Intelligent Systems: Reasoning and Recognition

James L. Crowley

ENSIMAG 2 / MoSIG M1 Second Semester 2014/2015

Lesson 16 8 April 2015

Linear Classifiers as Pattern Detectors

Contents

Bibliographical Sources :
"Pattern Recognition and Scene Analysis", R. E. Duda and P. E. Hart, Wiley, 1973.
"Pattern Recognition and Machine Learning", C. M. Bishop, Springer Verlag, 2006.

Notation

x	a variable
X	a random variable (unpredictable value)
V	The number of possible values for X (Can be infinite).
$\vec{x}$	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 k
k	Class index
K	Total number of classes
ω_k	The statement (assertion) 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.

N_k	Number of examples for the class k.
N	Total number of examples. $N = \sum_{k=1}^{K} N_k$
$P(X)$	Probability density function for X
$p(\vec{X})$	Probability density function for $\vec{X}$
$p(\vec{X} \mid \omega_k)$	Probability density for $\vec{X}$ the class k. ω_k = E $\in C_k$.
$\{\vec{X}_n\}$ $\{y_n\}$	N Training samples labeled with an indicator variable.

For two class problems, the indicator is: $y_n = +1$ for target class and $y_n = -1$ for other

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.

A pattern detector can be seen as a classifier with K=2.
Class k=1: The target pattern.
Class k=2: Everything else.

The pattern detector is learned as a detection function g(X) also known as a discriminant, followed by a decision rule. The detection function is learned from a set of training data composed of N sample observations $\{\vec{X}_n\}$ where each sample observation is labeled with an indicator variable

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

In todays lesson we will study linear detection functions (linear discriminants).

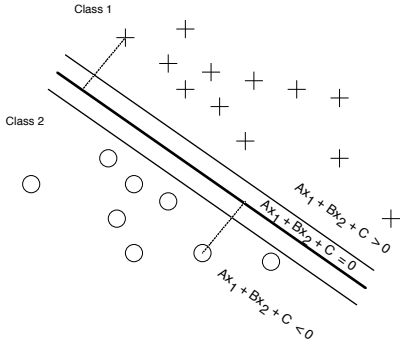
Linear detection function is a hyper-plane in the D dimensional space, $\vec{X}$, that separates observations of the target class from everything else.

A hyperplane is a set of points such that

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

This can be written: $\vec{W}^T \vec{X} + d = 0$
The decision rule is IF $\vec{W}^T \vec{X} + d > 0$ THEN E $\in$ C_1 else E $\notin$ C_1

We can add an additional bias term, B, that can act as an adjustable gain that sets the sensitivity of the detector. The bias term allows us to trade False positives for False negatives.



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

$d = -\vec{W}^T\vec{X}$ 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 $d' = \frac{d}{\|\vec{W}\|}$

In many cases it is convenient to treat d as the "0th" term of W, so that

$\vec{X} = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_D \end{pmatrix}$ and $\vec{W} = \begin{pmatrix} d \\ w_1 \\ \vdots \\ w_D \end{pmatrix}$

This is an form "homogeneous equation".
In this case our hyperplane detection function becomes:

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

A variety of techniques exist to estimate the plane.
Popular techniques include

1) Least Squares estimation
2) Perceptrons
3) Bayesian Discriminant functions
4) Fisher Discriminant analysis.

The best choice can depend on the nature of the data as well as the way in which the linear detection function is used. In many cases a fast sub-optimal technique may be preferred to a more expensive optimal technique.

To better understand we need a tool to explore the trade-off between making false detections (false positives) and missed detections (false negatives). The Receiver Operating Characteristic (ROC) provides such a tool

ROC Curves

Two-class linear classifiers have long been used for signal detection problems in communications. 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 (ROC) curve. This curve can be used to describe or compare any method for signal or pattern detection.

The ROC curve is generated by adding a variable Bias term to a linear discriminant

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

The bias term, B, is swept through a range of values.
Changing B changes the ratio of true positive detection to false positives.
The resulting curve is called a 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:

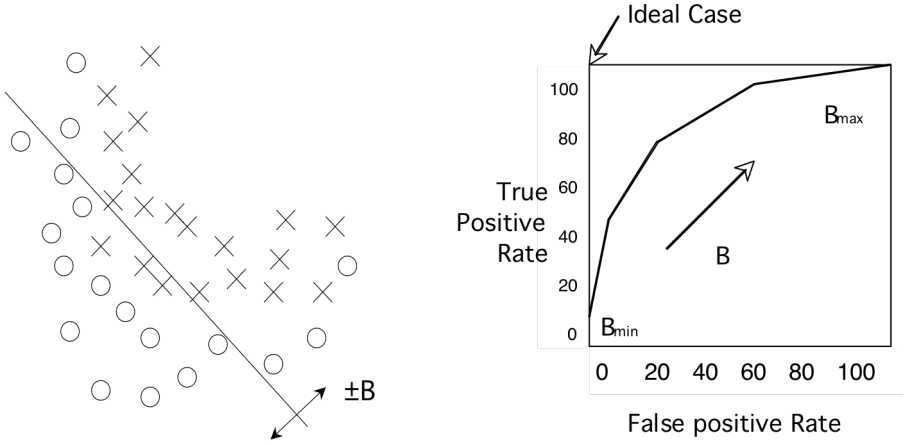
#TP as the number of True Positives,
#FP as the number of False Positives,
#TN as the number of True Negative,
#FN as the number of False Negatives.

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 a bias 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 $E \in C_2$

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

Least squares estimation of a hyperplane

A popular method for estimating a hyperplane is to compute a least-squares estimate using matrix algebra. Least squares estimation is a fast sub-optimal method that provides a direct, closed form solution to estimate a linear discriminant function from a set of labeled training data.

Assume a training set of N observations $\{\vec{X}_n\}$ where each observation is labeled with an indicator variable

$y_n = +1$ for examples of the target pattern (class 1)

$y_n = -1$ for all other examples (class 2)

Assume a notation in which the constant d is included as a "0th" term in X and W.

$$\vec{X} = \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_D \end{pmatrix} \text{ and } \vec{W} = \begin{pmatrix} d \\ w_1 \\ \vdots \\ w_D \end{pmatrix}$$

The detection function is $g(\vec{X}) = \vec{W}^T \vec{X}$

We seek the "best" $\vec{W}$. This can be determined by minimizing a "Loss" function that can be defined as the Square of the error.

$$L(\hat{W}) = \sum_{n=1}^{N} (y_n - \vec{X}_n^T \hat{W})^2$$

To build or function, we will use the M training samples to compose a matrix **X** and a vector **Y**.

$$X = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_{11} & x_{12} & \cdots & x_{1N} \\ x_{21} & x_{22} & \cdots & x_{2N} \\ \cdots & \cdots & \ddots & \vdots \\ x_{D1} & x_{D2} & \cdots & x_{DN} \end{pmatrix} \text{ (D+1 rows by N columns)}$$

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}$$ (N coefficients).

We can factor the loss function to obtain: $L(\hat{W}) = (\mathbf{Y} - \mathbf{X}^T\mathrm{W})^T (\mathbf{Y} - \mathbf{X}^T\mathrm{W})$

To minimize the loss function, we calculate the derivative and solve for W when the derivative is 0.

$$\frac{\partial L(\vec{W})}{\partial \vec{W}} = -2\,\mathbf{XY} + \mathbf{2}\,\mathbf{X}\mathbf{X}^T\vec{W} = 0$$

which gives $\mathbf{XY} = \mathbf{X}\mathbf{X}^T\vec{W}$

and thus $\vec{W} = (\mathbf{X}\mathbf{X}^T)^{-1}\,\mathbf{X}\mathbf{Y}$

An unknown event $\vec{X}$ can then be classified as

if $\vec{W}^T\vec{X} > 0$ then $\hat{\omega}_1$ else $\hat{\omega}_2$

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 learning can be incremental. New training samples can be used to update the perceptron.

The perceptron learns from a training samples $\{\vec{X}_n\}$ with indicator variables $\{y_n\}$. Note that for all correctly classified examples :

$$y_n \cdot (\vec{W}^T \vec{X}_n + d) > 0$$

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

Algorithm:

$\vec{W}_o \leftarrow 0$; $d_o \leftarrow 0$; i = 0;
R $\leftarrow$ max { $\|\vec{X}_n\|$ }
WHILE update DO
    update $\leftarrow$ FALSE;
    FOR *n = 1* TO N DO
        IF $y_n \cdot (\vec{W}_i^T \vec{X}_n + d_i) \le 0$ THEN
            update $\leftarrow$ TRUE
            $\vec{W}_{i+1} \leftarrow \vec{W}_i + \alpha \cdot y_m \cdot \vec{X}_m$
            $d_{i+1} \leftarrow d_i + \alpha \cdot y_m \cdot R^2$
            $i \leftarrow i + 1$
        END IF
    END FOR
END WHILE.

After each stage the margin, γ_n, for each sample, n, is

$$\gamma_n = y_n \cdot (\vec{W}_i^T \vec{X}_n + d_i)$$

The coefficients must be normalized to compute the margin.

$$\vec{W}_i' = \frac{\vec{W}_i}{\| \vec{W}_i \|} \qquad d_i' = \frac{d_i}{\| \vec{W}_i \|}$$

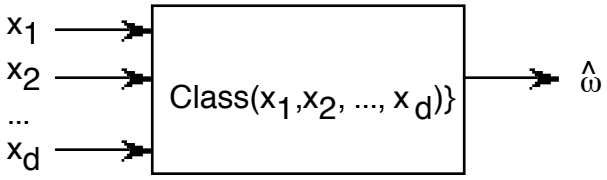
The decision rule then becomes:

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

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

Bayesian Linear Discriminant Functions

As we saw in earlier lessons, a Bayesian classifier maps a set of features $\vec{X}$ from an Observation, E into a class C_k from a set of K possible Classes.



Let ω_k be the proposition that the event belongs to class k: $\omega_k = E \in C_k$

ω_k Proposition that event E $\in$ the class k

In order to minimize the number of mistakes, we will maximize the probability that $\omega_k \equiv E \in T_k$

$$\hat{\omega}_k = \arg\max_{\omega_k} \left\{ P(\omega_k \mid \vec{X}) \right\}$$

We will call on two tools for this:

1) Baye's Rule : $P(\omega_k \mid \vec{X}) = \frac{p(\vec{X} \mid \omega_k)}{p(\vec{X})} P(\omega_k)$

2) Normal Density Functions $p(\vec{X}) = \mathcal{N}(\vec{X}; \vec{\mu}, \Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}} \det(\Sigma)^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{X}-\vec{\mu})^T \Sigma^{-1} (\vec{X}-\vec{\mu})}$

The classification function can be decomposed into two parts: d() and $g_k()$:

$$\hat{\omega}_k = d\left(g_k\left(\vec{X}\right)\right)$$

$g_k\left(\vec{X}\right)$: A discriminant function : $R^D \rightarrow R^K$

d() : a decision function $R^K \rightarrow \omega_k$

The discriminant is a vector of functions: $\vec{g}(\vec{X}) = \begin{pmatrix} g_1(\vec{X}) \\ g_2(\vec{X}) \\ \vdots \\ g_K(\vec{X}) \end{pmatrix}$

Quadratic discrimination functions can be derived directly from $p(\omega_k \mid \vec{X})$

$$p(\omega_k \mid \vec{X}) = \frac{P(\vec{X} \mid \omega_k)p(\omega_k)}{P(\vec{X})}$$

To minimize the number of errors, we will choose k such that

$$\hat{\omega}_k = \underset{\omega_k}{\arg\max}\{\frac{P(\vec{X} \mid \omega_k)p(\omega_k)}{P(\vec{X})}\}$$

but because P(X) is constant for all k, it is common to use a likelihood function:

$$\hat{\omega}_k = \underset{\omega_k}{\arg\max}\{P(\vec{X} \mid \omega_k)p(\omega_k)\}$$

This is called a "Maximum Likelihood" classifier.
The functions $g_k()$ are commonly constructed from the Log of the likelihood:

$$g_k(X) = Log\{P(\vec{X} \mid \omega_k)P(\omega_k)\}$$

as Log is a monotonic function.

For a Gaussian (Normal) density function

$$p(\vec{X} \mid \omega_k) = \mathcal{N}(\vec{X}; \vec{\mu}_k, \Sigma_k)$$

$$Log(p(\vec{X} \mid \omega_k)) = Log(\frac{1}{(2\pi)^{\frac{D}{2}} \det(\Sigma_k)^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{X}-\vec{\mu}_k)^T \Sigma_k^{-1}(\vec{X}-\vec{\mu}_k)})$$

$$Log(p(\vec{X} \mid \omega_k)) = -\frac{D}{2}Log(2\pi) - \frac{1}{2}Log\{Det(\Sigma_k)\} - \frac{1}{2}(\vec{X}-\vec{\mu}_k)^T \Sigma_k^{-1}(\vec{X}-\vec{\mu}_k)$$

We can observe that $-\frac{D}{2}Log(2\pi)$ can be ignored because it is constant for all k.

The discrimination function becomes:

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

Different families of Bayesian classifiers can be defined by variations of this formula.

This becomes more evident if we reduce the equation to a quadratic polynomial.
This function can be reduced to a standard (canonical) form.
Let us start with the term $(\vec{X}-\vec{\mu}_k)^T\Sigma_k^{-1}(\vec{X}-\vec{\mu}_k)$.

This can be rewritten as :

$$(\vec{X}-\vec{\mu}_k)^T\Sigma_k^{-1}(\vec{X}-\vec{\mu}_k) = \vec{X}^T\Sigma_k^{-1}\vec{X} - \vec{X}^T\Sigma_k^{-1}\vec{\mu}_k - \vec{\mu}_k{}^T\Sigma_k^{-1}\vec{X} + \vec{\mu}_k{}^T\Sigma_k^{-1}\vec{\mu}_k$$

We note that $\vec{X}^T\Sigma_k^{-1}\vec{\mu}_k = \vec{\mu}_k{}^T\Sigma_k^{-1}\vec{X}$
and thus : $-\vec{X}^T\Sigma_k^{-1}\vec{\mu}_k - \vec{\mu}_k{}^T\Sigma_k^{-1}\vec{X} = -(2\Sigma_k^{-1}\vec{\mu}_k)^T\vec{X}$

we define: $\vec{W}_k = -2\Sigma_k^{-1}\vec{\mu}_k$
to obtain $-\vec{X}^T\Sigma_k^{-1}\vec{\mu}_k - \vec{\mu}_k{}^T\Sigma_k^{-1}\vec{X} = \vec{W}_k{}^T\vec{X}$

Let us also define $D_k = -\frac{1}{2}\Sigma_k^{-1}$

The remaining terms are constant. Let us defined the constant

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

which gives a quadratic polynomial

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

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

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

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

A set of K discrimination functions $g_k(\vec{X})$ partitions the space $\vec{X}$ into a disjoint set of regions with quadratic boundaries. The boundaries are the functions $g_i(\vec{X}) - g_j(\vec{X}) = 0$
The boundaries are defined by points for which

$$g_i(\vec{X}) = g_j(\vec{X}) \geq g_k(\vec{X}) \; \forall k \neq i, j$$

Under certain conditions, the quadratic discrimination function can be simplified by eliminating either the quadratic or the linear term.

For example if $\vec{N}_s >> \vec{N}_k$ then the term Σ_k will be nearly constant for all k.
In this case, the discrimination function can be reduced to a linear equation.

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

Bayesian Linear Detectors:

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:

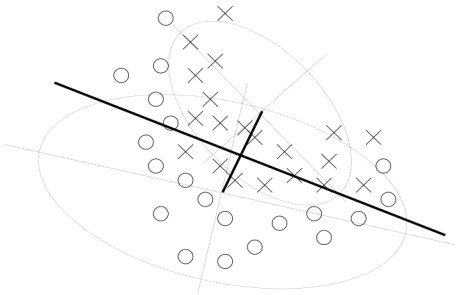
Let $g_1(\vec{X}) = \vec{W}_1^T \vec{X} + d_1$ and $g_2(\vec{X}) = \vec{W}_2^T \vec{X} + d_2$

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

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

The decision boundary is

$$g_1(\vec{X}) - g_2(\vec{X}) = 0$$
$$(\vec{W}_1^T - \vec{W}_2^T)\vec{X} + d_1 - d_2 = 0$$
$$(\Sigma_1^{-1}\vec{\mu}_1 - \Sigma_2^{-1}\vec{\mu}_2)^T X + d_1 - d_2 = 0$$



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.