SYDE 372
Introduction to Pattern Recognition

Distance Measures for Pattern Classification: Part II

Alexander Wong

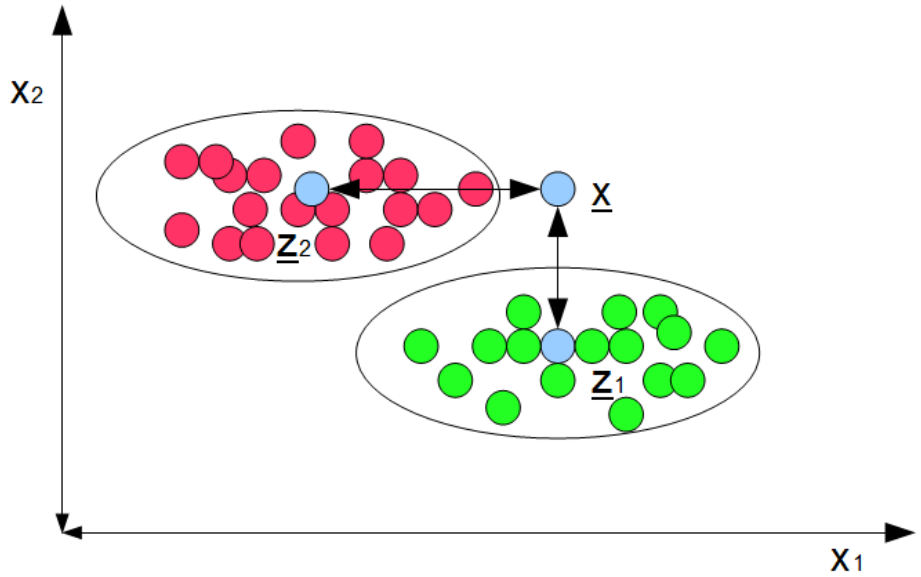
Department of Systems Design Engineering
University of Waterloo

Outline

Weighted Euclidean distance metric

- Motivation: problem with using Euclidean distance is that pattern space in general is NOT in Euclidean vector space!
- Different measurements and features may:
  - be more or less dependent
  - have different units and scales
  - have different variances
- The use of Euclidean distance can lead to poor classification performance in certain cases where the above situations hold true.

Example where Euclidean distance can cause issues



- The Euclidean distance from $\underline{x}$ to class mean prototype $\underline{z}_1$ is shorter than that to cluster mean prototype $\underline{z}_2$, even though intuitively it should belong to class 2.
- Could use NN prototypes, but that is more computationally expensive and less robust to noise

Weighted Euclidean distance metric

- Idea: Since the features may have different units, scales, and variances, why don't we weight the features differently when measuring distances?

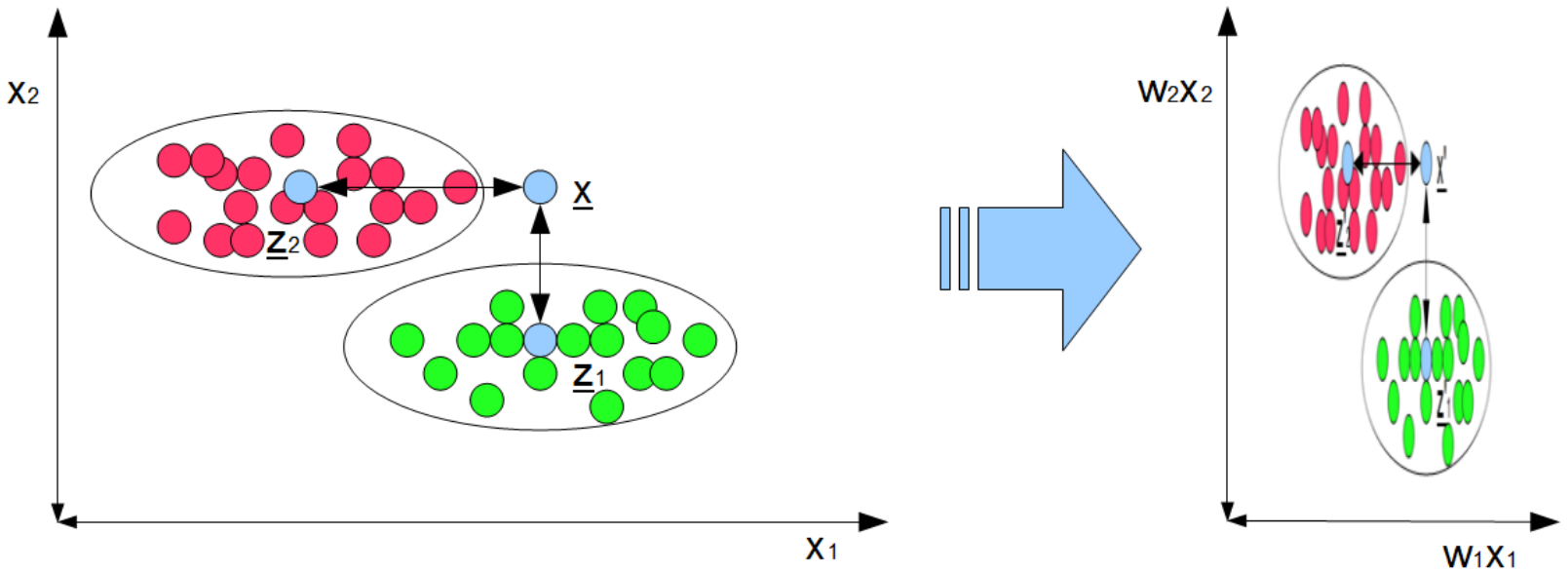
$$d_{W_o}(\underline{x}, \underline{z}) = \left[\sum_{i=1}^{n} (w_i(x_i - z_i))^{\frac{1}{2}} \right]^{\frac{1}{2}} \tag{1}$$

- What we are doing is essentially scaling the feature axes with a linear transformation and then applying Euclidean distance metric.

$$d_{W_o}(\underline{x}, \underline{z}) = d_E(\underline{x}', \underline{z}') \tag{2}$$

where $\underline{x}' = W_o\underline{x}$, $\underline{z}' = W_o\underline{x}$, and W_o is a diagonal matrix of weights.

Example revisited



- The weighted Euclidean distance in the original feature space is just Euclidean distance in transformed space!

WED Classifier: Example

- Suppose we are given the following statistical information about the classes:
  - Class 1: $\underline{m}_1 = [3 \;\; 3]^T, S_1 = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$.
  - Class 2: $\underline{m}_2 = [4 \;\; 5]^T, S_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$.
- Suppose we wish to build a WED classifier using sample means as prototypes and the following weight matrices $W_{0,1}$ and $W_{0,2}$:

$$W_{0,1} = \begin{bmatrix} 1/\sqrt{3} & 0 \\ 0 & 1 \end{bmatrix} \quad W_{0,2} = \begin{bmatrix} 1/\sqrt{2} & 0 \\ 0 & 1 \end{bmatrix} \tag{3}$$

  - Compute the discriminate function for each class.
  - Compute the decision boundary.

WED Classifier: Example

- Step 1: Find discriminant functions for each class based on WED decision rule:
- Recall that the WED decision criteria for the two class case is:

$$d_{W_o}(\underline{x}, \underline{z}_1) < d_{W_o}(\underline{x}, \underline{z}_2) \tag{4}$$

$$[(\underline{x}-\underline{z}_1)^T W_{o,1}^T W_{o,1}(\underline{x}-\underline{z}_1)]^{1/2} < [(\underline{x}-\underline{z}_2)^T W_{o,2}^T W_{o,2}(\underline{x}-\underline{z}_2)]^{1/2} \tag{5}$$

$$(\underline{x}-\underline{z}_1)^T W_{o,1}^T W_{o,1}(\underline{x}-\underline{z}_1) < (\underline{x}-\underline{z}_2)^T W_{o,2}^T W_{o,2}(\underline{x}-\underline{z}_2) \tag{6}$$

WED Classifier: Example

- Plugging in $z_1 = m_1$, $z_2 = m_2$, $W_{o,1}$, and $W_{o,2}$ gives us:

$$(\underline{x}-\underline{z}_1)^T W_{o,1}^T W_{o,1}(\underline{x}-\underline{z}_1) < (\underline{x}-\underline{z}_2)^T W_{o,2}^T W_{o,2}(\underline{x}-\underline{z}_2) \quad (7)$$

$$\begin{aligned}&([x_1\ x_2]^T - [3\ 3]^T)^T \begin{bmatrix} 1/\sqrt{3} & 0 \\ 0 & 1 \end{bmatrix}^T \begin{bmatrix} 1/\sqrt{3} & 0 \\ 0 & 1 \end{bmatrix} ([x_1\ x_2]^T - [3\ 3]^T) \\ &< ([x_1\ x_2]^T - [4\ 5]^T)^T \begin{bmatrix} 1/\sqrt{2} & 0 \\ 0 & 1 \end{bmatrix}^T \begin{bmatrix} 1/\sqrt{2} & 0 \\ 0 & 1 \end{bmatrix} ([x_1\ x_2]^T - [4\ 5]^T)\end{aligned} \quad (8)$$

$$\begin{aligned}&([x_1 - 3\ \ x_2 - 3]) \begin{bmatrix} 1/3 & 0 \\ 0 & 1 \end{bmatrix} ([x_1 - 3\ \ x_2 - 3])^T \\ &< ([x_1 - 4\ \ x_2 - 5]) \begin{bmatrix} 1/2 & 0 \\ 0 & 1 \end{bmatrix} ([x_1 - 4\ \ x_2 - 5])^T\end{aligned} \quad (9)$$

WED Classifier: Example

- Plugging in $z_1 = m_1$, $z_2 = m_2$, $W_{o,1}$, and $W_{o,2}$ gives us:

$$\begin{aligned} &([x_1 - 3 \;\; x_2 - 3]) \begin{bmatrix} 1/3 & 0 \\ 0 & 1 \end{bmatrix} ([x_1 - 3 \;\; x_2 - 3])^T \\ &< ([x_1 - 4 \;\; x_2 - 5]) \begin{bmatrix} 1/2 & 0 \\ 0 & 1 \end{bmatrix} ([x_1 - 4 \;\; x_2 - 5])^T \end{aligned} \tag{10}$$

$$\begin{aligned} &([(x_1 - 3)/3 \;\; x_2 - 3])([x_1 - 3 \;\; x_2 - 3])^T \\ &< ([(x_1 - 4)/2 \;\; x_2 - 5])([x_1 - 4 \;\; x_2 - 5])^T \end{aligned} \tag{11}$$

$$(x_1 - 3)^2/3 + (x_2 - 3)^2 < (x_1 - 4)^2/2 + (x_2 - 5)^2 \tag{12}$$

WED Classifier: Example

- Expanding gives us:

$$(x_1 - 3)^2/3 + (x_2 - 3)^2 < (x_1 - 4)^2/2 + (x_2 - 5)^2 \quad (13)$$

$$2x_1^2 + 6x_2^2 - 12x_1 - 36x_2 + 72 < 3x_1^2 + x_2^2 - 24x_1 - 10x_2 + 73 \quad (14)$$

- Therefore, the discriminant functions are:

$$g_1(x_1, x_2) = 2x_1^2 + 6x_2^2 - 12x_1 - 36x_2 + 72 \quad (15)$$

$$g_2(x_1, x_2) = 3x_1^2 + x_2^2 - 24x_1 - 10x_2 + 73 \quad (16)$$

MED Classifier: Decision Boundary

- Step 2: Find decision boundary between classes 1 and 2
- For WED classifier, the decision boundary is

$$g(x_1, x_2) = g_1(x_1, x_2) - g_2(x_1, x_2) = 0. \tag{17}$$

Plugging in the discriminant functions g_1 and g_2 gives us:

$$\begin{aligned} g(x_1, x_2) &= 2x_1^2 + 6x_2^2 - 12x_1 - 36x_2 + 72 \\ &-(3x_1^2 + x_2^2 - 24x_1 - 10x_2 + 73) = 0 \end{aligned} \tag{18}$$

WED Classifier: Decision Boundary

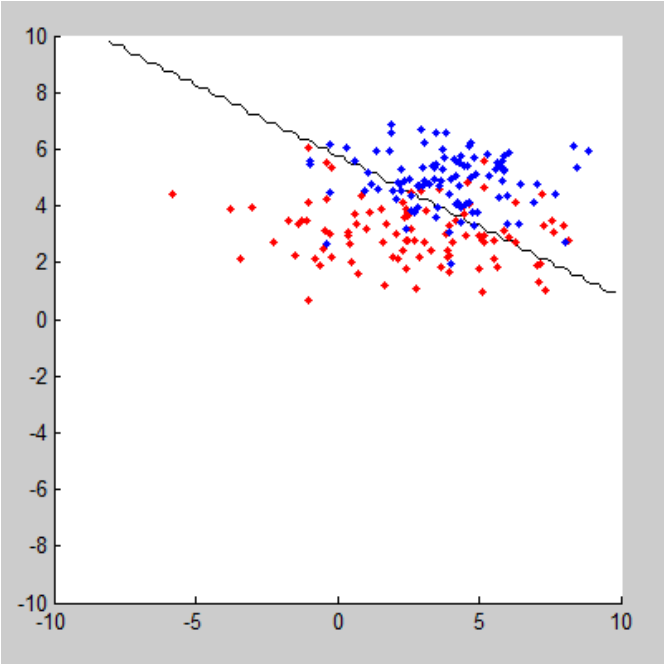
- Grouping terms:

$$g(x_1, x_2) = -x_1^2 + 5x_2^2 + 12x_1 - 26x_2 - 1 = 0 \qquad (19)$$

- Therefore, the decision boundary is a quadratic!

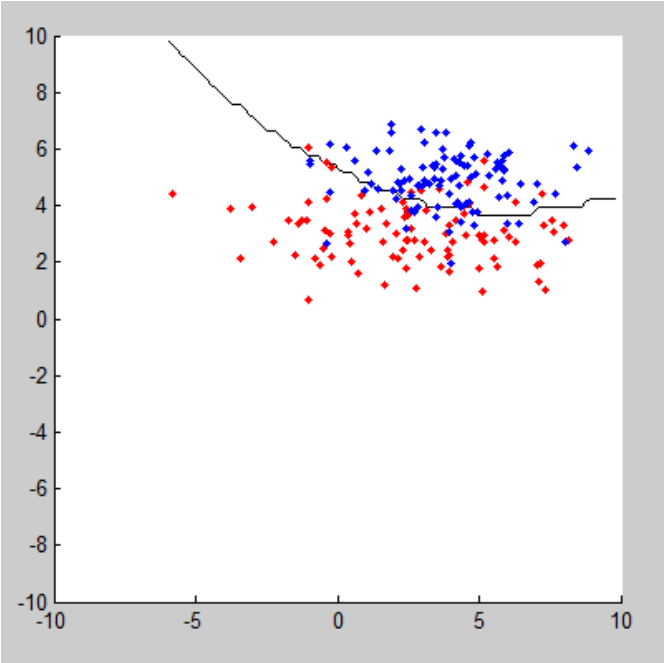
WED Classifier: Decision Boundary

- The decision boundary for a MED classifier looks like this



WED Classifier: Decision Boundary

- The decision boundary for this WED classifier looks like this



Weighted Euclidean distance metric

- A more general form of the weighted Euclidean distance metric can be defined as:

$$d_W(\underline{x}, \underline{z}) = \left[(\underline{x} - \underline{z})^T W^T W(\underline{x} - \underline{z})\right]^{\frac{1}{2}} \tag{20}$$

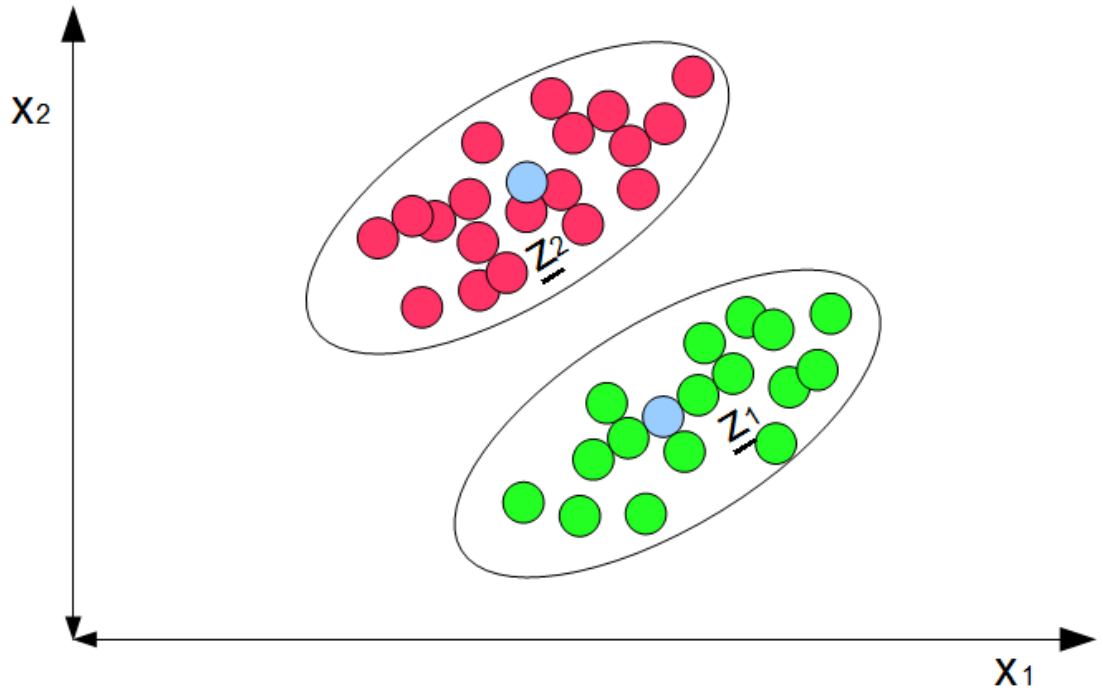
where W is the general weight matrix of the form:

$$\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{12} & & \\ \vdots & & \ddots & \\ w_{n1} & & & w_{nn} \end{bmatrix} \tag{21}$$

- Allows scaling AND rotation of axes!

Weighted Euclidean distance metric

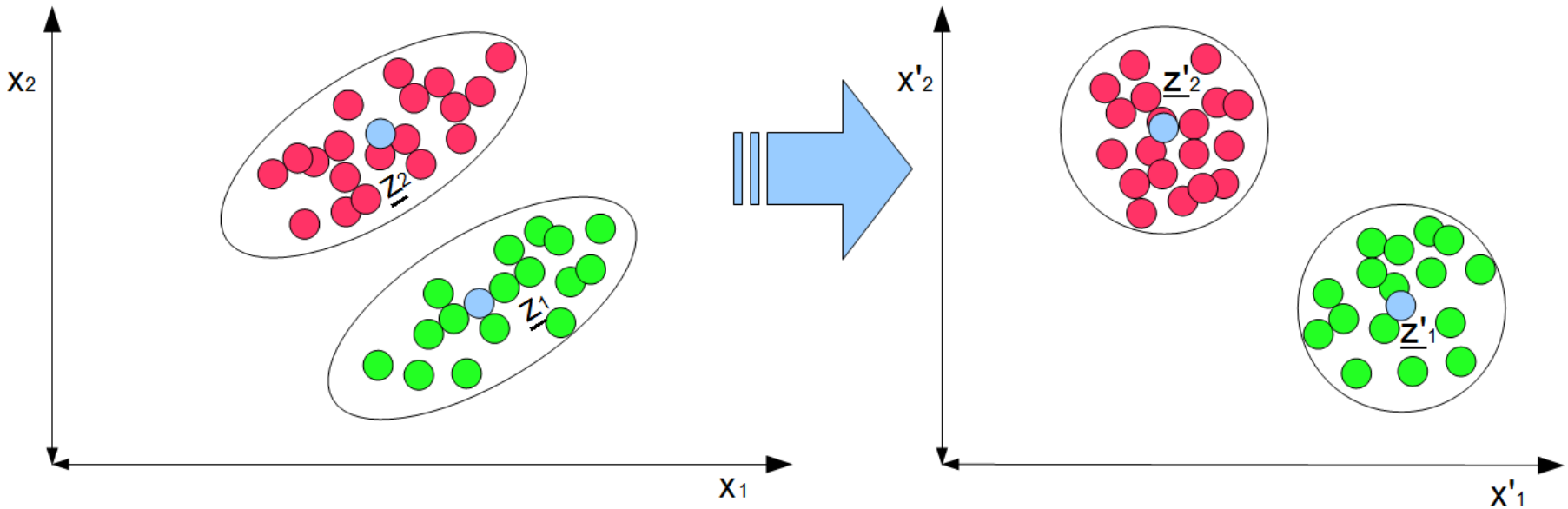
- Question: Why do we care about rotation of the axes?
- Answer: Cases like this...



Orthonormal Covariance Transforms

- Question: How do we determine the weights W?
- Intuition: Euclidean distance is only valid for cases where features are:
  - uncorrelated
  - unit variance
- Visually, shape of distribution in feature space is a hypersphere.
- Therefore, we wish to find W that transforms the shape of the distribution into a hypersphere!

Desired Transform: Visualization



Orthonormal Covariance Transforms

- Question: How do we compute this transformation?
- Intuition: As a first step, we wish to transform the samples into a space in which features are uncorrelated
- This can be accomplished by finding the transform A that diagonalizes the covariance matrix Σ (since non-zero non-diagonal elements in covariance matrix implies correlation)

$$A\Sigma A^T = \Lambda = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & & \\ \vdots & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix} \tag{22}$$

Orthonormal Covariance Transforms

- A covariance matrix can be diagonalized based on the following formulation:

$$\Phi^T \Sigma \Phi = \Lambda = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & & \\ \vdots & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix} \tag{23}$$

where the columns of Φ are the eigenvectors of Σ, and the elements of Λ are the eigenvalues

Orthonormal Covariance Transforms

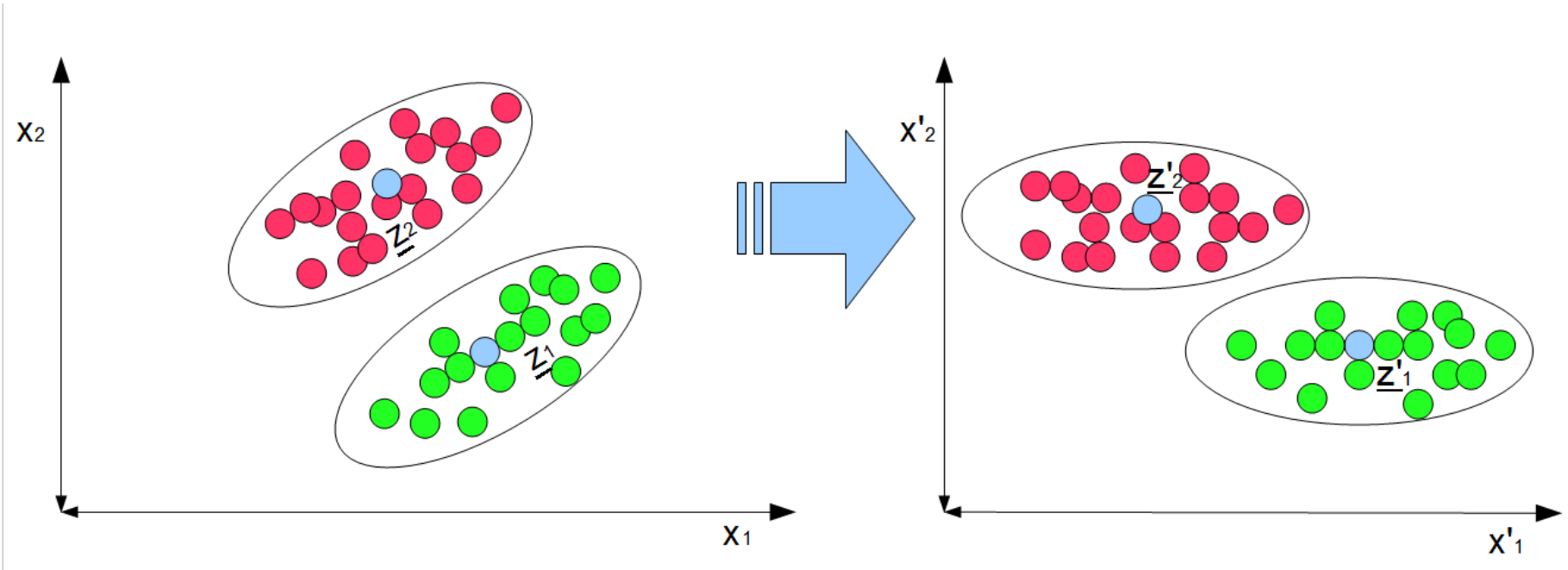
- Therefore, our transform A that diagonalizes Σ is

$$A = \begin{bmatrix} \underline{\phi}_1^T \\ \underline{\phi}_2^T \\ \vdots \\ \underline{\phi}_n^T \end{bmatrix} \tag{24}$$

- The new covariance matrix in transformed space is

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & & \\ \vdots & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix} \tag{25}$$

Orthonormal Covariance Transform: Visualization



Generalized Euclidean Metric

- We wish to find a transform W that transform the samples into a space in which
  - features are uncorrelated
  - features have unit variances
- The orthonormal covariance transform solves the first part of the problem (getting uncorrelated features)
- Now we need to transform these uncorrelated features into ones with unit variances

Generalized Euclidean Metric

- Intuition:
  - The eigenvalues Λ are the variances along the principal axes in the transformed space after orthonormal covariance transform
  - To achieve equal variance features, we just need to scale the feature axes based on their eignvalues!
- The necessary scaling transformation (whitening trnasformation) is $\Lambda^{\frac{-1}{2}}$:

$$\Lambda^{\frac{-1}{2}} = \begin{bmatrix} \frac{1}{\sqrt{\lambda_1}} & 0 & \dots & 0 \\ 0 & \frac{1}{\sqrt{\lambda_2}} & & \\ \vdots & & \ddots & \\ 0 & & & \frac{1}{\sqrt{\lambda_n}} \end{bmatrix} \tag{26}$$

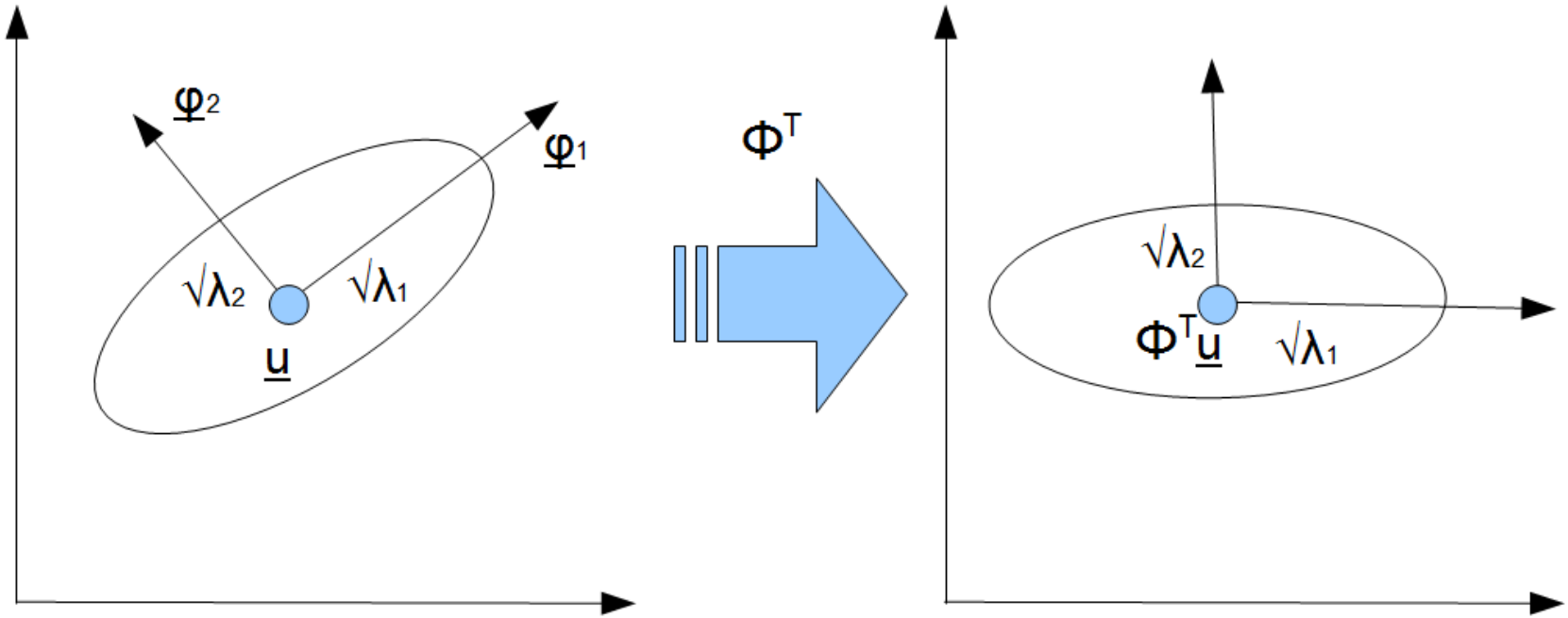
Generalized Euclidean Metric

- Therefore, the weight matrix W that we want can be defined as:

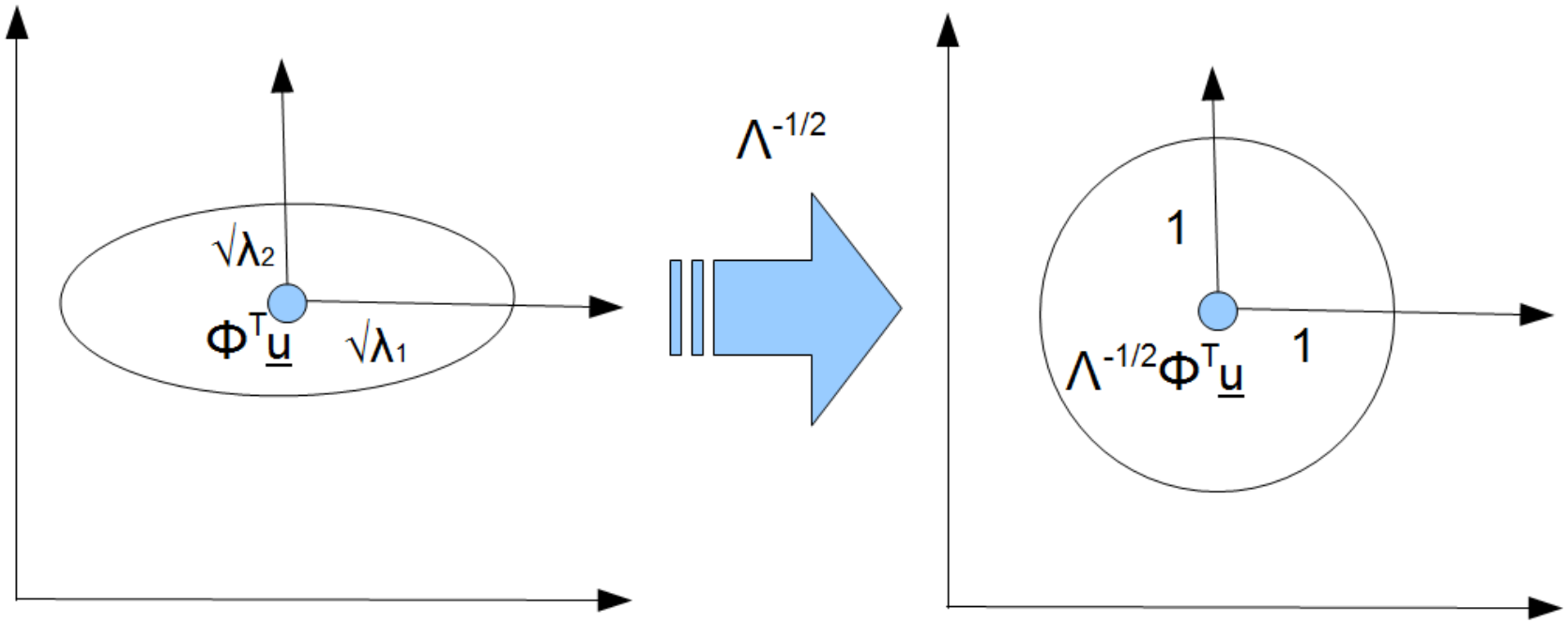
$$W = \Lambda^{\frac{-1}{2}} \Phi^T \tag{27}$$

- What this weight matrix does is:
  1. Rotate coordinate axes to get diagonal covariance matrix
  2. Scale the axes to obtain identity matrix

Generalized Euclidean Metric: Visualization



Generalized Euclidean Metric: Visualization



Generalized Euclidean Metric

- Given W, the final generalized Euclidean metric is:

$$d_G(\underline{x}, \underline{z}) = \left[(\underline{x} - \underline{z})^T (\Lambda^{-\frac{1}{2}} \Phi^T)^T (\Lambda^{-\frac{1}{2}} \Phi^T)(\underline{x} - \underline{z})\right]^{\frac{1}{2}} \quad (28)$$

- Simplifying the formulation gives:

$$d_G(\underline{x}, \underline{z}) = \left[(\underline{x} - \underline{z})^T (\Phi \Lambda^{-\frac{1}{2}} \Lambda^{-\frac{1}{2}} \Phi^T)(\underline{x} - \underline{z})\right]^{\frac{1}{2}} \quad (29)$$

$$d_G(\underline{x}, \underline{z}) = \left[(\underline{x} - \underline{z})^T (\Phi \Lambda^{-1} \Phi^T)(\underline{x} - \underline{z})\right]^{\frac{1}{2}} \quad (30)$$

$$d_G(\underline{x}, \underline{z}) = \left[(\underline{x} - \underline{z})^T (S^{-1})(\underline{x} - \underline{z})\right]^{\frac{1}{2}} \quad (31)$$

- Makes perfect sense since $S^{-1}S = I$, thus giving us features that are uncorrelated and have unit variance.

Generalized Euclidean Metric: Example

- Given a class with the following mean and covariance matrix:

$$\underline{m} = [10 \;\; 0]^T \tag{32}$$

$$S = \begin{bmatrix} 16 & -12 \\ -12 & 34 \end{bmatrix} \tag{33}$$

- Plot unit standard deviation contour in original space.
- Find the transformation that yields equal, unit variance features. Plot unit standard deviation contour in transformed space.

Generalized Euclidean Metric: Example

- Compute eigenvalues λ_1 and λ_2

$$\det(S - \lambda I) = 0 \tag{34}$$

$$\det(\begin{bmatrix} 16 & -12 \\ -12 & 34 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}) = 0 \tag{35}$$

$$(16 - \lambda)(34 - \lambda) - 144 = 0 \tag{36}$$

$$(\lambda - 40)(\lambda - 10) = 0 \tag{37}$$

- Therefore, the eigenvalues are $\lambda_1 = 40$ and $\lambda_2 = 10$

Generalized Euclidean Metric: Example

- Compute eigenvectors Φ_1 and Φ_2

$$(S - \lambda I)\underline{v} = \underline{0} \tag{38}$$

- For $\lambda_1 = 40$:

$$\left(\begin{bmatrix} 16 & -12 \\ -12 & 34 \end{bmatrix} - \begin{bmatrix} 40 & 0 \\ 0 & 40 \end{bmatrix}\right)\underline{v} = \underline{0} \tag{39}$$

$$\begin{bmatrix} 2 & 1 \\ 0 & 0 \end{bmatrix}\underline{v} = \underline{0} \tag{40}$$

$$\Phi_1 = \frac{\sqrt{5}}{5}\begin{bmatrix} 1 \\ -2 \end{bmatrix} \tag{41}$$

Generalized Euclidean Metric: Example

- Compute eigenvectors Φ_1 and Φ_2

$$(S - \lambda I)\underline{v} = \underline{0} \tag{42}$$

- For $\lambda_2 = 10$:

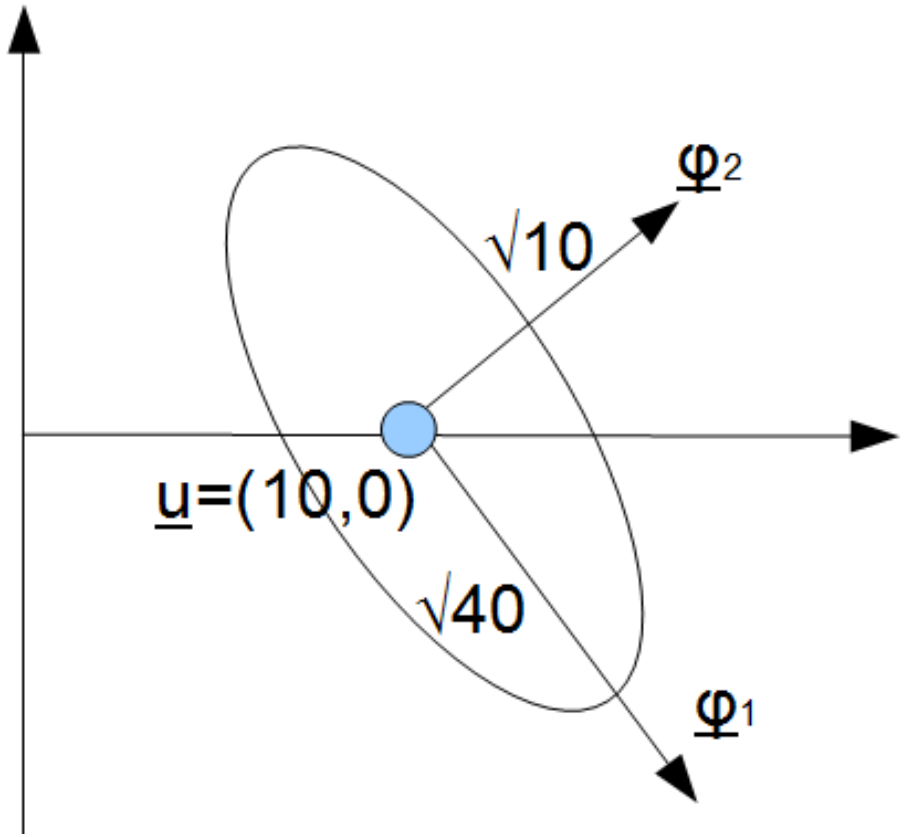
$$\left(\begin{bmatrix} 16 & -12 \\ -12 & 34 \end{bmatrix} - \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}\right)\underline{v} = \underline{0} \tag{43}$$

$$\begin{bmatrix} 1 & -2 \\ 0 & 0 \end{bmatrix}\underline{v} = \underline{0} \tag{44}$$

$$\Phi_2 = \frac{\sqrt{5}}{5}\begin{bmatrix} 2 \\ 1 \end{bmatrix} \tag{45}$$

Generalized Euclidean Metric: Example

- Sketch unit standard deviation contour in original space



Generalized Euclidean Metric: Example

- Compute orthonormal whitening transform $\Lambda^{-1/2}\Phi^T$

$$\Lambda^{-1/2}\Phi^T = \begin{bmatrix} \frac{1}{\sqrt{40}} & 0 \\ 0 & \frac{1}{\sqrt{10}} \end{bmatrix} \frac{\sqrt{5}}{5} \begin{bmatrix} 1 & -2 \\ 2 & 1 \end{bmatrix} \tag{46}$$

$$\Lambda^{-1/2}\Phi^T = \begin{bmatrix} \frac{\sqrt{2}}{20} & -\frac{\sqrt{2}}{10} \\ \frac{\sqrt{2}}{5} & \frac{\sqrt{2}}{10} \end{bmatrix} \tag{47}$$

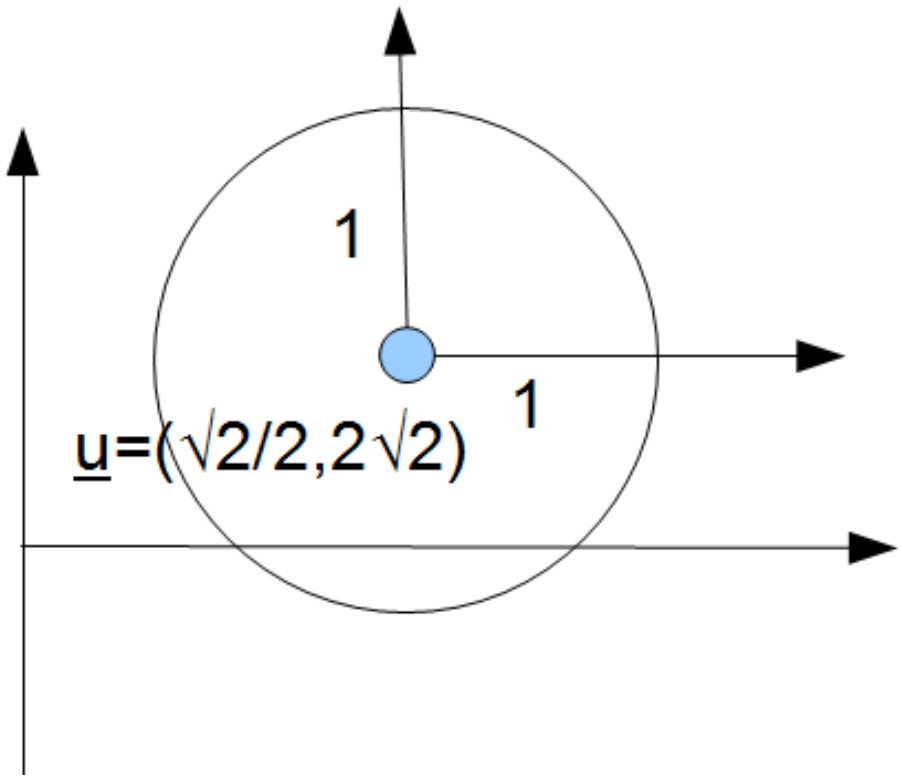
- Compute new mean m' in transformed space:

$$m' = \Lambda^{-1/2}\Phi^T m = \begin{bmatrix} \frac{\sqrt{2}}{20} & -\frac{\sqrt{2}}{10} \\ \frac{\sqrt{2}}{5} & \frac{\sqrt{2}}{10} \end{bmatrix} \begin{bmatrix} 10 \\ 0 \end{bmatrix} \tag{48}$$

$$m' = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ 2\sqrt{2} \end{bmatrix} \tag{49}$$

Generalized Euclidean Metric: Example

- Sketch unit standard deviation contour in transformed space



Minimum Intra-Class Distance (MICD) Classifier

- For classification, we want to maximize within class similarity
- In terms of distance metrics, we want to minimize **intra-class distance**
- How do we judge intra-class distance?
  - A reasonable objective measure is the **mean squared distance** within the class
- Based on the criterion of minimum mean squared distance within classes, the generalized Euclidean metric IS the minimum intra-class distance (MICD) metric!

Minimum Intra-Class Distance (MICD) Classifier

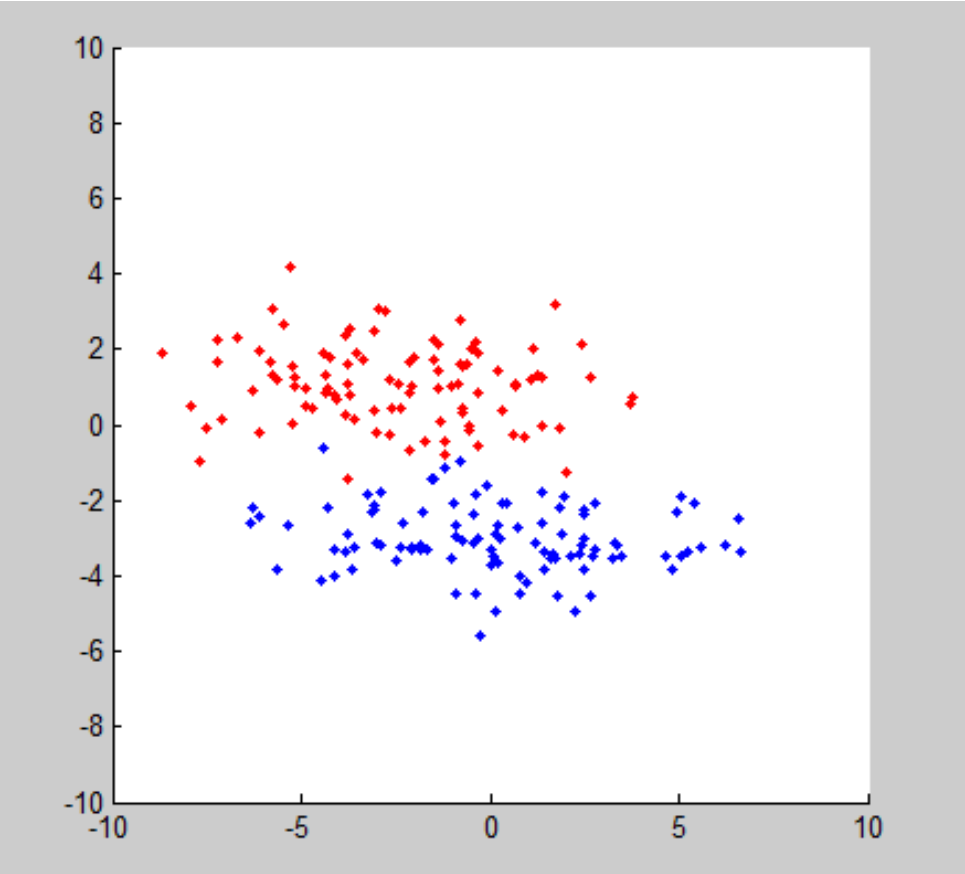
- The MICD classifier is defined by the following decision rule:

$$\underline{x} \in A \text{ iff } (\underline{x} - \underline{m}_A)^T S_A^{-1} (\underline{x} - \underline{m}_A) < (\underline{x} - \underline{m}_B)^T S_B^{-1} (\underline{x} - \underline{m}_B) \tag{50}$$

- Important observations:
  - The distance to each class is measured with its *own* metric determined by its *own* covariance matrix (e.g., $\underline{x}$ is transformed by S_A^{-1} to compute $d_{MICD}(\underline{x}, \underline{m}_A)$, while $\underline{x}$ is transformed by S_B^{-1} to compute $d_{MICD}(\underline{x}, \underline{m}_B)$)
  - Means that while a linear transformation is associated with each metric, it is not possible in general to map both hyperellipsoidal distributions to hyperspheres with **the same transformation**.

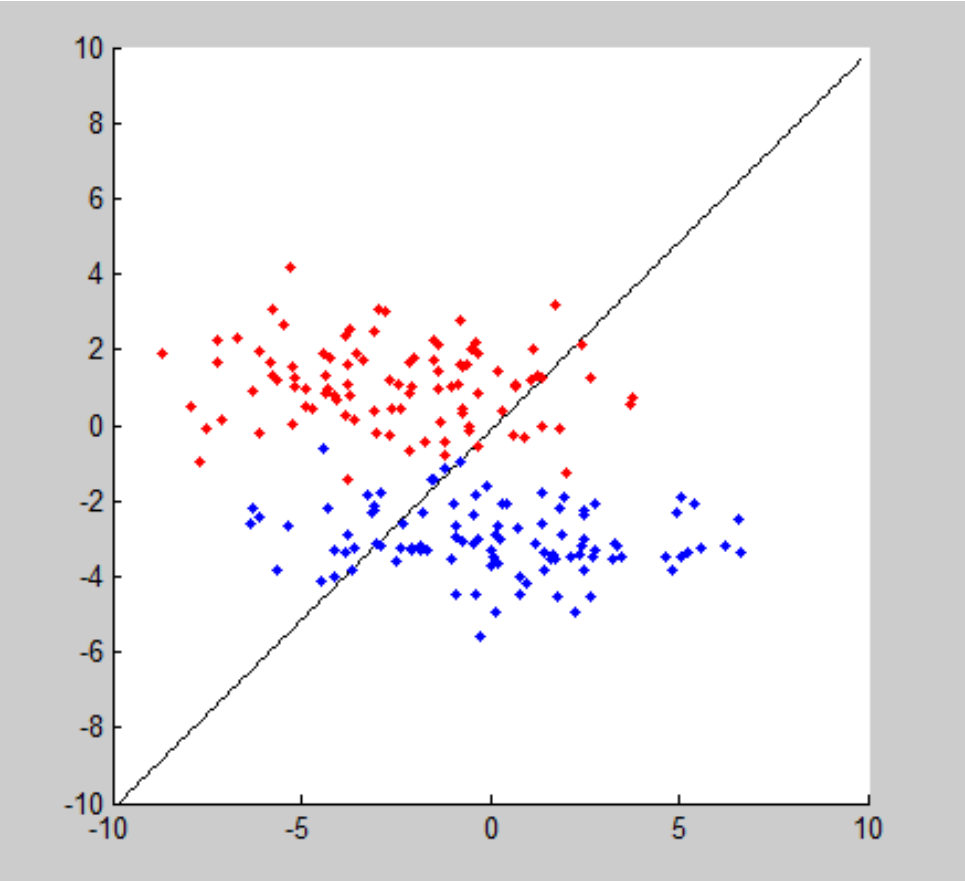
Example: Simple classification problem

- Features are Gaussian in nature, different means, uncorrelated, different variances:



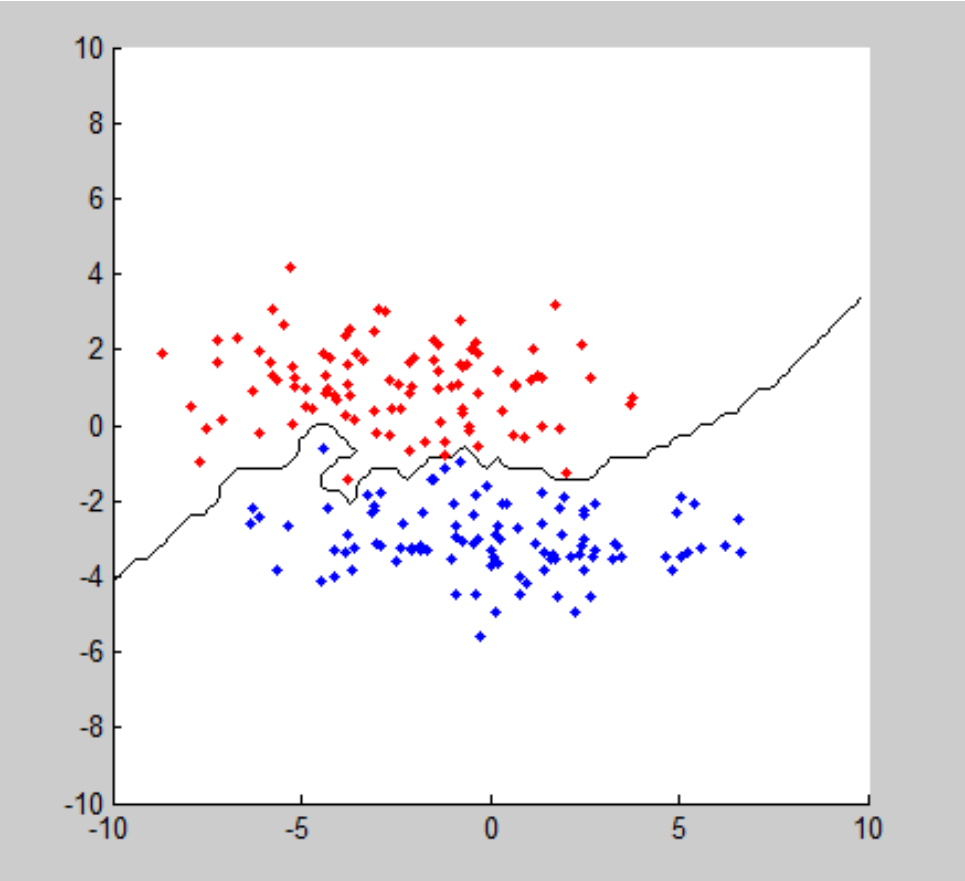
Example: Simple classification problem

- MED decision boundary:



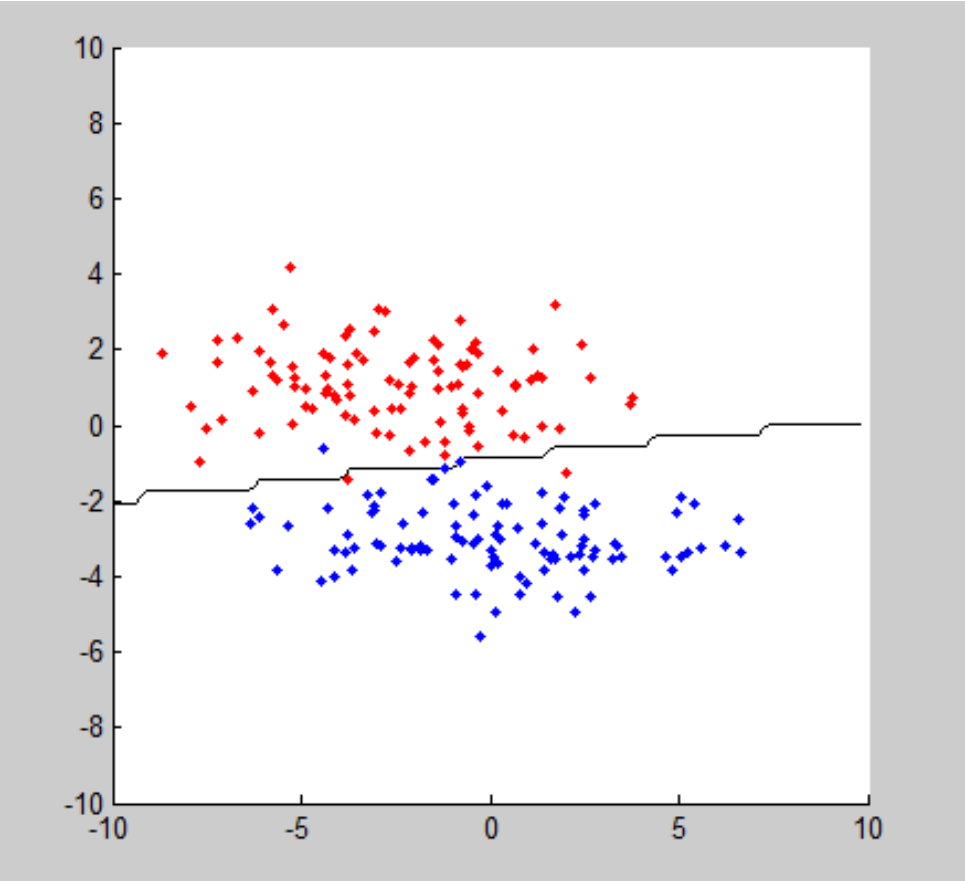
Example: Simple classification problem

- NN decision boundary:



Example: Simple classification problem

- MICD decision boundary:

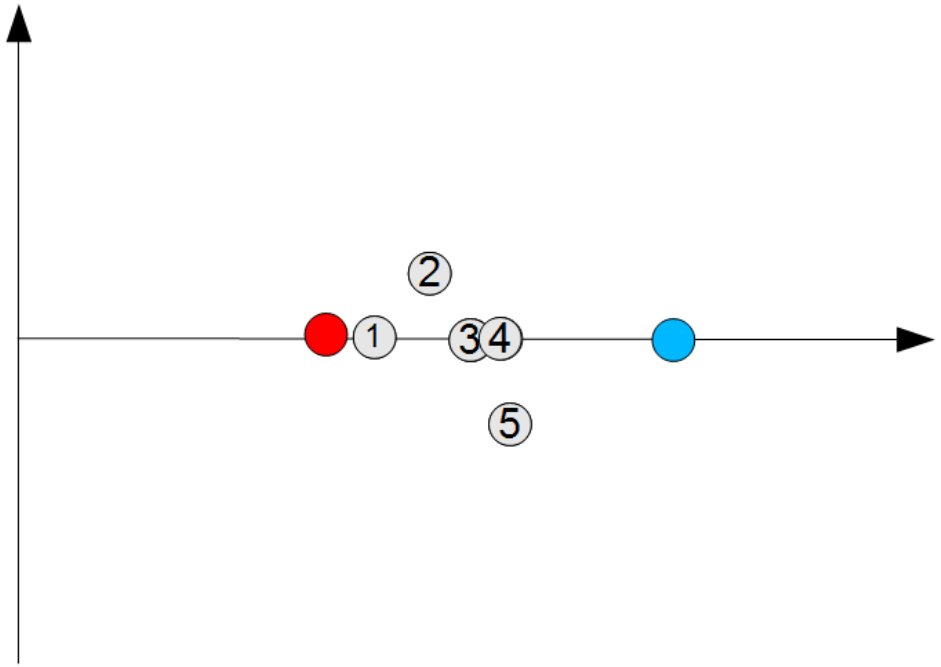


Why does MICD make sense?

- From a quick glance, there are some questions that seem to arise when trying to understand the MICD classifier:
  - Why does performing distance comparisons between a pattern and class prototypes in different transformed feature spaces make sense?
  - What distance are we really measuring when we compare a pattern and class prototype in a transformed feature space?

Why does MICD make sense?

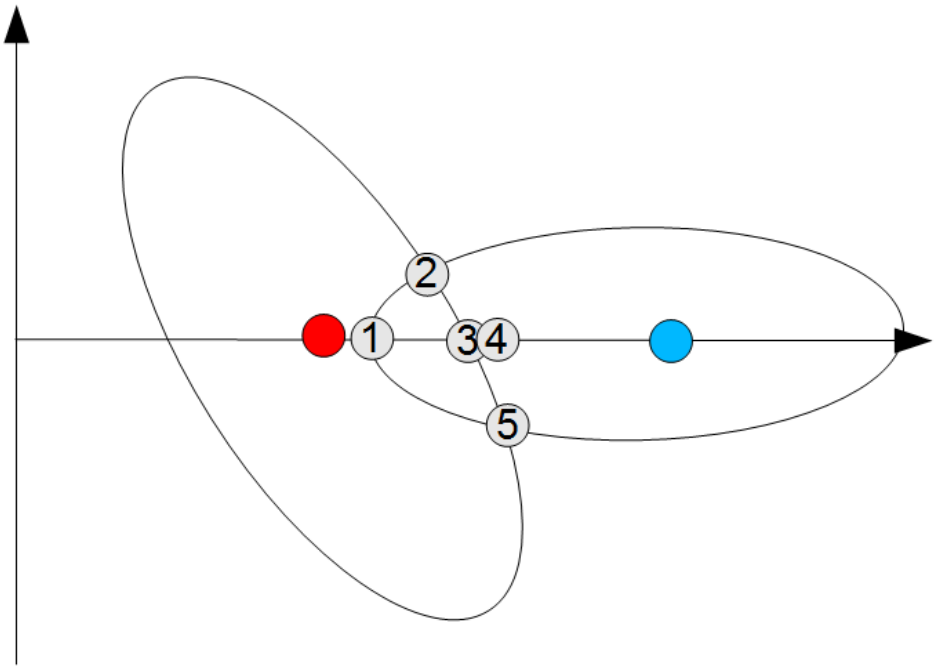
- To answer these questions, let's first take a look at a simple example



  - Based on the current units in this feature space, where does each of these patterns (in grey) belong to based on Euclidean distance?

Why does MICD make sense?

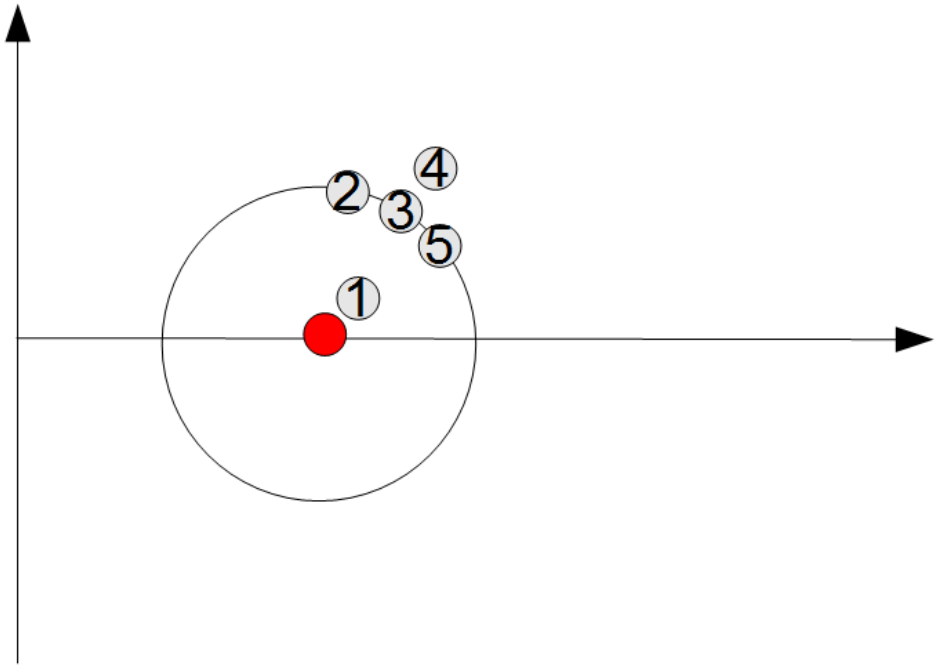
- Suppose that I now tell you that the classes have the following statistical distribution



  - Based on this new information, where should they belong to now?

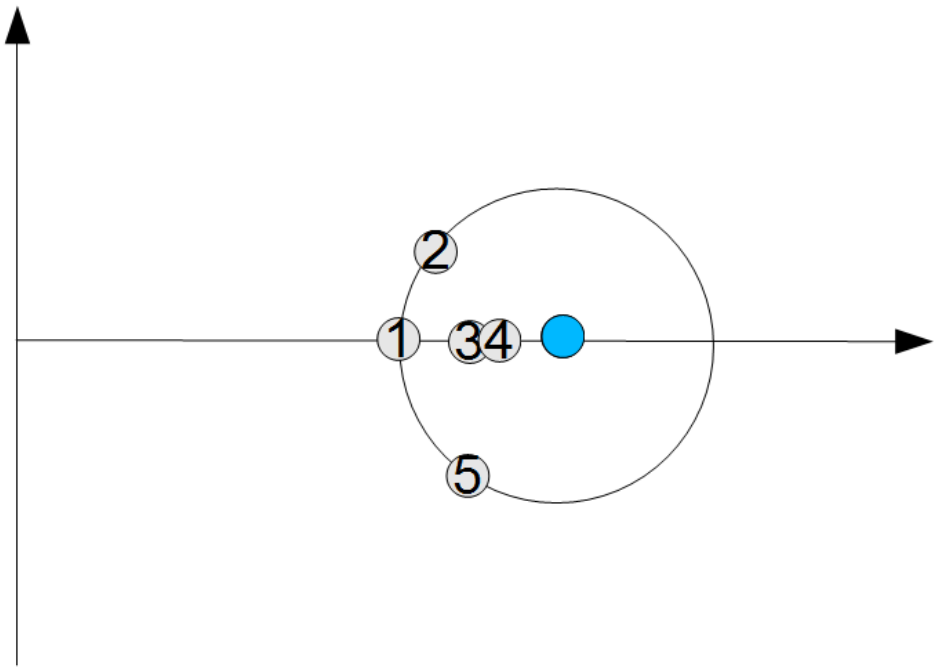
Why does MICD make sense?

- Let's transform the patterns relative to the statistics of class A:



Why does MICD make sense?

- Let's transform the patterns relative to the statistics of class B:



Why does MICD make sense?

- Looking at the features in the transformed spaces for class A and class B, we can now answer the question: what distance are we really measuring when we compare a pattern and class prototype in a transformed feature space?
  - To get into the transformed feature space for each class, we normalize the feature axes by its **standard deviation** for each feature.
  - What that means is that the new unit of measure in the transformed space is now in terms of the number of standard deviations.

Why does MICD make sense?

- Knowing that the distance between a pattern and the prototype of a class is now measured in units of standard deviation, we can now see how doing distance comparisons between a pattern and class prototypes make sense by taking a look at each of the patterns.
  - In Euclidean space, pattern 3 is three units from A and four units from B, so it is classified as A.
  - In the transform spaces, pattern 3 is one standard deviation from A and less than one standard deviation from B, so it is classified as B.
  - In Euclidean space, pattern 4 is 3.5 units from A and 3.5 units from B, so it can be A or B.
  - In the transform spaces, pattern 4 is greater than one standard deviation from A and less than one standard deviation from B, so it is classified as B.

MICD Decision Boundaries

- Unlike the Euclidean distance classifier, the decision boundaries between class regions are in general **NOT** linear.
- So how do we determine the decision boundaries?
  - Recall that patterns are equally similar (e.g., same distance) to both classes
  - Therefore, the decision boundaries lie on the intersections of the corresponding equidistance contours around the classes
  - The equidistance contour for a class A can be defined as:

$$(\underline{x} - \underline{m}_A)^T S_A^{-1} (\underline{x} - \underline{m}_A) = c \tag{51}$$

MICD Decision Boundaries

- So how do we determine the decision boundaries?
  - Therefore, the intersections of the corresponding equidistance contours around the classes is

$$(\underline{x} - \underline{m}_A)^T S_A^{-1} (\underline{x} - \underline{m}_A) = (\underline{x} - \underline{m}_B)^T S_B^{-1} (\underline{x} - \underline{m}_B) \quad (52)$$

  - Expanding and simplifying leads to the general quadratic surface in hyperspace:

$$\underline{x}^T Q_0 \underline{x} + Q_1 \underline{x} + Q_2 = 0, \quad (53)$$

where,

$$Q_0 = S_A^{-1} - S_B^{-1} \quad (54)$$

$$Q_1 = 2[\underline{m}_B^T S_B^{-1} - \underline{m}_A^T S_A^{-1}] \quad (55)$$

$$Q_2 = \underline{m}_A^T S_A^{-1} \underline{m}_A - \underline{m}_B^T S_B^{-1} \underline{m}_B \quad (56)$$

MICD Classifier: Example

- Suppose we are given the following statistical information about the classes:
  - Class 1: $\underline{m}_1 = [3 \;\; 3]^T, S_1 = \begin{bmatrix} 3 & -2 \\ -2 & 1 \end{bmatrix}$.
  - Class 2: $\underline{m}_2 = [4 \;\; 5]^T, S_2 = \begin{bmatrix} 1 & -2 \\ -2 & 3 \end{bmatrix}$.
- Suppose we wish to build a MICD classifier using sample means as prototypes.
  - Compute the discriminate function for each class.
  - Compute the decision boundary.

MICD Classifier: Example

- Step 1: Find discriminant functions for each class based on MICD decision rule:
- Recall that the MICD decision criteria for the two class case is:

$$d_{MICD}(\underline{x}, \underline{z}_1) < d_{MICD}(\underline{x}, \underline{z}_2) \tag{57}$$

$$[(\underline{x}-\underline{z}_1)^T {S_1}^{-1}(\underline{x}-\underline{z}_1)]^{1/2} < [(\underline{x}-\underline{z}_2)^T {S_2}^{-1}(\underline{x}-\underline{z}_2)]^{1/2} \tag{58}$$

$$(\underline{x}-\underline{z}_1)^T {S_1}^{-1}(\underline{x}-\underline{z}_1) < (\underline{x}-\underline{z}_2)^T {S_2}^{-1}(\underline{x}-\underline{z}_2) \tag{59}$$

- Compute ${S_1}^{-1}$ and ${S_2}^{-2}$:

$${S_1}^{-1} = \begin{bmatrix} -1 & -2 \\ -2 & -3 \end{bmatrix} \quad {S_2}^{-1} = \begin{bmatrix} -3 & -2 \\ -2 & -1 \end{bmatrix} \tag{60}$$

MICD Classifier: Example

- Plugging in $z_1 = m_1$, $z_2 = m_2$, ${S_1}^{-1}$, and ${S_2}^{-1}$ gives us:

$$(\underline{x} - \underline{z}_1)^T {S_1}^{-1} (\underline{x} - \underline{z}_1) < (\underline{x} - \underline{z}_2)^T {S_2}^{-1} (\underline{x} - \underline{z}_2) \tag{61}$$

$$\begin{aligned} &([x_1 \ x_2]^T - [3 \ 3]^T)^T \begin{bmatrix} -1 & -2 \\ -2 & -3 \end{bmatrix} ([x_1 \ x_2]^T - [3 \ 3]^T) \\ &< ([x_1 \ x_2]^T - [4 \ 5]^T)^T \begin{bmatrix} -3 & -2 \\ -2 & -1 \end{bmatrix} ([x_1 \ x_2]^T - [4 \ 5]^T) \end{aligned} \tag{62}$$

$$\begin{aligned} &([x_1 - 3 \ \ x_2 - 3]) \begin{bmatrix} -1 & -2 \\ -2 & -3 \end{bmatrix} ([x_1 - 3 \ \ x_2 - 3])^T \\ &< ([x_1 - 4 \ \ x_2 - 5]) \begin{bmatrix} -3 & -2 \\ -2 & -1 \end{bmatrix} ([x_1 - 4 \ \ x_2 - 5])^T \end{aligned} \tag{63}$$

MICD Classifier: Example

- Plugging in $z_1 = m_1$, $z_2 = m_2$, ${S_1}^{-1}$, and ${S_2}^{-1}$ gives us:

$$\begin{aligned}&([x_1 - 3 \;\; x_2 - 3]) \begin{bmatrix} -1 & -2 \\ -2 & -3 \end{bmatrix} ([x_1 - 3 \;\; x_2 - 3])^T \\ &< ([x_1 - 4 \;\; x_2 - 5]) \begin{bmatrix} -3 & -2 \\ -2 & -1 \end{bmatrix} ([x_1 - 4 \;\; x_2 - 5])^T \end{aligned} \tag{64}$$

$$\begin{aligned}&([(-x_1 - 2x_2 + 9) \;\; (-2x_1 - 3x_2 + 15)])([x_1 - 3 \;\; x_2 - 3])^T \\ &< ([(-3x_1 - 2x_2 + 22) \;\; (-2x_1 - x_2 + 13)])([x_1 - 4 \;\; x_2 - 5])^T \end{aligned} \tag{65}$$

MICD Classifier: Example

- Expanding and simplifying gives us:

$$-x_1^2 - 3x_2^2 + 18x_1 + 15x_2 - 72 < -3x_1^2 + x_2^2 + 44x_1 + 26x_2 - 23 \tag{66}$$

- Therefore, the discriminant functions are:

$$g_1(x_1, x_2) = -x_1^2 - 3x_2^2 + 18x_1 + 15x_2 - 72 \tag{67}$$

$$g_2(x_1, x_2) = -3x_1^2 + x_2^2 + 44x_1 + 26x_2 - 23 \tag{68}$$

MICD Classifier: Decision Boundary

- Step 2: Find decision boundary between classes 1 and 2
- For MICD classifier, the decision boundary is

$$g(x_1, x_2) = g_1(x_1, x_2) - g_2(x_1, x_2) = 0. \tag{69}$$

Plugging in the discriminant functions g_1 and g_2 gives us:

$$\begin{aligned} &g(x_1, x_2) = -x_1^2 - 3x_2^2 + 18x_1 + 15x_2 - 72 \\ &-(-3x_1^2 + x_2^2 + 44x_1 + 26x_2 - 23) = 0 \end{aligned} \tag{70}$$

MICD Classifier: Decision Boundary

- Grouping terms:

$$g(x_1, x_2) = 2x_1^2 - 4x_2^2 - 26x_1 - 11x_2 - 49 = 0 \qquad (71)$$

- Therefore, the decision boundary is a quadratic!

MICD Decision Boundaries

- Case 1: different means, equal covariance ($a = b$, $S_A = S_B = S$)
- The parameters become:

$$Q_0 = S_A^{-1} - S_B^{-1} = S^{-1} - S^{-1} = 0 \tag{72}$$

$$Q_1 = 2[\underline{m}_B^T S_B^{-1} - \underline{m}_A^T S_A^{-1}] = 2[\underline{m}_B^T - \underline{m}_A]^T S^{-1} \tag{73}$$

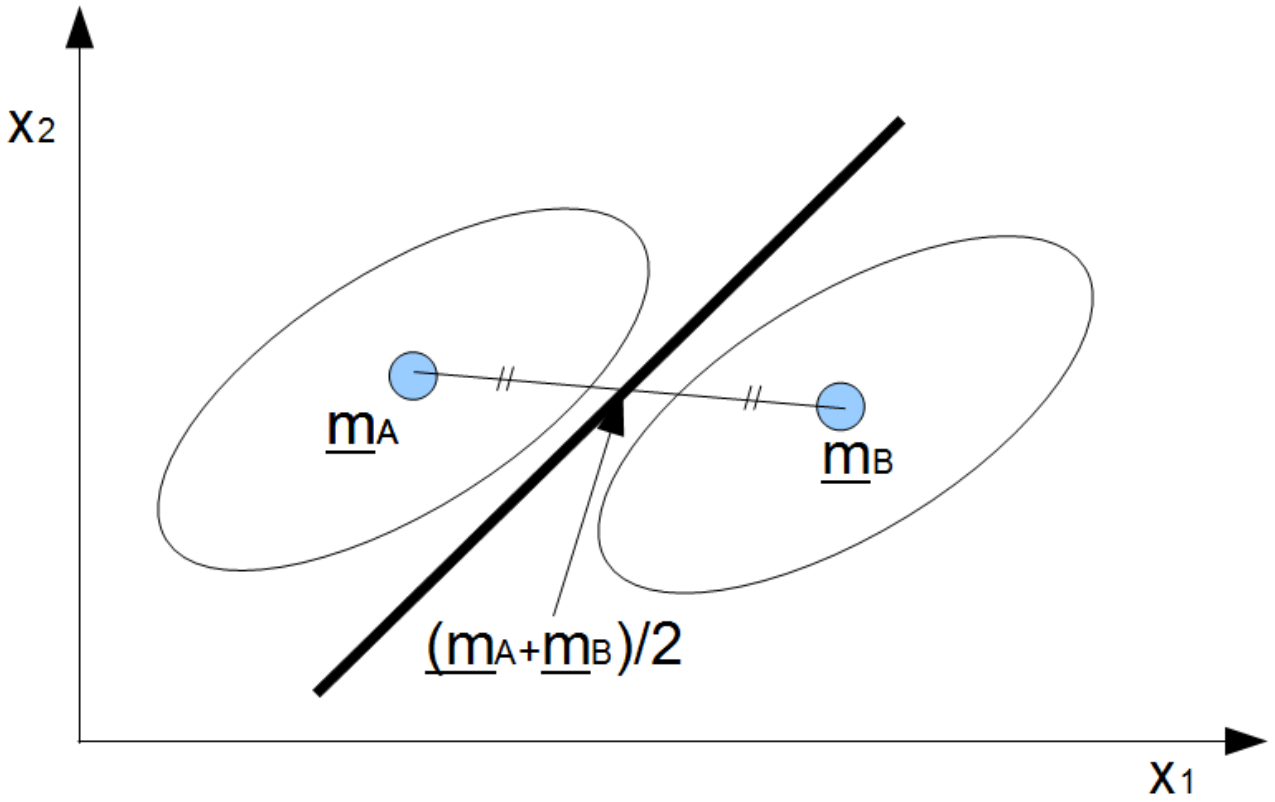
$$Q_2 = \underline{m}_A^T S^{-1} \underline{m}_A - \underline{m}_B^T S^{-1} \underline{m}_B = (\underline{m}_A - \underline{m}_B)^T S^{-1} (\underline{m}_A - \underline{m}_B) \tag{74}$$

- This gives us the final decision boundary:

$$(\underline{m}_A - \underline{m}_B)^T S^{-1} [\underline{x} - (\underline{m}_A + \underline{m}_B)/2] = 0 \tag{75}$$

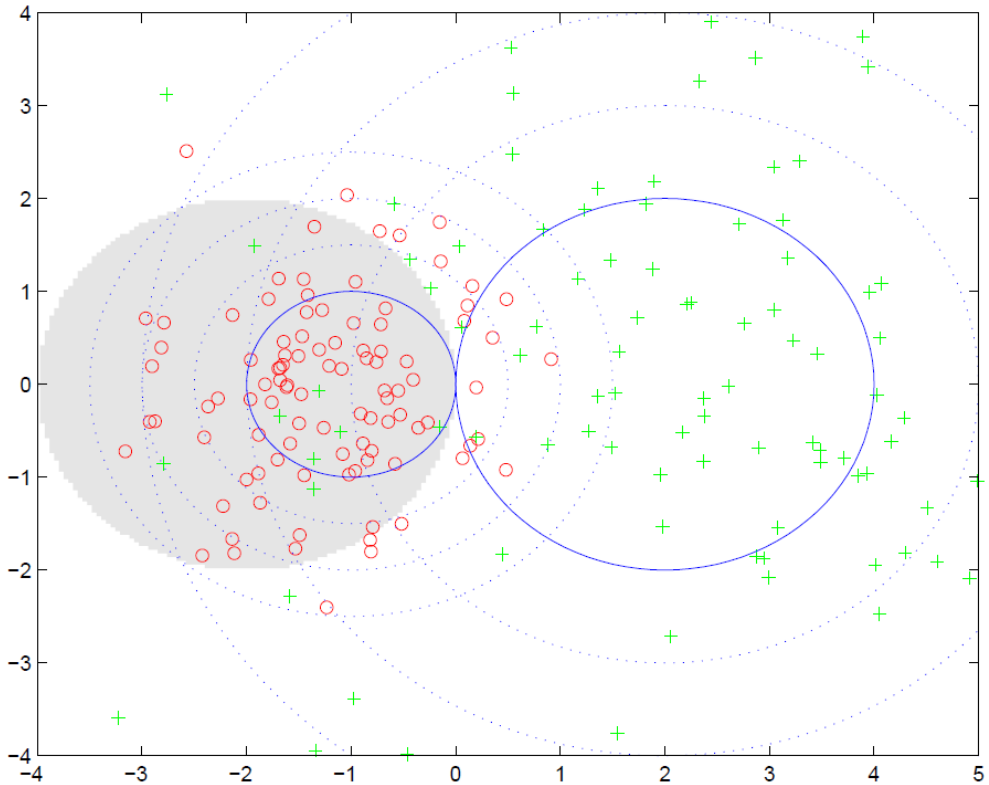
- This is just a straight line through $(\underline{m}_A + \underline{m}_B)/2$ (i.e., midpoint between the means), with a slope that's influenced by S.

MICD Decision Boundaries



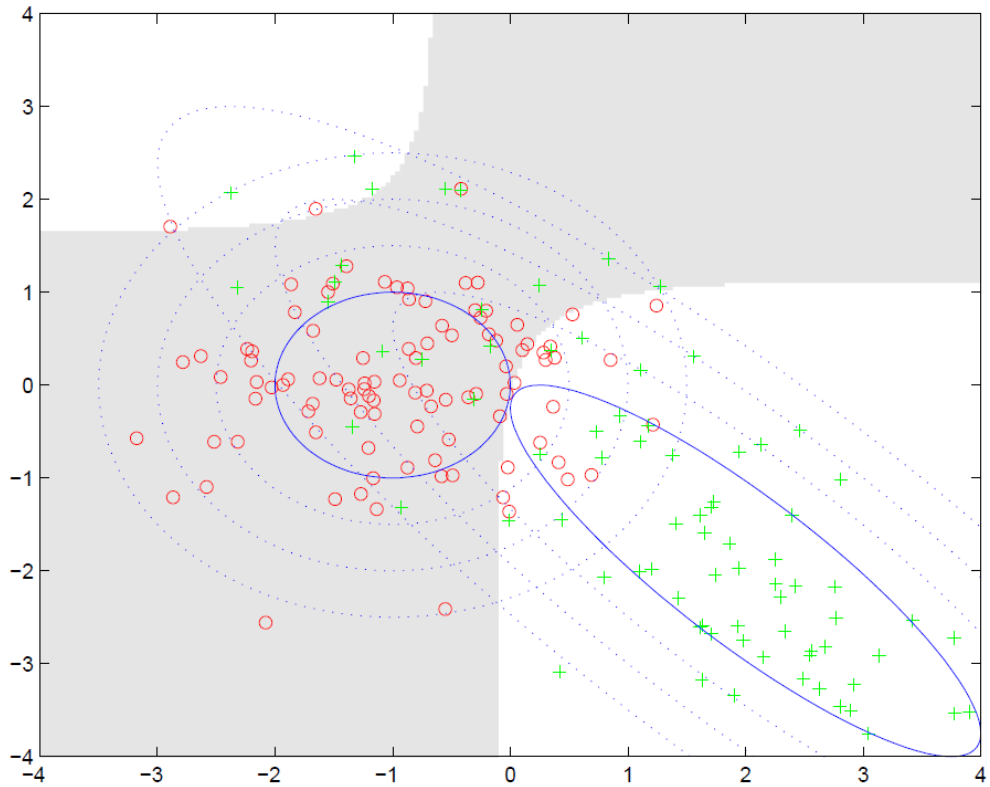
MICD Decision Boundaries

- Case 2: different means, different variances, uncorrelated features
- Example:



MICD Decision Boundaries

- Case 3: different means, different variances, correlated features
- Example:



MICD Decision Boundaries

- Case 4: same means, different covariance ($\underline{m}_a = \underline{m}_b = \underline{m}$, $S_A \neq S_B$)
- The parameters become:

$$Q_0 = S_A^{-1} - S_B^{-1} \tag{76}$$

$$Q_1 = 2[\underline{m}_B^T S_B^{-1} - \underline{m}_A^T S_A^{-1}] = -2\underline{m}^T[S_A^{-1} - S_B^{-1}] \tag{77}$$

$$Q_2 = \underline{m}_A^T S^{-1} \underline{m}_A - \underline{m}_B^T S^{-1} \underline{m}_B = \underline{m}^T (S_A^{-1} - S_B^{-1})\underline{m} \tag{78}$$

- This gives us the final decision boundary:

$$(\underline{x} - \underline{m})^T (S_A^{-1} - S_B^{-1})(\underline{x} - \underline{m}) = 0 \tag{79}$$

- This surface is complicated in general, but can be visualized more easily on some special cases.

MICD Decision Boundaries

- Example: Suppose that we have only one feature ($n = 1$) and $\underline{m} = 0$
- Given the MICD decision boundary for Case 4:

$$(\underline{x} - \underline{m})^T (S_A^{-1} - S_B^{-1})(\underline{x} - \underline{m}) = 0 \tag{80}$$

- We get the following:

$$(1/s_A^2 - 1/s_B^2)x^2 = 0 \tag{81}$$

- The only solution for this equation is $x = 0$!

MICD Decision Boundaries

- Therefore, given the MICD classification rule:

$$(1/s_A^2)x^2 < (1/s_B^2)x^2 \tag{82}$$

- the x's cancel out,

$$1/s_A^2 < 1/s_B^2 \tag{83}$$

$$s_A^2 > s_B^2 \tag{84}$$

- The MICD classification rule decides in favor of the class with the largest variance, regardless of x!

MICD Decision Boundaries

- Why does this happen?
  - When applying MICD, distance is really measured in units of standard deviation.
  - Therefore, any unknown x is always closer to $m = 0$ with the metric for the class with largest variance (the transform that scales that cluster's class to unit standard deviation is larger, thus pulling any unknown x closer)
- However, given a Gaussian distribution, the class with higher standard deviation would have a lower probability near $m = 0$.
- Therefore, MICD is sub-optimal in this case!

MICD Decision Boundaries

- Example in $n = 2$: same means, different covariance ($\underline{m}_a = \underline{m}_b = \underline{m}$, $S_A \neq S_B$)
- Mean: $\underline{m}_a = \underline{m}_b = \underline{m} = [0 \;\; 0]^T$
- Covariances:

$$S_A = \begin{bmatrix} 1/2 & 0 \\ 0 & 1 \end{bmatrix} \quad S_B = \begin{bmatrix} 1 & 0 \\ 0 & 1/2 \end{bmatrix} \tag{85}$$

- Recall that the MICD decision boundary for this case is:

$$(\underline{x} - \underline{m})^T (S_A^{-1} - S_B^{-1})(\underline{x} - \underline{m}) = 0 \tag{86}$$

MICD Decision Boundaries

- Plugging in the mean and covariance terms give us:

$$(\underline{x})^T(\begin{bmatrix} 1/2 & 0 \\ 0 & 1 \end{bmatrix}^{-1} - \begin{bmatrix} 1 & 0 \\ 0 & 1/2 \end{bmatrix}^{-1})(\underline{x}) = 0 \tag{87}$$

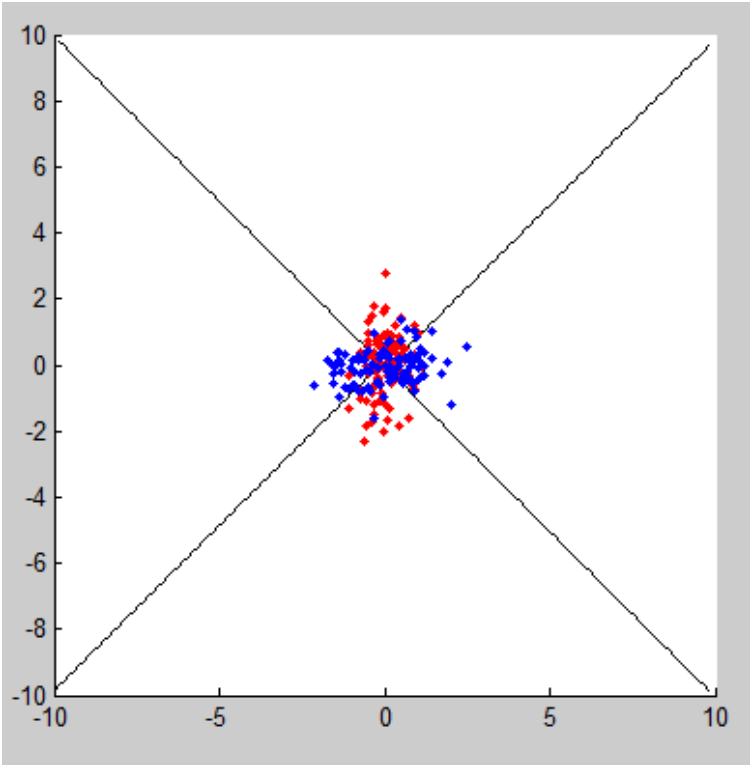
$$[x_1 \;\; x_2](\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix})[x_1 \;\; x_2]^T = 0 \tag{88}$$

$$x_1^2 - x_2^2 = 0 \tag{89}$$

- The solution to this equation is $x_1 = \pm x_2$.

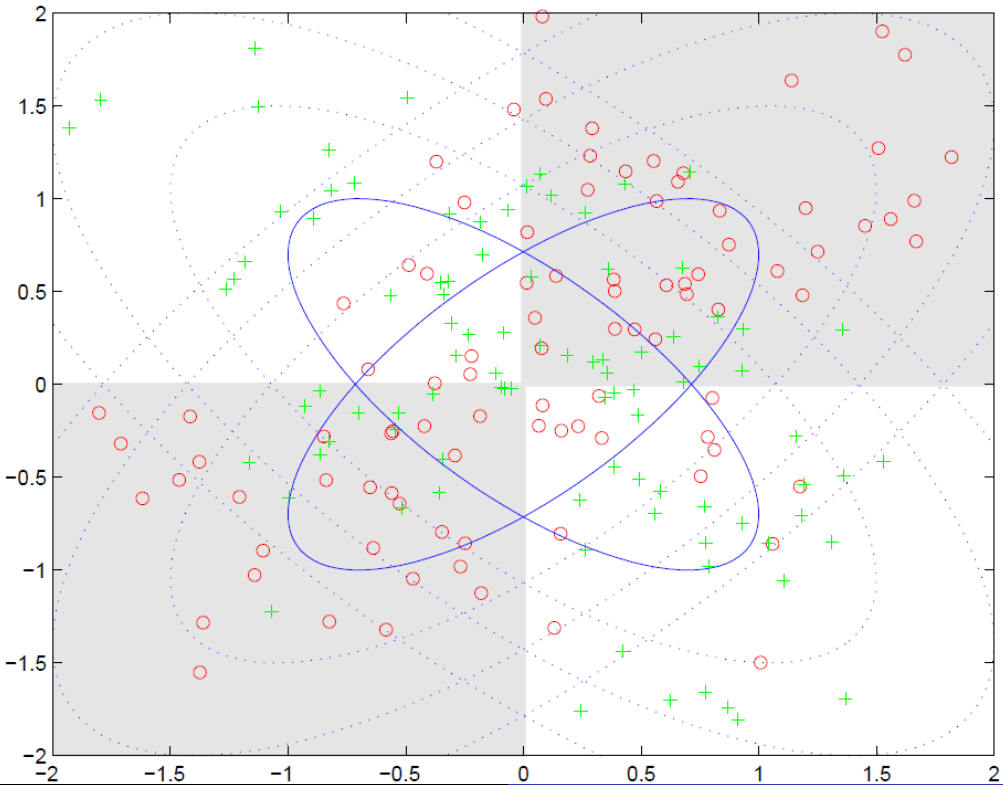
MICD Decision Boundaries

- Plotting the decision boundary gives us:



MICD Decision Boundaries

- Example: same means, different variances, correlated features
- Example:



MICD Classifier: advantages and disadvantages

- Advantages: lower sensitivity to noise and outliers, great for handling class distributions that are well modeled by Gaussian models (e.g., mean and variance)
- Disadvantages: poor at handling more complex class distributions
- Need to develop more power classifiers based on more complete information about the probabilistic behaviour of the classes.

Where can I use any of this?

- One question about all of this is: where can I apply concepts like orthonormal whitening, Generalized Euclidean distances, and etc?
- Here's one interesting application of these concepts: motion tracking for video games! (e.g., Microsoft Kinect)

Motion tracking

- When work on motion tracking for Kinect first started, they investigated developing a motion tracking framework by
  1. Creating an avatar
  2. Move avatar to match images of player as he or she moves
- Problems:
  - Started losing track of player after a short period of time
  - Only tracked players roughly the same size and shape as avatar
  - Couldn't process fast movements of the player

Ref: A. Bogdanowicz, "The Motion Tech Behind Kinect", The Institute, January 6, 2011.

Motion tracking

- How do you solve this problem?
- Solution: let's use a radically different approach.
- Instead of trying to use a fixed avatar model to match images of the player, let's instead figure out where the individual parts of the body is!
- Advantages:
  - No longer rely on the shape or size of avatar or person at all.
  - If we know where the individual parts of the body is, then we can just move the corresponding part of the avatar! (e.g., the player can be a basketball player moving a stumpy Hobbit around)
  - Since classification is done pixel-by-pixel on each frame, it doesn't lose track of the player!

Motion tracking

- New question: How do we know where each part of the body is in the image?
- Solution: Let's classify each pixel in the image, based on its characteristics, as a body part or as the background.
- For example, if we had n body parts, then maybe we want to classify pixels as one of $n + 1$ classes.
- Train system using people of different sizes and shapes under different poses, with corresponding class labels, so it can learn how to recognize body parts

Image Classification

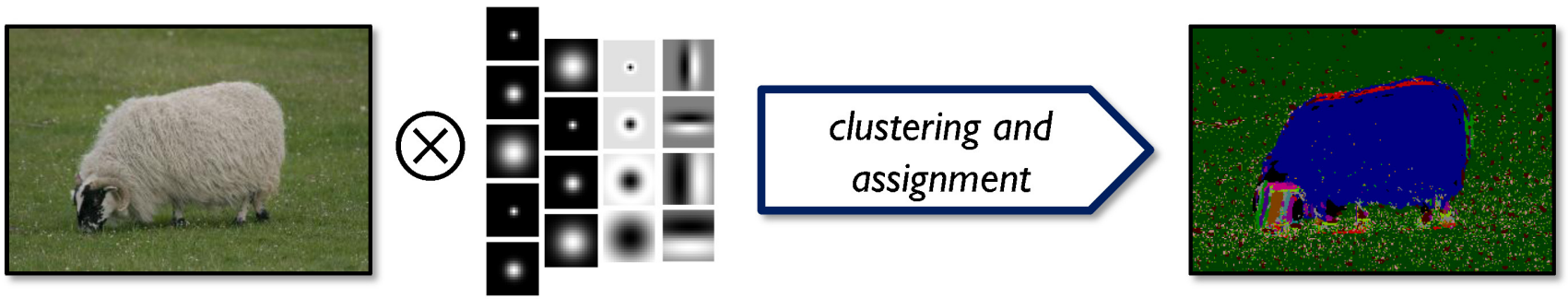
- What types of features may be appropriate for learning these individual classes from an image?
- One possible set of features within the full feature set is **textons**
  - Training images are filtered using a m-dimensional filter bank to get m responses at each pixel
  - These responses characterize the texture characteristics of a pixel based on it's surrounding pixels, and represents that pixel's *pattern*.
- Given all these patterns, what we want to do is learn the corresponding classes.

Image Classification

- But how do we learn these classes?
- Solution: Let's cluster them together to determine the prototype for each class!
- However, you may not get good delineation between classes, with one possible reason being the fact that the features are very likely to have different variances (means that Euclidean distance metric is not a good choice)
- From what we have learned so far, an effective way to solve this is to whiten the features so we have unit variant features!

Image Classification

- Once we have the cluster centers, and we have unit variant features, we can now classify each pixel based on its pattern's Euclidean distance to the cluster centers!



Ref: J. Shotton et al., "TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context", IJCV, January 2009.