# CSC 411: Naive Bayes

Raquel Urtasun & Rich Zemel

University of Toronto

Oct 6, 2014

# Today

- Classification – Multi-dimensional Bayes classifier
- Estimate probability densities from data
- Naive Bayes

# Generative vs Discriminative

Two approaches to classification:

- **Generative approach**: model the distribution of inputs characteristic of the class (Bayes classifier)
  - Build a model of $p(\mathbf{x}|t_k)$
  - Apply Bayes Rule
- **Discriminative** classifiers estimate parameters of decision boundary/class separator directly from labeled sample
  - learn boundary parameters directly (logistic regression), or
  - learn mappings from inputs to classes (least-squares, neural nets)

# Bayes Classifier

- Aim to diagnose whether patient has diabetes: classify into one of two classes (yes C=1; no C=0)
- Run battery of tests
- Given patient's results: $\mathbf{x} = [x_1, x_2, \cdots, x_d]^T$ we want to update class probabilities using Bayes Rule:

$$p(C|\mathbf{x}) = \frac{p(\mathbf{x}|C)p(C)}{p(\mathbf{x})}$$

- More formally

$$\text{posterior} = \frac{\text{Class likelihood} \times \text{prior}}{\text{Evidence}}$$

- How can we compute $p(\mathbf{x})$ for the two class case?

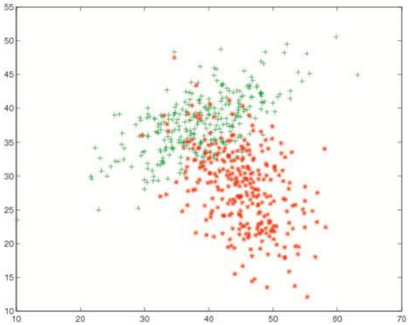
$$p(\mathbf{x}) = p(\mathbf{x}|C=0)p(C=0) + p(\mathbf{x}|C=1)p(C=1)$$

# Classification: Diabetes Example

- Last class we had a single input/observation per patient: white blood cell count

$$p(C = 1|x = 50) = \frac{p(x = 50|C = 1)p(C = 1)}{p(x = 50)}$$

- Add second observation: Plasma glucose value
- Can construct bivariate normal (Gaussian) distribution of each class



# Gaussian Bayes Classifier

- Gaussian (or normal) distribution:

$$p(\mathbf{x}|t_k) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-(\mathbf{x} - \mu_k)^T \Sigma^{-1} (\mathbf{x} - \mu_k)\right]$$

- Each class $k$ has associated mean vector, but typically the classes share a single covariance matrix

# Multivariate Data

- Multiple measurements (sensors)
- $d$ inputs/features/attributes
- $N$ instances/observations/examples

$$\mathbf{X} = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_d^1 \\ x_1^2 & x_2^2 & \cdots & x_d^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^N & x_2^N & \cdots & x_d^N \end{bmatrix}$$

# Multivariate Parameters

- Mean

$$\mathbb{E}[\mathbf{x}] = [\mu_1, \cdots, \mu_d]^T$$

- Covariance

$$\Sigma = Cov(\mathbf{x}) = \mathbb{E}[(\mathbf{x} - \mu)^T(\mathbf{x} - \mu)] = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1d} \\ \sigma_{12} & \sigma_2^2 & \cdots & \sigma_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{d1} & \sigma_{d2} & \cdots & \sigma_d^2 \end{bmatrix}$$

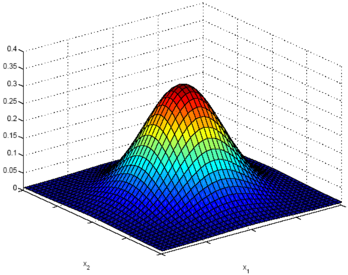
- Correlation $= Corr(\mathbf{x})$ is the covariance divided by the product of standard deviation

$$\rho_{ij} = \frac{\sigma_{ij}}{\sigma_i \sigma_j}$$

# Multivariate Gaussian Distribution

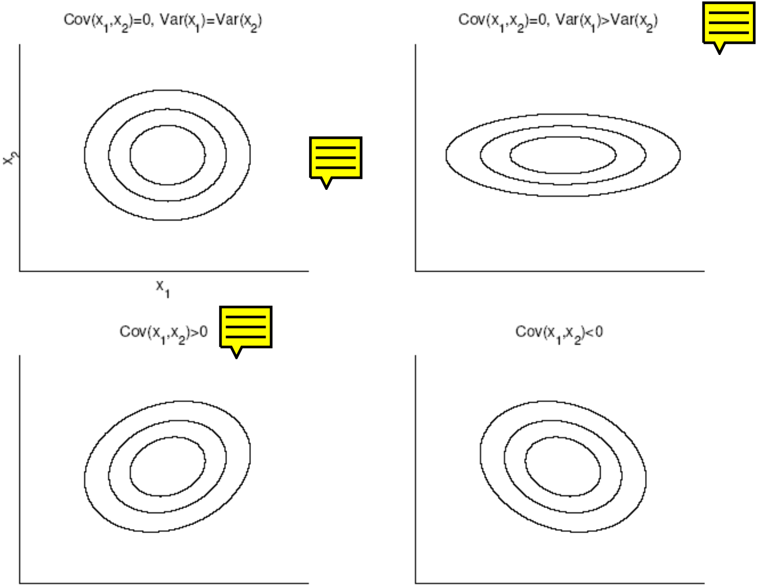
- $\mathbf{x} \sim \mathcal{N}(\mu, \Sigma)$, a Gaussian (or normal) distribution defined as

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-(\mathbf{x}-\mu_k)^T\Sigma^{-1}(\mathbf{x}-\mu_k)\right]$$

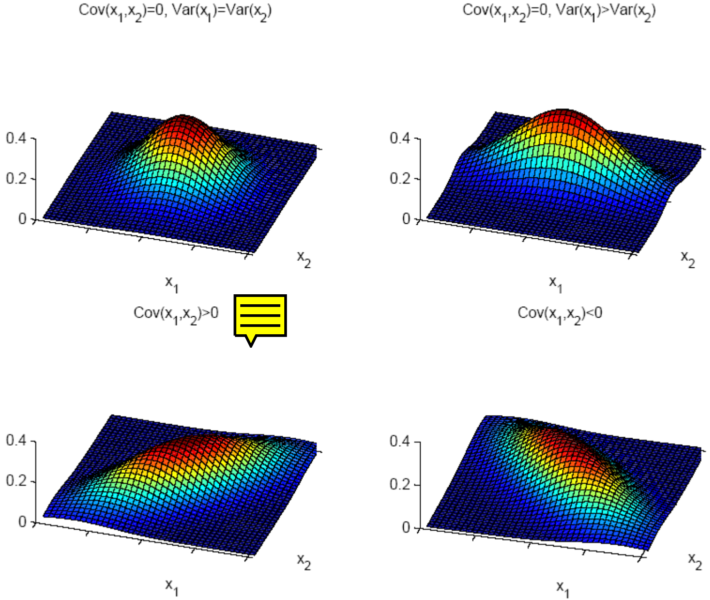



- Mahalanobis distance $(\mathbf{x}-\mu_k)^T\Sigma^{-1}(\mathbf{x}-\mu_k)$ measures the distance from $\mathbf{x}$ to $\mu$ in terms of $\Sigma$
- It normalizes for difference in variances and correlations

# Bivariate Normal



# Bivariate Normal



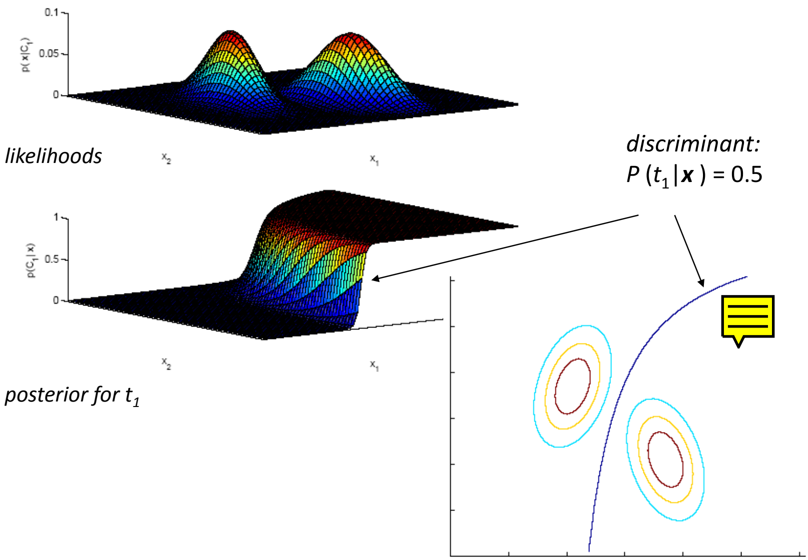
# Gaussian Bayes Classifier Decision Boundary

- GBC decision boundary: based on class posterior
- Take the class which has higher posterior probability

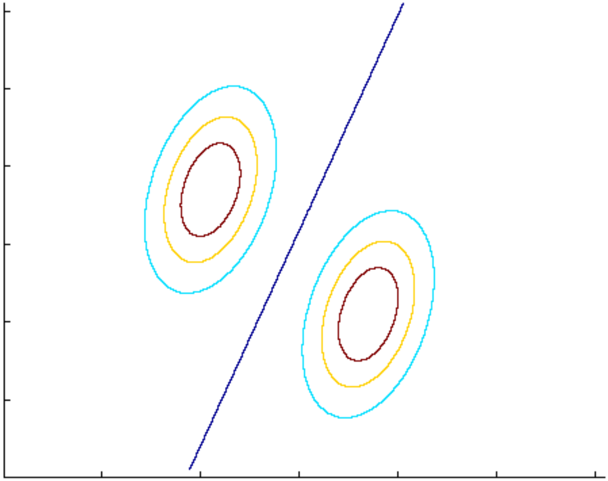
$$
\begin{aligned}
\log p(t_k|\mathbf{x}) &= \log p(\mathbf{x}|t_k) + \log p(t_k) - \log p(\mathbf{x}) \\
&= -\frac{d}{2}\log(2\pi) - \frac{1}{2}\log|\Sigma_k^{-1}| - \frac{1}{2}(\mathbf{x}-\mu_k)^T\sigma_k^{-1}(\mathbf{x}-\mu_k) + \\
&\quad + \log p(t_k) - \log p(\mathbf{x})
\end{aligned}
$$

- Decision: which class has higher posterior probability

# Decision Boundary



# Shared Covariance Matrix



# Learning Gaussian Bayes Classifier

- Learn the parameters using maximum likelihood

$$
\begin{aligned}
\ell(\phi, \mu_0, \mu_1, \Sigma) &= -\log \prod_{n=1}^{N} p(\mathbf{x}^n, t^n | \phi, \mu_0, \mu_1, \Sigma) \\
&= -\log \prod_{n=1}^{N} p(\mathbf{x}^n | t^n, \mu_0, \mu_1, \Sigma) p(t^n | \phi)
\end{aligned}
$$

- What have I assumed?

# More on MLE

- Assume the prior is Bernoulli (we have two classes)

$$p(t|\phi) = \phi^t (1-\phi)^{1-t}$$

- You can compute the ML estimate in close form

$$\begin{aligned}
\phi &= \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}[t^n = 1] \\
\mu_0 &= \frac{\sum_{n=1}^{N} \mathbb{1}[t^n = 0] \cdot \mathbf{x}^n}{\sum_{n=1}^{N} \mathbb{1}[t^n = 0]} \\
\mu_1 &= \frac{\sum_{n=1}^{N} \mathbb{1}[t^n = 1] \cdot \mathbf{x}^n}{\sum_{n=1}^{N} \mathbb{1}[t^n = 1]} \\
\Sigma &= \frac{1}{N}\sum_{n=1}^{N} (\mathbf{x}^n - \mu_{t^n})(\mathbf{x}^n - \mu_{t^n})^T
\end{aligned}$$

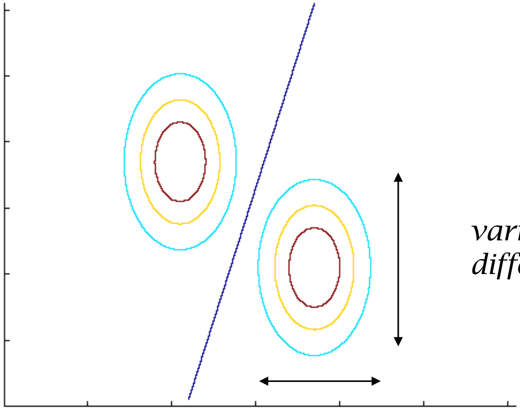
# Naive Bayes

- For Gaussian Bayes Classifier, if input $\mathbf{x}$ is high-dimensional, then covariance matrix has many parameters
- Save some parameters by using a shared covariance for the classes
- Nave Bayes is an alternative Generative model: assumes features independent given the class

$$p(\mathbf{x}|t_k) = \prod_{i=1}^{d} p(x_i|t_k)$$

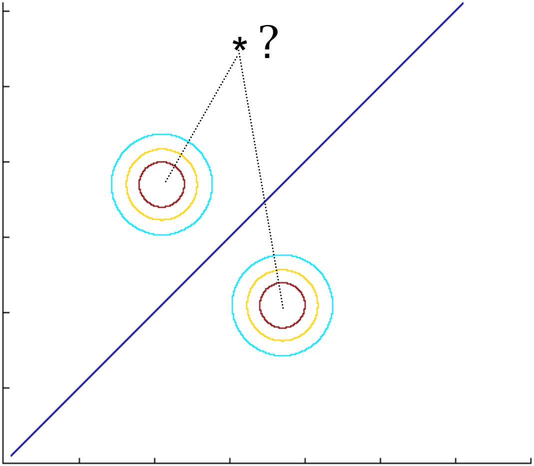
- How many parameters required now? And before?

# Diagonal Covariance



*variances may be different*

# Diagonal Covariance, isotropic




- Classification only depends on distance to the mean

# Naive Bayes Classifier

Given

- prior
- assuming features are conditionally independent given the class
- likelihood for each $x_i$

The decision rule

$$y = arg \max_k p(t_k) \prod_{i=1}^{d} p(x_i | t_k)$$

- If the assumption of conditional independence holds, NB is the optimal classifier
- If not, a heavily regularized version of generative classifier
- What's the regularization?

# Gaussian Naive Bayes

- Assume

$$p(x_i|t_k) = \frac{1}{\sqrt{2\pi}\sigma_{ik}} \exp\left[\frac{-(x_i - \mu_{ik})^2}{2\sigma_{ik}^2}\right]$$

- Maximum likelihood estimate of parameters

$$\mu_{ik} = \frac{\sum_{n=1}^{N} \mathbb{1}[t^n = k] \cdot x_i^n}{\sum_{n=1}^{N} \mathbb{1}[t^n = k]}$$

- Similar for the variance

# Gaussian Bayes Classifier (GBC) vs Logistic Regression

- If you examine $p(t = 1|\mathbf{x})$ under GBC, you will find that it looks like this:

$$p(t|\mathbf{x}, \phi, \mu_0, \mu_1, \Sigma) = \frac{1}{1 + \exp(-\mathbf{w}(\phi, \mu_0, \mu_1, \Sigma)^T \mathbf{x})}$$

- So the decision boundary has the same form as logistic regression!
- When should we prefer GBC to LR, and vice versa?

# GBC vs LR

- GBC makes stronger modeling assumption: assumes class-conditional data is multivariate Gaussian
- If this is true, GBC is asymptotically efficient (best model in limit of large N)
- But LR is more robust, less sensitive to incorrect modeling assumptions
- Many class-conditional distributions lead to logistic classifier
- When these distributions are non-Gaussian, in limit of large N, LR beats GBC

# Spam Filter Example

- Naive Bayes also applies to discrete input features (or mixed discrete/continuous)
- Represent email as feature vector, length equals number of words in vocabulary, binary feature $x_i$ is 1 iff the word i appears in email msg
- Each of these binary conditional probabilities is Bernoulli, with parameter $\phi_i$
- When we estimate parameters by maximizing joint likelihood of data, get sensible updates: $\phi_{i|t=1}$ is fraction of the spam emails in which word $i$ appears

# Laplace Smoothing

- What happens when some word appears in the test set but never in the training set?
- Counts = 0, so $\phi_{i|t=1} = \phi_{i|t=0} = 0$
- Class posterior probabilities = 0/0
- Instead use this parameter estimate:

$$\phi_{i|t=1} = \frac{1}{N} \sum_{n=1}^{N} \frac{\mathbb{1}[t^n = 1 \wedge x_i^n = 1]}{\mathbb{1}[t^n = 1] + \alpha K}$$

- $K$ is number of classes, parameter $\alpha$ acts like "pseudo-count": prior observations of words