y=K)}$$

Log of Odds Ratio

$$\frac{P(y=1|\mathbf{x})}{P(y=K|\mathbf{x})} = \frac{P(x_1=v_1|y=1)}{P(x_1=v_1|y=K)} \cdots \frac{P(x_n=v_n|y=1)}{P(x_n=v_n|y=K)} \cdot \frac{P(y=1)}{P(y=K)}$$

$$\log \frac{P(y=1|\mathbf{x})}{P(y=K|\mathbf{x})} = \log \frac{P(x_1=v_1|y=1)}{P(x_1=v_1|y=K)} + \ldots \log \frac{P(x_n=v_n|y=1)}{P(x_n=v_n|y=K)} + \log \frac{P(y=1)}{P(y=K)}$$

Suppose each x_j is binary and define

$$\alpha_{j,0} = \log \frac{P(x_j=0|y=1)}{P(x_j=0|y=K)}$$

$$\alpha_{j,1} = \log \frac{P(x_j=1|y=1)}{P(x_j=1|y=K)}$$

Log Odds (2)

- Now rewrite as

$$\log \frac{P(y=1|\mathbf{x})}{P(y=K|\mathbf{x})} = \sum_j (\alpha_{j,1} - \alpha_{j,0}) x_j + \alpha_{j,0} + \log \frac{P(y=1)}{P(y=K)}$$

$$\log \frac{P(y=1|\mathbf{x})}{P(y=K|\mathbf{x})} = \sum_j (\alpha_{j,1} - \alpha_{j,0}) x_j + \left(\sum_j \alpha_{j,0} + \log \frac{P(y=1)}{P(y=K)} \right)$$

- We classify into class 1 if this is ≥ 0 and into class K otherwise

Learning the Probability Distributions by Direct Computation

- P(*y*=*k*) is just the fraction of training examples belonging to class *k*.
- For multinomial variables, P(x_j = *v* | *y* = *k*) is the fraction of training examples in class *k* where x_j = *v*
- For Gaussian variables, $\hat{\mu}_{jk}$ is the average value of x_j for training examples in class *k*. $\hat{\sigma}_{jk}$ is the sample standard deviation of those points:

$$\hat{\sigma}_{jk} = \sqrt{\frac{1}{N_k} \sum_{\{i|y_i=k\}} (x_{i,j} - \hat{\mu}_{jk})^2}$$

Improved Probability Estimates via Laplace Corrections

- When we have very little training data, direct probability computation can give probabilities of 0 or 1. Such extreme probabilities are "too strong" and cause problems
- Suppose we are estimate a probability P(z) and we have n_0 examples where z is false and n_1 examples where z is true. Our direct estimate is

$$P(z=1) = \frac{n_1}{n_0 + n_1}$$

- Laplace Estimate. Add 1 to the numerator and 2 to the denominator

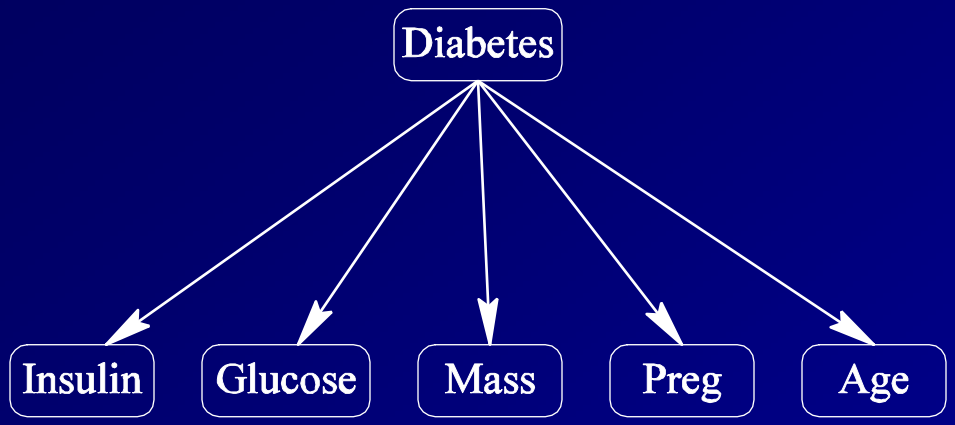
$$P(z=1) = \frac{n_1 + 1}{n_0 + n_1 + 2}$$

This says that in the absence of any evidence, we expect P(z) = 0.5, but our belief is weak (equivalent to 1 example for each outcome).

- Generalized Laplace Estimate. If z has K different outcomes, then we estimate it as

$$P(z=1) = \frac{n_1 + 1}{n_0 + \cdots + n_{K-1} + K}$$

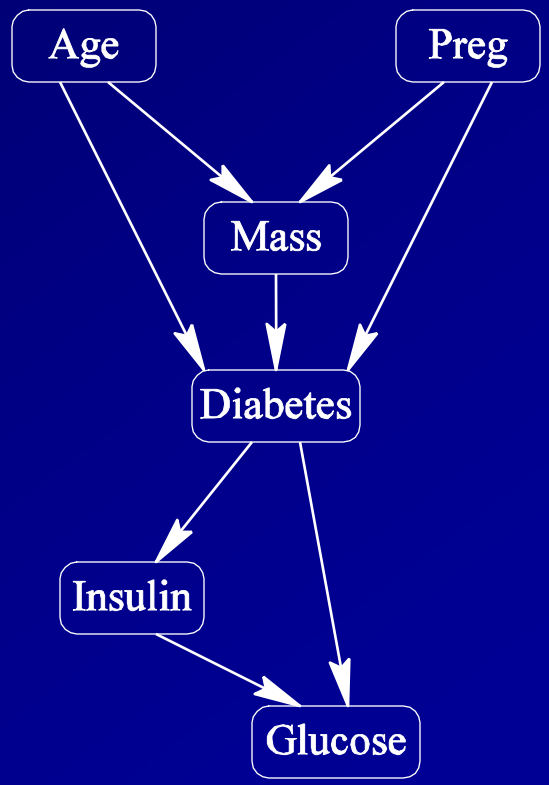
Naïve Bayes Applied to Diabetes Diagnosis



- Bayes nets and causality
  - Bayes nets work best when arrows follow the direction of causality
    - two things with a common cause are likely to be conditionally independent given the cause; arrows in the causal direction capture this independence
  - In a Naïve Bayes network, arrows are often not in the causal direction
    - diabetes does not cause pregnancies
    - diabetes does not cause age
  - But some arrows are correct
    - diabetes does cause the level of blood insulin and blood glucose

Non-Naïve Bayes

- Manually construct a graph in which all arcs are causal
- Learning the probability tables is still easy. For example, P(Mass | Age, Preg) involves counting the number of patients of a given age and number of pregnancies that have a given body mass
- Classification:



$$P(D = d|A, P, M, I, G) =$$

$$\frac{P(I|D = d)P(G|I, D = d)P(D = d|A, M, P)}{P(I, G)}$$

Evaluation of Naïve Bayes

Criterion	LMS	Logistic	LDA	Trees	Nets	NNbr	SVM	NB
Mixed data	no	no	no	yes	no	no	no	yes
Missing values	no	no	yes	yes	no	some	no	yes
Outliers	no	yes	no	yes	yes	yes	yes	disc
Monotone transformations	no	no	no	yes	some	no	no	disc
Scalability	yes	yes	yes	yes	yes	no	no	yes
Irrelevant inputs	no	no	no	some	no	no	some	some
Linear combinations	yes	yes	yes	no	yes	some	yes	yes
Interpretable	yes	yes	yes	yes	no	no	some	yes
Accurate	yes	yes	yes	no	yes	no	yes	yes

- Naïve Bayes is very popular, particularly in natural language processing and information retrieval where there are many features compared to the number of examples
- In applications with lots of data, Naïve Bayes does not usually perform as well as more sophisticated methods

Naïve Bayes Summary

- Advantages of Bayesian networks
  - Produces stochastic classifiers
    - can be combined with utility functions to make optimal decisions
  - Easy to incorporate causal knowledge
    - resulting probabilities are easy to interpret
  - Very simple learning algorithms
    - if all variables are observed in training data
- Disadvantages of Bayesian networks
  - Fixed sized hypothesis space
    - may underfit or overfit the data
    - may not contain any good classifiers if prior knowledge is wrong
  - Harder to handle continuous features