Bayes Decision Theory - II

Nuno Vasconcelos

(Ken Kreutz-Delgado)

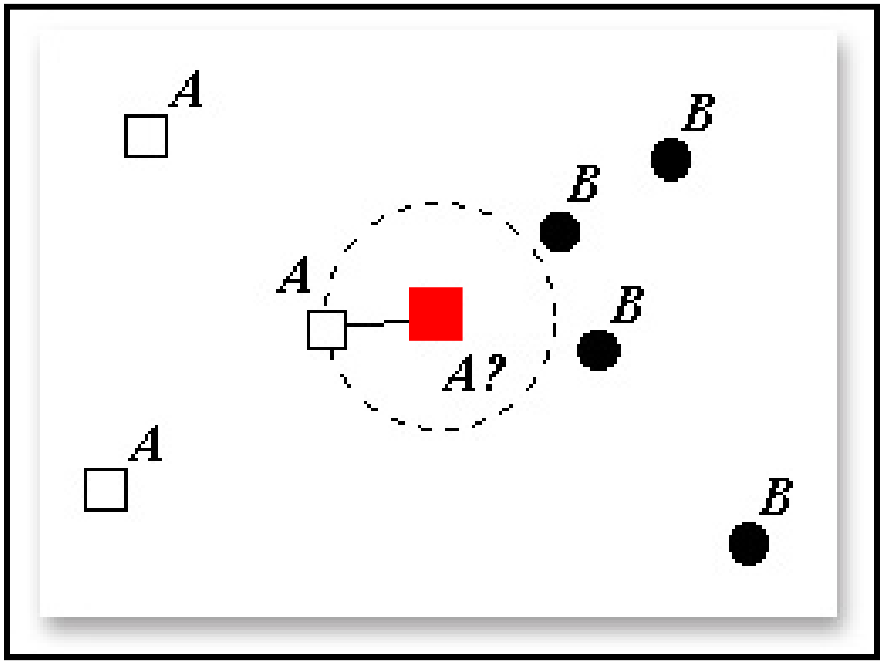
UCSD

Nearest Neighbor Classifier

- We are considering *supervised* classification
- Nearest Neighbor (NN) Classifier
  - A *training set* $\mathcal{D} = \{(x_1,y_1), \ldots, (x_n,y_n)\}$
  - x_i is a *vector of observations*, y_i is the *corresponding class label*
  - a vector x to classify
- The *"NN Decision Rule"* is

$$\boxed{\begin{aligned}&\textbf{Set} \quad y = y_{i^*}\\ &\textbf{where}\\ &i^* = \arg \min_{i \in \{1,\ldots,n\}} d(x, x_i)\end{aligned}}$$

  - argmin means: "the i that minimizes the distance"



Optimal Classifiers

- We have seen that performance depends on metric
- Some metrics are "better" than others
- The meaning of "better" is connected to how well adapted the metric is to the properties of the data
- But can we be more rigorous? what do we mean by optimal?
- To talk about optimality we define cost or loss

$$\xrightarrow{\;x\;} \boxed{f(\cdot)} \xrightarrow{\;\hat{y} = f(x)\;} \qquad \boxed{L(y, \hat{y})}$$

  - Loss is the function that we want to minimize
  - Loss depends on true y and prediction $\hat{y}$
  - Loss tells us how good our predictor is

Loss Functions

- Loss is a function of *classification errors*
  - What errors can we have?
  - Two types: *false positives* and *false negatives*
    - consider a face detection problem (decide “face” or “non-face”)
    - if you see this and say

“face”

“non-face”

    - you have a

false – positive (false alarm)

false-negative (miss, failure to detect)

  - Obviously, we have corresponding sub-classes for non-errors
    - true-positives and true-negatives
  - positive/negative part reflects what we say or decide,
  - true/false part reflects the true class label (“true state of the world”)

(Conditional) Risk

- To weigh different errors differently
  - We introduce a loss function
  - Denote the cost of classifying *X* from class *i* as *j* by

$$\boxed{L[i \rightarrow j]}$$

  - One way to measure how good the classifier is to use the (data-conditional) expected value of the loss, aka the (conditional) Risk,

$$\boxed{R(x,i) = E\{L[Y \rightarrow i] \mid x\} = \sum_j L[j \rightarrow i] P_{Y|X}(j \mid x)}$$

- this means
  - risk of classifying x as *i* is equal to
  - sum, over all classes, of the loss of classifying as *i* when truth is *j*
  - times probability that true class is *j* (given x)

Loss Functions

- example: two snakes and eating poisonous dart frogs
  - Regular snake will die
  - Frogs are a good snack for the predator dart-snake
  - This leads to the losses




Regular snake	dart frog	regular frog
regular	∞	0
dart	0	10

Predator snake	dart frog	regular frog
regular	10	0
dart	0	10

  - What is optimal decision when snakes find a frog like these?


Minimum Risk Classification

- We have seen that
  - if both snakes have

$$\boxed{P_{Y|X}(j\,|\,x) = \begin{cases} 0 & j = \text{dart} \\ 1 & j = \text{regular} \end{cases}}$$

    then both say "regular"

  - However, if

$$\boxed{P_{Y|X}(j\,|\,x) = \begin{cases} 0.1 & j = \text{dart} \\ 0.9 & j = \text{regular} \end{cases}}$$

    then the vulnerable snake says "dart"
    while the predator says "regular"

- Its infinite loss for saying regular when frog is dart, makes the vulnerable snake much more cautious!

Bayes decision rule

- Note that the definition of risk:
  - Immediately *defines the optimal classifier* as the one that *minimizes the conditional risk* for a *given observation x*
  - The *Optimal Decision* is the *Bayes Decision Rule (BDR)* :

$$\boxed{\begin{aligned} i^*(x) &= \arg\min_i R(x,i) \\ &= \arg\min_i \sum_j L[j \to i] P_{Y|X}(j \mid x). \end{aligned}}$$

  - The BDR yields the *optimal (minimal) risk* :

$$\boxed{R^*(x) = R(x, i^*) = \min_i \sum_j L[j \to i] P_{Y|X}(j \mid x)}$$

The 0/1 Loss Function

- An important special case of interest:
  - zero loss for no error and equal loss for two error types
- This is equivalent to the "zero/one" loss :

$$L[i \rightarrow j] = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases}$$

snake prediction	dart frog	regular frog
regular	1	0
dart	0	1

- Under this loss the optimal Bayes decision rule (BDR) is

$$d^*(x) = i^*(x) = \arg\min_i \sum_j L[j \rightarrow i] P_{Y|X}(j|x)$$
$$= \arg\min_i \sum_{j \neq i} P_{Y|X}(j|x)$$

0/1 Loss yields MAP Decision Rule

- Note that :

$$\boxed{\begin{aligned} i^*(x) &= \arg\min_i \sum_{j \neq i} P_{Y|X}(j|x) \\ &= \arg\min_i \left[1 - P_{Y|X}(i|x) \right] \\ &= \arg\max_i P_{Y|X}(i|x) \end{aligned}}$$

- Thus the Optimal Decision for the 0/1 loss is :
  - Pick the class that is most probable given the observation *x*
  - *i*(x)* is known as the Maximum *a Posteriori* Probability (MAP) solution
- This is also known as the Bayes Decision Rule (BDR) for the 0/1 loss
  - We will often simplify our discussion by assuming this loss
  - But you should always be aware that other losses may be used

BDR for the 0/1 Loss

- Consider the evaluation of the BDR for 0/1 loss

$$\boxed{i^*(x) = \arg\max_i P_{Y|X}(i \mid x)}$$

  - This is also called the Maximum a Posteriori Probability (MAP) rule
  - It is usually *not* trivial to evaluate the posterior probabilities $P_{Y|X}(i \mid x)$
  - This is due to the fact that we are trying to infer the cause (class *i*) from the consequence (observation *x*) – i.e. we are trying to solve a nontrivial inverse problem
    - E.g. imagine that I want to evaluate

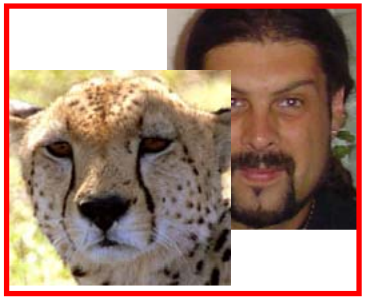
$$P_{Y|X}(\, person \mid \text{“has two eyes”})$$

    - This strongly depends on *what the other classes are*

Posterior Probabilities and Detection

- If the two classes are "people" and "cars"
  - then $P_{Y|X}$(*person* | *"has two eyes"*) = 1

- But if the classes are "people" and "cats"
  - then $P_{Y|X}$(*person* | *"has two eyes"*) = ½
    if there are equal numbers of cats and people
    to uniformly choose from [this is additional info!]



- How do we deal with this problem?
  - We note that it is much easier to infer consequence from cause
  - E.g., it is easy to infer that

$$P_{X|Y}(\text{"has two eyes"} \mid \textit{person}) = 1$$

  - This does *not* depend on any other classes
  - We do *not* need any additional information
  - Given a class, *just count* the frequency of observation

Bayes Rule

- How do we go from $P_{X|Y}(x \,|\, j)$ *to* $P_{Y|X}(j \,|\, x)$?
- We use *Bayes rule*:

$$\boxed{P_{Y|X}(i \,|\, x) = \frac{P_{X|Y}(x \,|\, i)\, P_Y(i)}{P_X(x)}}$$

- Consider the two-class problem, i.e. *Y=0* or *Y=1*
  - the BDR under 0/1 loss is

$$\boxed{\begin{aligned} i^*(x) &= \arg\max_i P_{Y|X}(i \,|\, x) \\ &= \left\{ \begin{array}{ll} 0, & \text{if } P_{Y|X}(0 \,|\, x) \geq P_{Y|X}(1 \,|\, x) \\ 1, & \text{if } P_{Y|X}(0 \,|\, x) < P_{Y|X}(1 \,|\, x) \end{array} \right\} \end{aligned}}$$

BDR for 0/1 Loss Binary Classification

- Pick "0" when $P_{Y|X}(0|x) \geq P_{Y|X}(1|x)$ and "1" otherwise
- Using Bayes rule on both sides of this inequality yields

$$P_{Y|X}(0|x) \geq P_{Y|X}(1|x) \Leftrightarrow$$

$$\frac{P_{X|Y}(x|0)P_Y(0)}{P_X(x)} \geq \frac{P_{X|Y}(x|1)P_Y(1)}{P_X(x)}$$

  – Noting that $P_X(x)$ is a non-negative quantity this is the same as the rule pick "0" when

$$P_{X|Y}(x|0)P_Y(0) \geq P_{X|Y}(x|1)P_Y(1)$$

i.e.

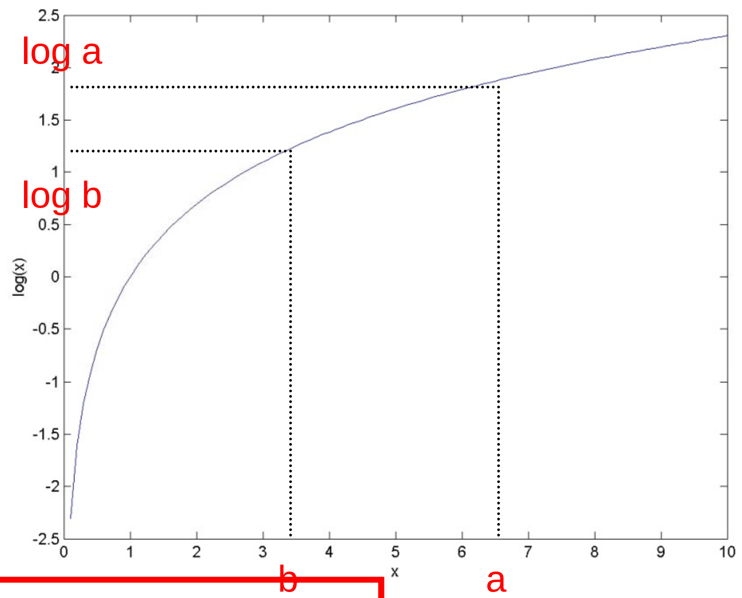
$$\boxed{i^*(x) = \arg\max_i P_{X|Y}(x|i)\, P_Y(i)}$$

The "Log Trick"

- Sometimes it's not convenient to work directly with pdf's
  - One helpful trick is to take logs
  - Note that the log is a monotonically increasing function

$$\boxed{a > b \Leftrightarrow \log a > \log b}$$

from which we have



$$\begin{aligned}
i^*(x) &= \arg\max_i P_{X|Y}(x|i)\,P_Y(i) \\
&= \arg\max_i \log\left(P_{X|Y}(x|i)\,P_Y(i)\right) \\
&= \arg\max_i \left(\log P_{X|Y}(x|i) + \log P_Y(i)\right) \\
&= \arg\min_i \left(-\log P_{X|Y}(x|i) - \log P_Y(i)\right)
\end{aligned}$$

“Standard” (0/1) BDR

- In summary
  - for the zero/one loss, the following three decision rules are optimal and equivalent

1) $$\boxed{i^*(x) = \arg\max_i P_{Y|X}(i \mid x)}$$

2) $$\boxed{i^*(x) = \arg\max_i \left[P_{X|Y}(x \mid i) P_Y(i) \right]}$$

3) $$\boxed{i^*(x) = \arg\max_i \left[\log P_{X|Y}(x \mid i) + \log P_Y(i) \right]}$$

The form 1) is usually hardest to use, 3) is frequently easier than 2)

BDR - Example

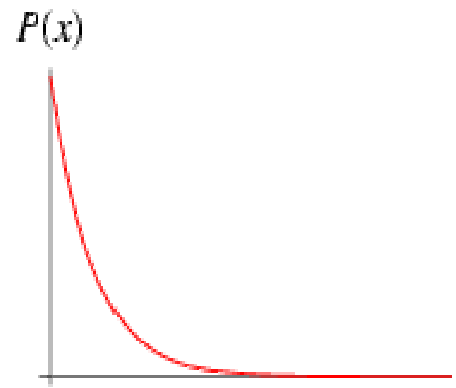
- So far the BDR is an abstract rule
  - How does one implement the optimal decision in practice?
  - In addition to having a loss function, you need to *know, model, or estimate the probabilities*!
  - Example
    - Suppose that you run a gas station
    - On Mondays you have a promotion to sell more gas
    - Q: is the promotion working? I.e., is $Y = 0$ (no) or $Y = 1$ (yes) ?
    - A good observation to answer this question is the interarrival time (τ) between cars

high τ: not working ($Y = 0$) low τ: working well ($Y = 1$)

BDR - Example

- What are the class-conditional and prior probabilities?
  - the probability of arrival of a car follows a Poisson distribution
  - Poisson inter-arrival times are exponentially distributed
    - Hence $$P_{X|Y}(\tau \mid i) = \lambda_i \mathrm{e}^{-\lambda_i \tau}$$

      where λ_i is the arrival rate (cars/s).
    - The expected value of the interarrival time is $$\mathrm{E}_{X|Y}[x \mid y = i] = 1/\lambda_i$$



    - Consecutive times are assumed to be independent :

$$P_{X_1,\ldots,X_n|Y}(\tau_1,\ldots,\tau_n \mid i) = \prod_{k=1}^{n} P_{X|Y}(\tau_k \mid i) = \prod_{k=1}^{n} \lambda_i \mathrm{e}^{-\lambda_i \tau_k}$$

BDR - Example

- Let's assume that we
  - know λ_i and the (prior) class probabilities $P_Y(i) = \pi_i$, $i = 0,1$
  - Have measured a collection of times during the day, $\mathcal{D} = \{\tau_1,\ldots,\tau_n\}$
- The probabilities are of exponential form
  - Therefore it is easier to use the log-based BDR

$$
\begin{aligned}
i^*(\mathrm{D}) &= \arg\max_i \left[\log P_{X|Y}(\mathrm{D} \mid i) + \log P_Y(i) \right] \\
&= \arg\max_i \left[\log\left(\prod_{k=1}^{n} \lambda_i \mathrm{e}^{-\lambda_i \tau_k} \right) + \log \pi_i \right] \\
&= \arg\max_i \left[-\sum_{k=1}^{n} \lambda_i \tau_k + n \log \lambda_i + \log \pi_i \right] \\
&= \arg\max_i \left[-\sum_{k=1}^{n} \lambda_i \tau_k + n \log\left(\lambda_i \sqrt[n]{\pi_i} \right) \right]
\end{aligned}
$$

BDR - Example

- This means we pick "0" when

$$-\sum_{k=1}^{n}\lambda_0\,\tau_k + n\log\left(\lambda_0\sqrt[n]{\pi_0}\right) \geq -\sum_{k=1}^{n}\lambda_1\,\tau_k + n\log\left(\lambda_1\sqrt[n]{\pi_1}\right), \quad \text{or}$$

$$(\lambda_1-\lambda_0)\sum_{k=1}^{n}\tau_k \geq n\log\left(\frac{\lambda_1\sqrt[n]{\pi_1}}{\lambda_0\sqrt[n]{\pi_0}}\right), \quad \text{or}$$

$$\boxed{\frac{1}{n}\sum_{k=1}^{n}\tau_k \geq \frac{1}{(\lambda_1-\lambda_0)}\log\left(\frac{\lambda_1\sqrt[n]{\pi_1}}{\lambda_0\sqrt[n]{\pi_0}}\right)}$$

(reasonably taking $\lambda_1 > \lambda_0$)

and "1" otherwise

- Does this decision rule make sense?
  - Let's assume, for simplicity, that $\pi_1 = \pi_2 = 1/2$

BDR - Example

- For $\pi_1 = \pi_2 = ½$, we pick "promotion did not work" *(Y=0)* if

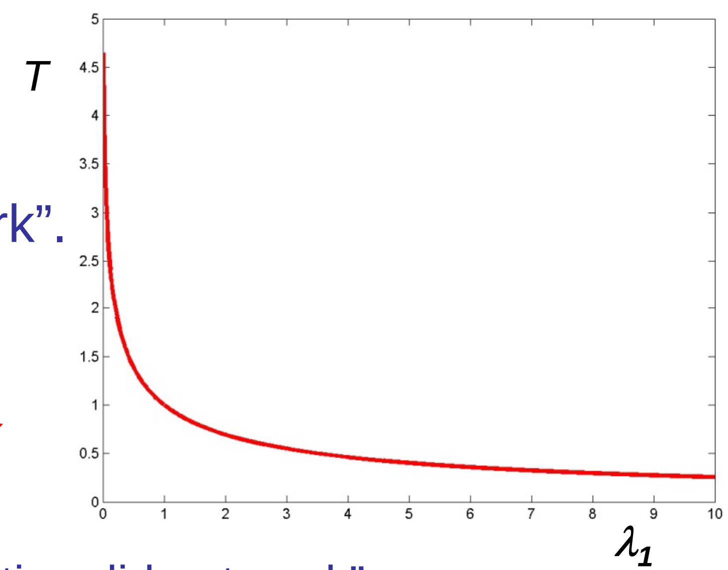
$$\boxed{\frac{1}{n}\sum_{k=1}^{n}\tau_k \geq \frac{1}{(\lambda_1 - \lambda_0)}\log\left(\frac{\lambda_1}{\lambda_0}\right)}$$

The left hand side is the (sample) average interarrival time for the day

– This means that there is an optimal choice of a *"threshold"*

$$\boxed{T = \frac{1}{(\lambda_1 - \lambda_0)}\log\left(\frac{\lambda_1}{\lambda_0}\right)}$$

above which we say "promotion did not work". This makes sense!

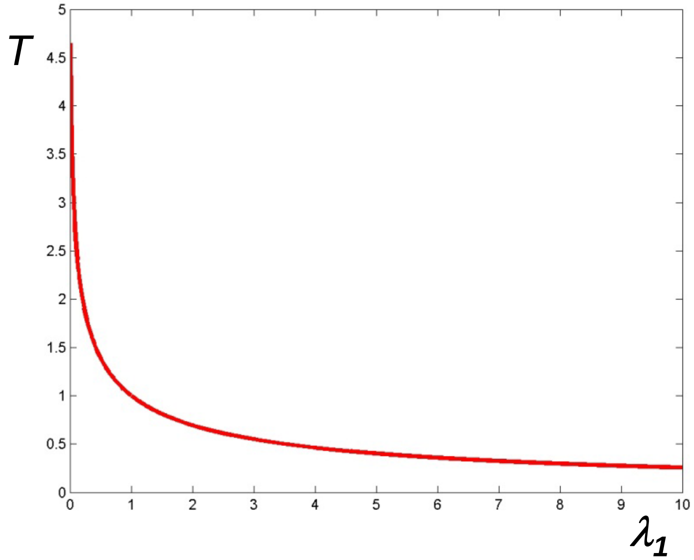


– What is the shape of this threshold?
  - Assuming $\lambda_0 = 1$, it looks like this.
  - Higher the λ_1, the more likely to say "promotion did not work".

BDR - Example

- When $\pi_1 = \pi_2 = ½$, we pick "did not work" *(Y=0)* when

$$\boxed{\frac{1}{n}\sum_{k=1}^{n}\tau_k \geq T} \qquad \boxed{T = \frac{1}{(\lambda_1 - \lambda_0)}\log\left(\frac{\lambda_1}{\lambda_0}\right)}$$



  - Assuming $\lambda_0 = 1$, T decreases with λ_1
  - I.e. for a given daily average,
    - Larger λ_1: easier to say "did not work"
  - This means that
    - As the expected rate of arrival for good days increases we are going to impose a tougher standard on the average measured interarrival times
    - The average has to be smaller for us to accept the day as a good one
  - Once again, *this makes sense!*
  - usually the case with the BDR (a good way to check your math)

The Gaussian Classifier

- One important case is that of Multivariate Gaussian Classes
  - The pdf of class *i* is a Gaussian of mean μ_i and covariance Σ_i

$$\boxed{P_{X|Y}(x \mid i) = \frac{1}{\sqrt{(2\pi)^d \,|\Sigma_i|}} \exp\left\{ -\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i) \right\}}$$

- The BDR is

$$\boxed{\begin{aligned} i^*(x) = \arg\max_i \Big[& -\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i) \\ & -\frac{1}{2}\log(2\pi)^d \left|\Sigma_i\right| + \log P_Y(i) \Big] \end{aligned}}$$

Implementation

- To design a Gaussian classifier (e.g. homework)
  - Start from a collection of datasets, where the *i*-th class dataset $\mathcal{D}^{(i)} = \{x_1^{(i)}, \ldots, x_n^{(i)}\}$ is a set of $n^{(i)}$ examples from class *i*
  - For each class *estimate* the Gaussian parameters :

$$\boxed{\hat{\mu}_i = \frac{1}{n^{(i)}} \sum_j x_j^{(i)}} \quad \boxed{\hat{\Sigma}_i = \frac{1}{n^{(i)}} \sum_j (x_j^{(i)} - \hat{\mu}_i)(x_j^{(i)} - \hat{\mu}_i)^T} \quad \boxed{\hat{P}_Y(i) = \frac{n^{(i)}}{T}}$$

  where T is the total number of examples over all c classes

- the BDR is approximated as

$$\boxed{\begin{aligned} i^*(x) = \arg\max_i \Big[& -\frac{1}{2}(x - \hat{\mu}_i)^T \hat{\Sigma}_i^{-1} (x - \hat{\mu}_i) \\ & -\frac{1}{2}\log(2\pi)^d \left|\hat{\Sigma}_i\right| + \log \hat{P}_Y(i) \Big] \end{aligned}}$$

Gaussian Classifier

- The Gaussian Classifier can be written as

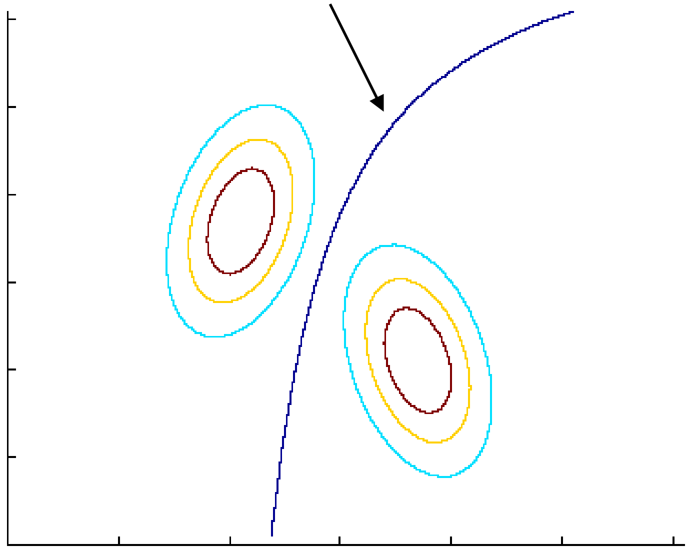
$$\boxed{i^*(x) = \arg\min_i \left[d^2{}_i(x, \mu_i) + \alpha_i \right]}$$

with

$$\boxed{d^2{}_i(x, y) = (x - y)^T \Sigma_i^{-1} (x - y)}$$

$$\boxed{\alpha_i = \log(2\pi)^d |\Sigma_i| - 2 \log P_Y(i)}$$

discriminant:
$P_{Y|X}(1|\mathbf{x}) = 0.5$



and can be seen as a nearest "class-neighbor" classifier with a "funny metric"

- Each class has its own "distance" measure:
  - Sum the Mahalanobis-squared for that class, then add the α constant.
  - We effectively have different "metrics" in the data (feature) space that are class i dependent.

Gaussian Classifier

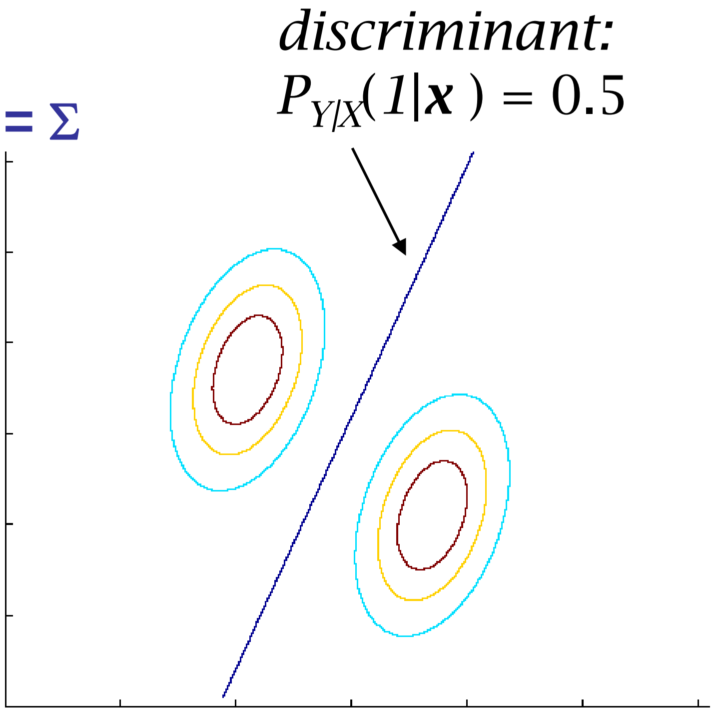
- A special case of interest is when
  - All classes have the same covariance $\Sigma_i = \Sigma$

$$\boxed{i^*(x) = \arg\min_i \left[d^2(x, \mu_i) + \alpha_i \right]}$$

with

$$\boxed{d^2(x, y) = (x - y)^T \Sigma^{-1} (x - y)}$$

$$\boxed{\alpha_i = -2 \log P_Y(i)}$$



- Note that:
  - α_i can be dropped when all classes have *equal prior probability*
  - This is reminiscent of the NN classifier with Mahalanobis distance
  - Instead of finding the *nearest data point neighbor* of x, it looks for the *nearest class "prototype*," (or "archetype," or "exemplar," or "template," or "representative", or "ideal", or "form") , defined as the class mean μ_i

Binary Classifier – Special Case

- Consider $\Sigma_i = \Sigma$ with two classes
  - One important property of this case is that the *decision boundary* is a hyperplane (Homework)
  - This can be shown by computing the set of points x such that

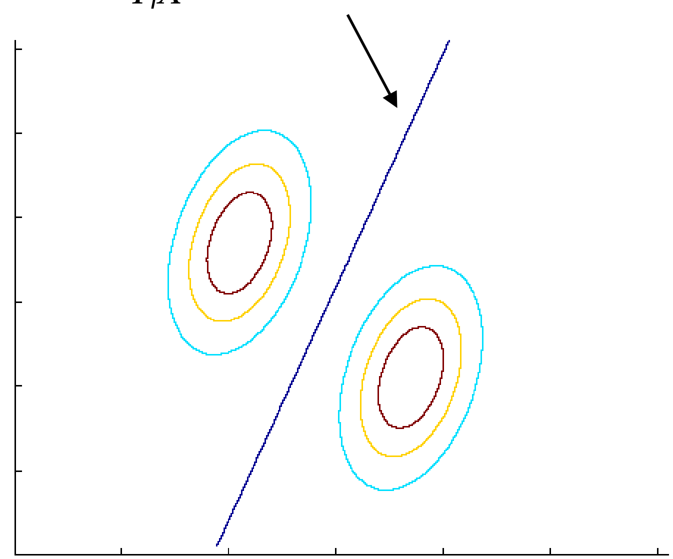
$$\boxed{d^2(x,\mu_0)+\alpha_0 = d^2(x,\mu_1)+\alpha_1}$$

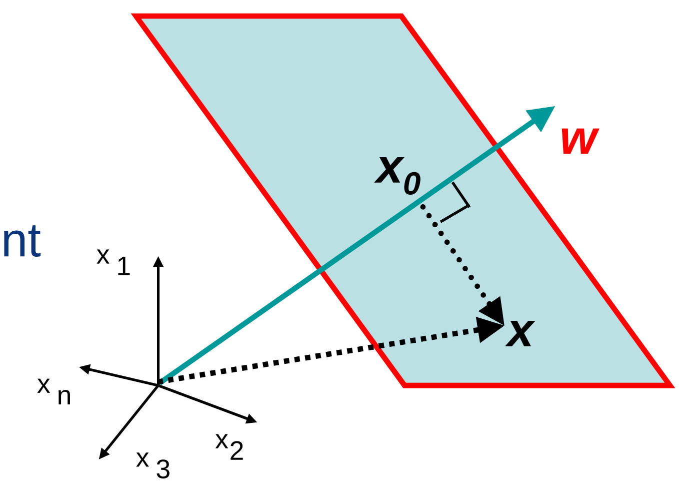
and showing that they satisfy

$$\boxed{w^T(x - x_0) = 0}$$

  - This is the equation of a *hyperplane* with normal w. x_0 can be any fixed point on the hyperplane, but it is *standard* to choose it to have minimum norm, in which case w and x_0 are then parallel

Discriminant Surface:
$P_{Y|X}(1|\mathbf{x}) = 0.5$





Gaussian Classifier

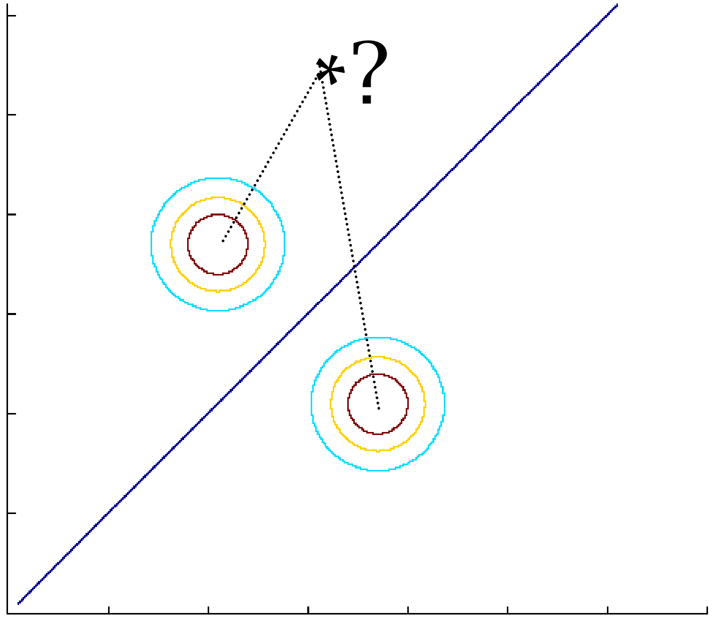
- If *all* the class covariances are the identity, $\Sigma_i = I$, then

$$\boxed{i^*(x) = \arg\min_i \left[d^2(x, \mu_i) + \alpha_i \right]}$$

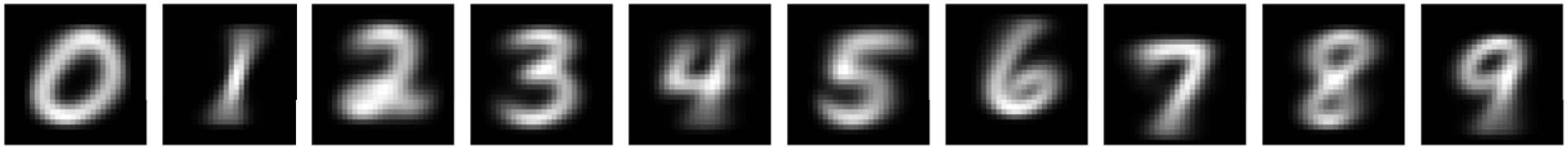
with

$$\boxed{d^2(x, y) = \| x - y \|^2}$$

$$\boxed{\alpha_i = -2 \log P_Y(i)}$$



- This is called template matching with class means as templates
  - E.g. for digit classification



Compare the complexity of this classifier to NN Classifier!

The Sigmoid Function

- We have derived much of the above from the log-based BDR

$$\boxed{i^*(x) = \arg\max_i \left[\log P_{X|Y}(x|i) + \log P_Y(i) \right]}$$

- When there are only two classes, $i = 0,1$, it is also interesting to consider the original definition

$$\boxed{i^*(x) = \arg\max_i g_i(x)}$$

where

$$\boxed{\begin{aligned} g_i(x) = P_{Y|X}(i|x) &= \frac{P_{X|Y}(x|i)P_Y(i)}{P_X(x)} \\ &= \frac{P_{X|Y}(x|i)P_Y(i)}{P_{X|Y}(x|0)P_Y(0) + P_{X|Y}(x|1)P_Y(1)} \end{aligned}}$$

The Sigmoid Function

- Note that this can be written as

$$\boxed{i^*(x) = \arg\max_i g_i(x)}$$

$$\boxed{g_1(x) = 1 - g_0(x)}$$

$$\boxed{g_0(x) = \frac{1}{1 + \frac{P_{X|Y}(x|1)\,P_Y(1)}{P_{X|Y}(x|0)\,P_Y(0)}}}$$

- For Gaussian classes, the posterior probabilities are

$$\boxed{g_0(x) = \frac{1}{1 + \exp\left\{d^2{}_0(x,\mu_0) - d^2{}_1(x,\mu_1) + \alpha_0 - \alpha_1\right\}}}$$

where, as before,

$$\boxed{d^2{}_i(x,y) = (x-y)^T \Sigma_i^{-1}(x-y)}$$

$$\boxed{\alpha_i = \log(2\pi)^d\left|\Sigma_i\right| - 2\log P_Y(i)}$$

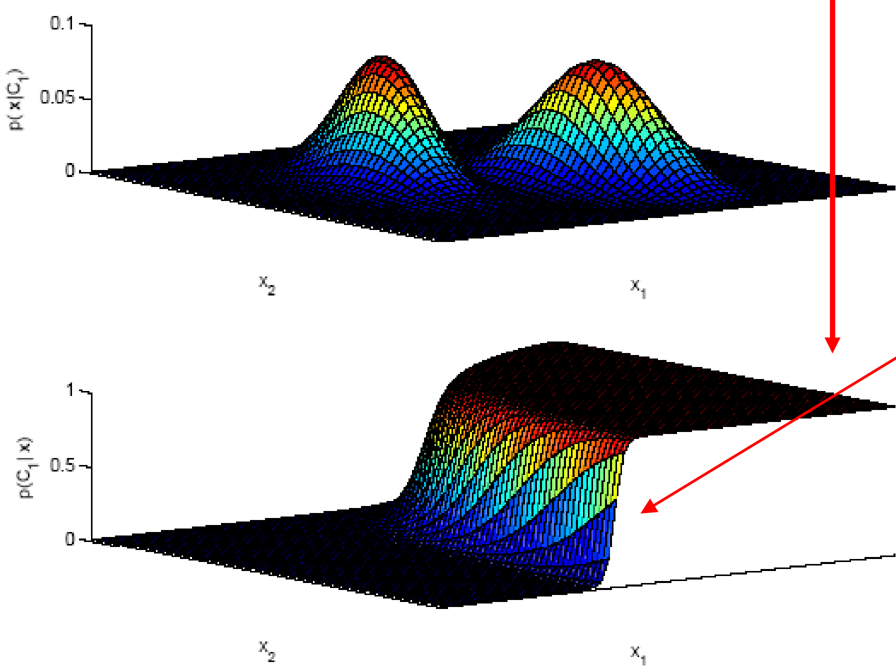
The Sigmoid ("S-shaped") Function

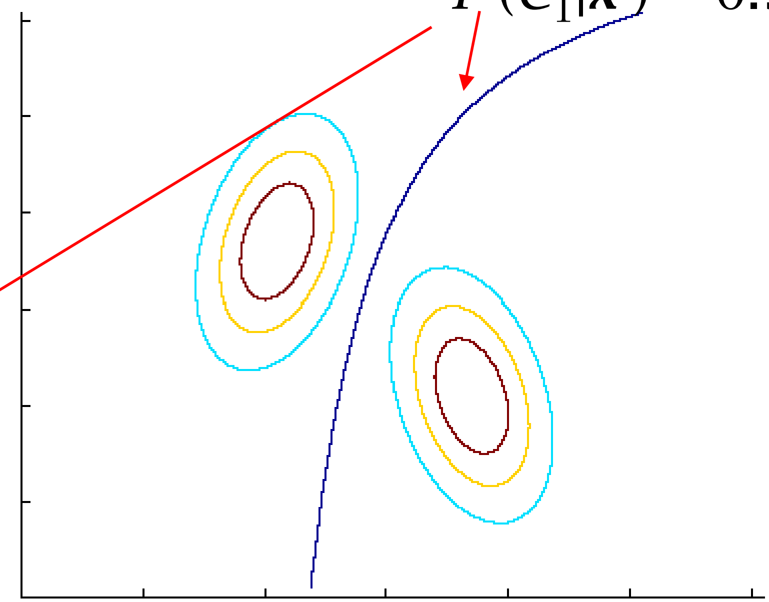
- The posterior pdf for class ***i = 0***,

$$\boxed{g_0(x) = \frac{1}{1+\exp\left\{d^2{}_0(x,\mu_0) - d^2{}_1(x,\mu_1) + \alpha_0 - \alpha_1\right\}}}$$

is a sigmoid and looks like this

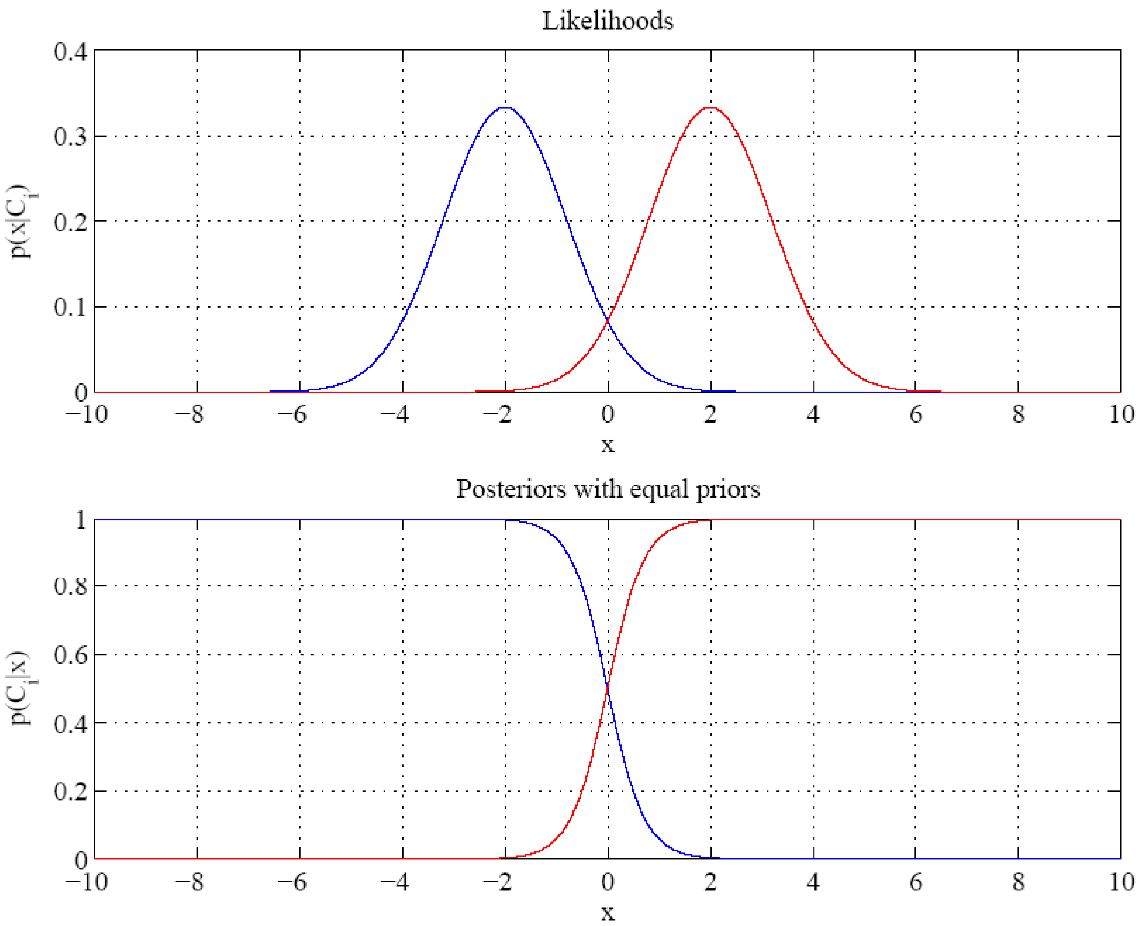
discriminant:
$P\,(C_1|\boldsymbol{x}\,) = 0.5$





The Sigmoid

- The sigmoid appears in neural networks, where it can be interpreted as a posterior pdf for a Gaussian binary classification problem when the covariances are the same

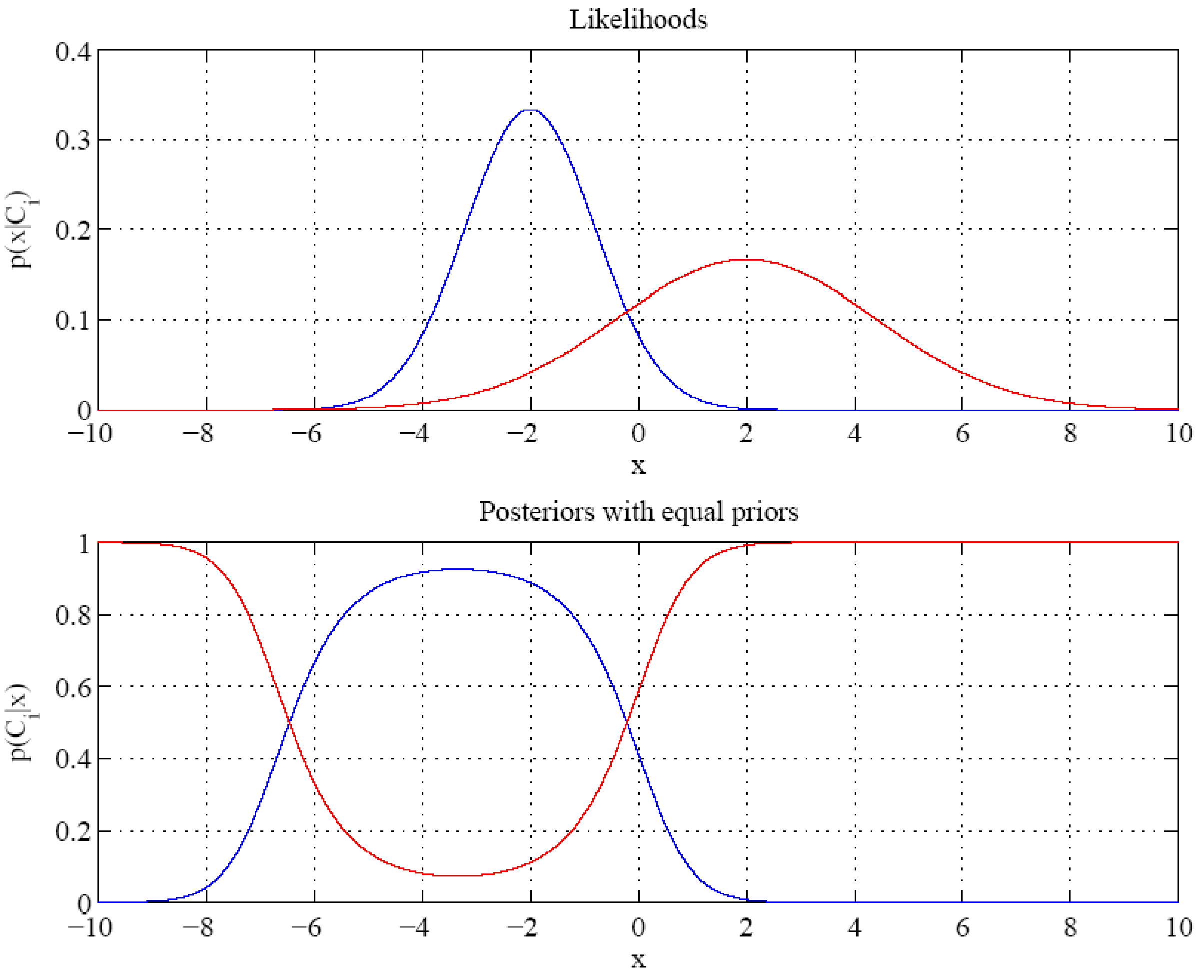


Equal variances

Single boundary at halfway between means

The Sigmoid

- But not necessarily when the covariances are *different*



Variances are different

Yields two boundaries

Any questions?