Intelligent Systems: Reasoning and Recognition

James L. Crowley

ENSIMAG 2 / MoSIG M1 Second Semester 2013/2014

Lesson 18 23 April 2014

Linear Classifiers as Pattern Detectors

Contents

Sources Bibliographiques :

"Pattern Recognition and Machine Learning", C. M. Bishop, Springer Verlag, 2006.

Notation

x	a variable
X	a random variable (unpredictable value)
$\vec{x}$	A vector of D variables.
$\vec{X}$	A vector of D random variables.
D	The number of dimensions for the vector $\vec{x}$ or $\vec{X}$
E	An observation. An event.
C_k	The class (tribe) k
k	Class index
K	Total number of classes
ω_k	The fact that E $\in C_k$
$\hat{\omega}_k$	The decision (estimation) that E $\in C_k$
$p(\omega_k) = p(E \in C_k)$	Probability that the observation E is a member of the class k.
M_k	Number of examples for the class k. (think M = Mass)
M	Total number of examples. $M = \sum_{k=1}^{K} M_k$
$\{X_m\}$ $\{y_m\}$	M Training samples labeled with an indicator variable.

Two class indicator: $y_m = +1$ for target class and $y_m = -1$ for other

Indicators for K classes $\vec{y}_m{}^T = \left(y_m^1 \quad \ldots \quad y_m^K\right)$

where $y_m^k = \begin{cases} 1 \text{ if sample m is class k} \\ 0 \qquad\qquad \text{otherwise} \end{cases}$

K class indicator, soft margin: $\vec{y}_m{}^T = \left(y_m^1 \quad \ldots \quad y_m^K\right)$

where y_m^k is the likelihood that sample m is class k (as with EM).

Linear Classifiers as Pattern Detectors

Linear classifiers are widely used to define pattern "detectors". This is used in computer vision, for example to detect faces, road signs, publicity logos, or other patterns of interest.

In the case of pattern detectors, K=2.

Class k=1: The target pattern.
Class k=2: Everything else.

The detector is learned form a set of training data training composed of M sample observations $\{\vec{X}_m\}$ where each sample observation is labeled with an indicator variable

$y_m = +1$ for examples of the target pattern (class 1)
$y_m = -1$ for all other examples.

Our goal is to build a linear classifier (a hyper-plane) that provides a best separation of class 1 from class 2.

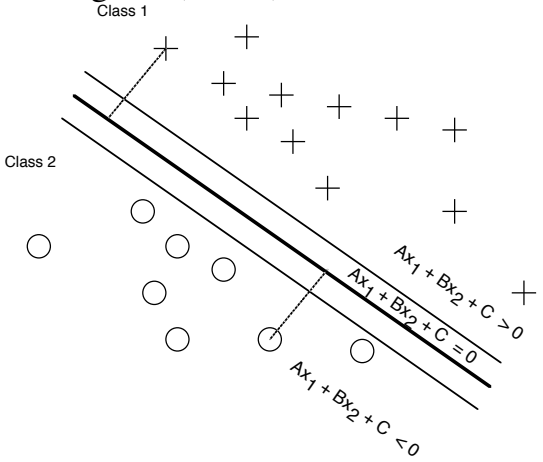
A hyperplane is a set of points such that

$$w_1 x_1 + w_2 x_2 + ... + w_D x_D + B = 0$$

This can be written: $\vec{W}^T \vec{X} + B = 0$

The decision rule is IF $\vec{W}^T \vec{X} + B > 0$ THEN $E \in C_1$ else $E \notin C_1$

We can use B as an adjustable gain (a bias) that sets the sensitivity of the detector.



Note that $\vec{W} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_D \end{pmatrix}$ is the normal to the hyperplane.

When $\vec{W}$ is normalized to unit length, $\|\vec{W}\| = 1$, then

$B = -\vec{W}^T\vec{X}$ is the perpendicular distance to the origin for a hyperplane that includes the point $\vec{X}$

if $\|\vec{W}\| \neq 1$ then we can normalize with $\vec{W}' = \frac{\vec{W}}{\|\vec{W}\|}$ and $B' = \frac{B}{\|\vec{W}\|}$

A variety of techniques exist to calculate the plane. The best choice can depend on the nature of the pattern class as well as the nature of the non-class data.

ROC Curves

Two-class linear classifiers are practical for many problems. Among other uses, they provide the optimal solution to many signal detection problems in communications theories. In the case of radio communications, the noise is typically additive, Gaussian and independent of the signal, and the Bayesian Classifier reduces to a linear classifier.

Historically two class linear classifiers have been used to demonstrate optimality for some signal detection methods. The quality metric that is used is the Receiver Operating Characteristic curve. This curve should be used to describe or compare any method for signal or pattern detection.

$$g(\vec{X}) = \vec{W}^T \cdot \vec{X} + B$$

B is a Bias term that can be swept through a range of values.
We can bias the classifier to one or the other class by adjusting B.
Changing B changes the ratio of true positive detection to false detections.
This is illustrated by the Receiver Operating Characteristics (ROC) curve.

The ROC plots True Positive Rate (TPR) against False Positive Rate (FNR) as a function of B for the training data $\{\vec{X}_m\}$, $\{y_m\}$.

Les us define a detection as either Positive (P) or Negative (N)

IF $\vec{W}^T\vec{X}_m + B > 0$ THEN P else N

The detection can be TRUE (T) or FALSE (F) depending on the indicator y_m

IF $y_m \cdot (\vec{W}^T\vec{X}_m + B) > 0$ THEN T else F

Combining these two values, any detection can be a True Positive (TP), False Positive (FP), True Negative (TN) or False Negative (FN).

For the M samples of the training data $\{\vec{X}_m\}$, $\{y_m\}$ let us define:
#P as the number of Positives,
#N as the number of Negatives,
#T as the number of True and
#F as the number of False,

From this we can define

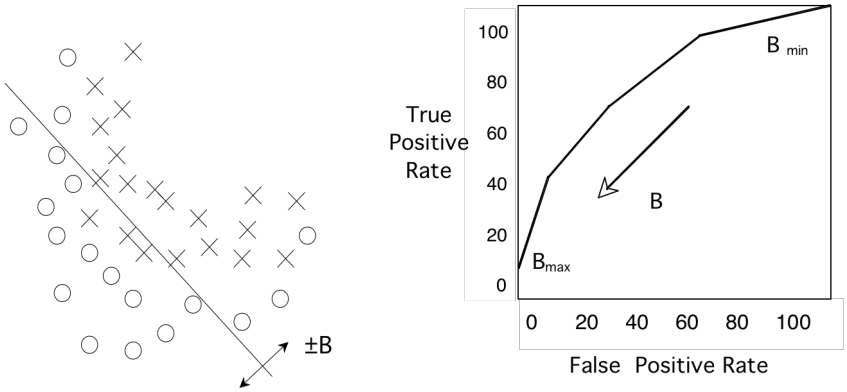
#TP as the number of True Positives,
#FP as the number of False Positives,
#TN as the number of True Negative,
#FN as the number of False Negatives.

Note that #P = #TP + #FN
And #N = #FP+ #TN

The True Positive Rate (TPR) is $TPR = \frac{\#TP}{\#P} = \frac{\#TP}{\#TP+\#FN}$

The False Positive Rate (FPR) is $FPR = \frac{\#FP}{\#N} = \frac{\#FP}{\#FP+\#TN}$

The ROC plots the TPR against the FPR as B is swept through a range of values.



When B is large, all the samples are detected as N, and both the TPR and FPR are 0. As B decreases both the TPR and FPR increase. Normally TPR is larger than FPR for any B. If TPR and FPR are equal, then the detector is no better than chance.

The more the curve approaches the upper left corner the better the detector.
The ROC is a powerful descriptor for the "goodness" of a linear classifier.
For a target class C_1 a Positive (P) detection is the decision that $E \in C_1$
a Negative (N) detection is the decision that $E \in C_2$

		$y_m \cdot (\vec{W}^T \vec{X}_m + B) > 0$	
$\vec{W}^T \vec{X}_m + B > 0$		T	F
	P	True Positive (TP)	False Positive (FP)
	N	False Negative (FN)	True Negative (TN)

Linear Discriminant Functions

In lesson 17 we saw that the classification function in a Bayesian Classifier can be decomposed into two parts: a decision function – d() and a discrimination function – $g_k()$:

$$\hat{\omega}_k = d(\vec{g}(\vec{X}))$$

Quadratic discrimination functions can be derived directly from maximizing the probability of $p(\omega_k \mid X)$

$$\vec{g}(\vec{X}) = \begin{pmatrix} g_1(\vec{X}) \\ g_2(\vec{X}) \\ \dots \\ g_K(\vec{X}) \end{pmatrix}$$ A set of discriminate functions : $R^D \rightarrow R^K$

d() : a decision function $R^K \rightarrow \{\omega_K\}$

We derived the canonical form for the discriminate function.

$$g_k(\vec{X}) = \vec{X}^T D_k \vec{X} + \vec{W}_k^T \vec{X} + b_k$$

where: $D_k = -\frac{1}{2}\Sigma_k^{-1}$

$\vec{W}_k = -2\Sigma_k^{-1}\vec{\mu}_k$

and $b_k = -\frac{1}{2}\vec{\mu}_k{}^T\Sigma_k^{-1}\vec{\mu}_k - Log\{\det(\Sigma_k)\} + Log\{p(\omega_k)\}$

A set of K discrimination functions $g_k(\vec{X})$ partitions the space $\vec{X}$ into a disjoint set of regions with quadratic boundaries. At the boundaries between classes:

$$g_i(\vec{X}) - g_j(\vec{X}) = 0$$

In many cases the quadratic term can be ignored and the partitions take on the form of hyper-surfaces. In this case, the discrimination function can be reduced to a linear equation.

$$g_k(\vec{X}) = \vec{W}_k^T \vec{X} + b_k$$

This is very useful because there are simple powerful techniques to calculate the coefficients for linear functions from training data.

Vector between center of gravities

Suppose that we have two classes with mean and covariance ($\vec{\mu}_1$, Σ_1), and ($\vec{\mu}_2$, Σ_2). These can be used to define two linear discriminant functions:

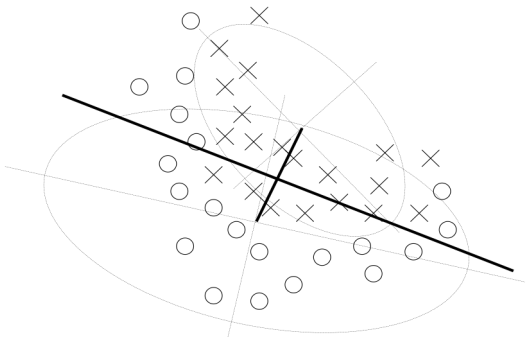
Let $g_1(\vec{X}) = \vec{W}_1^T \vec{X} + b_1$ and $g_2(\vec{X}) = \vec{W}_2^T \vec{X} + b_2$

where : $\vec{W}_k = \Sigma_k^{-1} \vec{\mu}_k$

and $b_k = -\frac{1}{2}(\mu_k^T \Sigma_k^{-1} \mu_k) - \frac{1}{2} Log\{\det(\Sigma_k)\} + Log\{p(\omega_k)\}$

The decision boundary is

$$g_1(\vec{X}) - g_2(\vec{X}) = 0$$
$$(\vec{W}_1^T - \vec{W}_2^T)\vec{X} + b_1 - b_2 = 0$$
$$(\Sigma_1^{-1}\vec{\mu}_1 - \Sigma_2^{-1}\vec{\mu}_2)^T X + b_1 - b_2 = 0$$



The direction is determined by the vector between the center of gravities of the two classes, weighted by the inverse of the covariance matrices.

This approach is based on the assumption that the two classes are well modeled by Normal density functions. This assumption is not reasonable in many cases.
If one of the classes is not well modeled as a normal, the results can be unreliable.

In some other cases, the data are so well separated that a large variety of hyperplanes can be used. In this case it can be interesting to use a simpler learning method.

Perceptrons

A perceptron is an incremental learning algorithm for linear classifiers invented by Frank Rosenblatt in 1956. The perceptron is an on-line learning method in which a linear classifier is improved by its own errors.

A perceptron learns a set of possible hyper-planes to separate training samples. When the training data are perfectly separated the data is said to be "separable". Otherwise, the data is said to be non-separable.

The "margin", γ, is the smallest separation between the two classes.

When are the training samples are separable, the algorithm uses the errors to update a plane until there are no more errors. When the training data is non-separable, the method may not converge, and must be stopped after a certain number of iterations.

The perceptron learns from training samples $\{\vec{X}_m\}$ with indicator variables $\{y_m\}$.

Note that for all positive examples.

$y_m \cdot (\vec{W}^T \vec{X}_m + B) > 0$ if the classification is correct.

The algorithm will apply a learning gain, α, to accelerate learning.

Algorithm:

$\vec{W}_o \leftarrow 0$; $b_o \leftarrow 0$; i = 0;
R $\leftarrow$ max { $\| \vec{X}_m \|$ }
REPEAT
  FOR *m = 1* TO M DO
    IF $y_m \cdot (\vec{W}_i^T \vec{X}_m + b_i) \le 0$ THEN
      $\vec{W}_{i+1} \leftarrow \vec{W}_i + \alpha \cdot y_m \cdot \vec{X}_m$
      $b_{i+1} \leftarrow b_i + \alpha \cdot y_m \cdot R^2$
      $i \leftarrow i + 1$
    END IF
  END FOR
UNTIL no mistakes in FOR loop.

After each stage the margin, γ_m, for each sample, m, is

$$\gamma_m = y_m \cdot (\vec{W}_i^T \vec{X}_m + b_i)$$

The coefficients must be normalized to compute the margin.

$$W_i' = \frac{W_i}{\| W_i \|} \qquad b_i' = \frac{b_i}{\| W_i \|}$$

The decision rule is as before:

$$\text{if } \operatorname{sgn}(\vec{W}_i^T \vec{X} + b_i)) > 0 \text{ then } \hat{\omega}_1 \text{ else } \hat{\omega}_2$$

The quality of a perceptron is given by the histogram of its margins, $h(\gamma_m)$
If the data is not linearly separable, then the Perceptron will not converge.

Fisher Linear Discriminant.

The discrimination problem can be viewed as a problem of projecting the D dimensional feature space onto a lower dimensional K dimensional space.

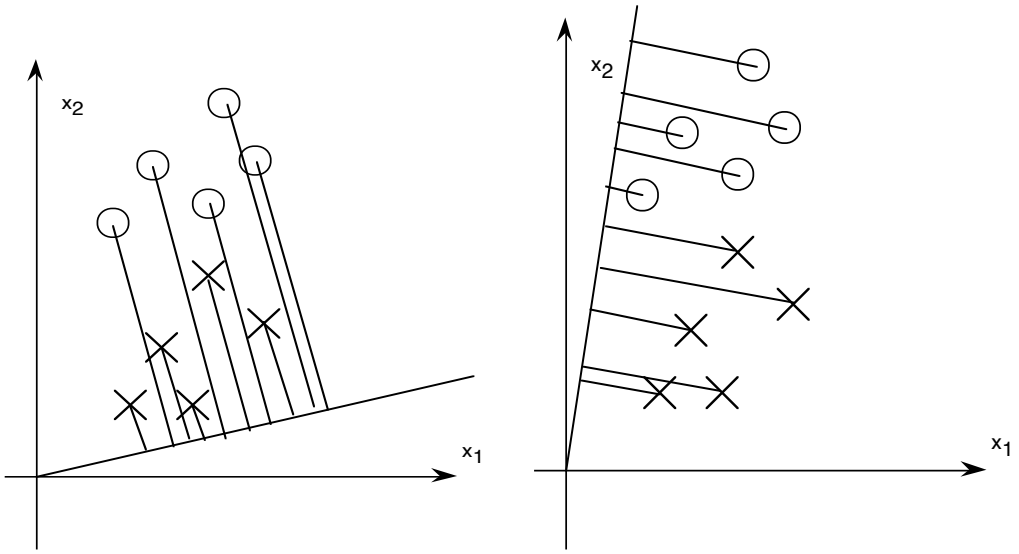
The tool for such projection is the Fisher discriminate.

Two Class solution

The principle of the Fisher linear discriminate is to project the vector X with D_x onto a variable z (D=1) by a linear projection F such that the classes are most separated.

$$z = \vec{F}^T \cdot \vec{X}$$

A Fisher metric, J(F) is used to choose F such that the two classes are most separated.



The error rates of the classification (FP, FN) depends on the direction of F.

Note that F is commonly normalized so that $\left\|\vec{F}\right\| = 1$

Assume a set of M_k training samples for each class, $\{\vec{X}_m^k\}$

The average for each class is:

$$\vec{\mu}^k = E\{\vec{X}^k\} = \frac{1}{M_k}\sum_{m=1}^{M_k}\vec{X}_m^k$$

Moments are invariant under projections. Thus the projection of the average is the average of the projection.

$$\mu_z^k = E\{F^T \cdot \vec{X}_m^k\} = F^T \cdot E\{\vec{X}_m^k\} = F^T \cdot \vec{\mu}_k$$

The inter-class distance between between classes 1 and 2 is

$$d_{12} = \mu_z^1 - \mu_z^2 = \vec{F}(\vec{\mu}_1 - \vec{\mu}_2)$$

The Fisher metric is designed to make the inter-class distance, d_{12}, as large as possible. The key concept is the "scatter" of the samples. Scatter can be seen as unormalised covariance.

The "scatter" for the M_k samples $\{\vec{X}_m^k\}$ of the set k is a matrix : $\mathbf{S}_k$.
This is the same as an "unnormalised" covariance.

$$S_k = M_k \Sigma_k = \sum_{m=1}^{M_k} (\vec{X}_m^k - \vec{\mu}^k)(\vec{X}_m^k - \vec{\mu}^k)^T$$

The transformation F projects the vector $\vec{X}$ onto a scalar z.

$$z = \vec{F}^T \cdot \vec{X}$$

The scatter of the class after projection is

$$S_z^k = \sum_{m=1}^{M_k} (z_m^k - \mu_z^k)^2$$

The fisher criteria tries to maximize the ratio of the separation of the classes compared to their scatter by maximizing the ratio of within and between class scatter.

$$J(F) = \frac{\left(\mu_z^1 - \mu_z^2\right)^2}{s_z^1 + s_z^2}$$

Let us define the between class scatter as $S_B = (\vec{\mu}_1 - \vec{\mu}_2)(\vec{\mu}_1 - \vec{\mu}_2)^T$

then $\left(\mu_z^1 - \mu_z^2\right)^2 = F^T\left((\vec{\mu}_1 - \vec{\mu}_2)(\vec{\mu}_1 - \vec{\mu}_2)^T\right)F = F^T S_B F$

And let us define within class scatter as

$$S_W = S_1 + S_2 = \sum_{m=1}^{M_1} (\vec{X}_m^1 - \vec{\mu}_1)(\vec{X}_m^1 - \vec{\mu}_1)^T + \sum_{m=1}^{M_2} (\vec{X}_m^2 - \vec{\mu}_2)(\vec{X}_m^2 - \vec{\mu}_2)^T$$

Then

$$s_z^1 + s_z^2 = F^T(S_1 + S_2)F = F^T S_W F$$

Then

$$J(F) = \frac{\left(\mu_z^1 - \mu_z^2\right)^2}{s_z^1 + s_z^2} = \frac{F^T S_B F}{F^T S_W F}$$

Taking the derivative with respect to F, we find that J(F) is maximized when

$$\left(F^T S_B F\right) S_W F = \left(F^T S_W F\right) S_B F$$

Because $S_B F$ is always in the direction $\vec{\mu}_1 - \vec{\mu}_2$

Dropping the scale factors $\left(F^T S_B F\right)$ and $\left(F^T S_W F\right)$ we obtain

$$S_W F = \vec{\mu}_1 - \vec{\mu}_2$$

and thus $F = S_W^{-1}\left(\vec{\mu}_1 - \vec{\mu}_2\right)$

Fisher's Discriminant for Multiple Classes.

Fisher's method can be extended to the derivation of K > 2 linar discriminates.
Let us assume that the number of features is greater than the number of classes,
D > K.

We will look for functions that project the D features on D' < D features to form a new feature vector, $\vec{Y} = \vec{w}^T \vec{X}$ (note that there is no constant term).

as before, we define the class Mean, $\vec{\mu}_k$, class Scatter S_k and within-class scatter S_W

Class Mean: $\vec{\mu}_k = \frac{1}{M_k} \sum_{m=1}^{M_k} \vec{X}_m^k$

Class Scatter: $S_k = \sum_{m=1}^{M_k} (\vec{X}_m^k - \vec{\mu}_k)(\vec{X}_m^k - \vec{\mu}_k)^T$

Within Class Scattter $\vec{\mu}_k = \frac{1}{M_k} \sum_{m=1}^{M_k} \vec{X}_m^k$

We need to generalization of the between class covariant.
The total mean is:

$$\vec{\mu} = \frac{1}{M} \sum_{k=1}^{K} \sum_{m=1}^{M_k} \vec{X}_m^k = \frac{1}{M} \sum_{k=1}^{K} M_k \vec{\mu}_k$$

The between class scatter is:

$$S_B = \sum_{k=1}^{K} M_k (\vec{\mu}_k - \vec{\mu})(\vec{\mu}_k - \vec{\mu})^T$$

Which gives the total scatter as

$$S_T = S_W + S_B$$

We can define similar scatters in the target space:

$$\vec{\mu}_k = \frac{1}{M_k} \sum_{m=1}^{M_k} \vec{Y}_m^k \qquad \vec{\mu} = \frac{1}{M} \sum_{k=1}^{K} \sum_{m=1}^{M_k} \vec{Y}_m^k = \frac{1}{M} \sum_{k=1}^{K} M_k \vec{\mu}_k$$

$$S'_W = \sum_{k=1}^{K} \sum_{m=1}^{M_k} (\vec{Y}_m^k - \vec{\mu}_k)(\vec{Y}_m^k - \vec{\mu}_k)^T$$

$$S'_B = \sum_{k=1}^{K} M_k(\vec{\mu}_k - \vec{\mu})(\vec{\mu}_k - \vec{\mu})^T$$

We want to construct a set of projections that maximizes the between class scatter

$$J(W)=Tr\{W \cdot S_W \cdot W^T)^{-1}(WS_BW^T)$$

The W values are detemined by the D eigenvectors of $S_W^{-1}S_B$ that correspond to the D largest Eigen-values.