AARHUS
UNIVERSITET

Optimization and Data Analytics

Alexandros Iosifidis
@
Aarhus University, Department of Engineering

Linear Decision Functions

The decision function g(**x**) divides the feature space in two regions:

1. If g(**x**) > 0, then we say that x belongs to class c1
2. If g(**x**) < 0, then we say that x belongs to class c2
3. If g(**x**) = 0, x is on the hyperplane.

Due to the above (binary) form of the discriminant function, two-class classification problems are usually called as binary classification problems.

We will see how to define multi-class classifiers later.

Linear Decision Functions

Given a set of N samples, each represented by a vector $\mathbf{x}_i \in \mathbb{R}^D$, and the corresponding labels $l_i = \{-1,1\}$ we want to optimize the parameters of $g(\cdot)$ in order to define a discriminant hyperplane discriminating the two classes.

We will do this optimization of the parameters of $g(\cdot)$ without setting assumptions on the distributions of each class.

The only assumption we will make is that the decision function corresponds to a linear discriminator (a hyperplane).

Non-linearly separable case

Instead of focusing on the misclassified samples, we can try to map the training vectors to some pre-fixed (target) values.

That is, we can optimize **w** so that

$$\mathbf{w}^T\mathbf{x}_i = b_i, \quad i = 1, \ldots, N$$

or by using a matrix notation

$$\mathbf{X}^T\mathbf{w} = \mathbf{b}$$

Non-linearly separable case

In order to determine the optimal **w**, we optimize (minimize) for

$$\mathcal{J}_s = \sum_{i=1}^{N} (\mathbf{w}^T \mathbf{x}_i - b_i)^2 = \|\mathbf{X}^T \mathbf{w} - \mathbf{b}\|_2^2$$

Why do we use the square?
- Easier to take the derivative
- Square root is monotonic

Why use the L2 norm?
- To get a single value instead of multiple values

Demo: https://phet.colorado.edu/sims/html/least-squares-regression/latest/least-squares-regression_en.html

Non-linearly separable case

In order to determine the optimal **w**, we optimize (minimize) for

$$\mathcal{J}_s = \sum_{i=1}^{N} (\mathbf{w}^T \mathbf{x}_i - b_i)^2 = \|\mathbf{X}^T \mathbf{w} - \mathbf{b}\|_2^2$$

The derivative w.r.t. **w** is

$$\begin{aligned} \nabla \mathcal{J}_s &= \nabla \left[(\mathbf{X}^T \mathbf{w} - b)^T (\mathbf{X}^T \mathbf{w} - b) \right] \\ &= \nabla \left[\mathbf{w}^T \mathbf{X} \mathbf{X}^T \mathbf{w} - 2\mathbf{w}^T \mathbf{X} \mathbf{b} + \mathbf{b}^T \mathbf{b} \right] \\ &= 2\mathbf{X}\mathbf{X}^T \mathbf{w} - 2\mathbf{X}\mathbf{b}, \end{aligned}$$

Non-linearly separable case

In order to determine the optimal **w**, we optimize (minimize) for

$$\mathcal{J}_s = \sum_{i=1}^{N} (\mathbf{w}^T \mathbf{x}_i - b_i)^2 = \|\mathbf{X}^T \mathbf{w} - \mathbf{b}\|_2^2$$

Setting the derivative w.r.t. **w** to zero, we have

$$\nabla \mathcal{J}_s = 0 \Rightarrow \mathbf{X}\mathbf{X}^T \mathbf{w} = \mathbf{X}\mathbf{b} \Rightarrow \mathbf{w} = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{b} = \mathbf{X}^{\dagger}\mathbf{b}$$

Non-linearly separable case

In order to determine the optimal $\mathbf{w}$, we optimize (minimize) for

$$\mathcal{J}_s = \sum_{i=1}^{N} (\mathbf{w}^T \mathbf{x}_i - b_i)^2 = \|\mathbf{X}^T \mathbf{w} - \mathbf{b}\|_2^2$$

Setting the derivative w.r.t. $\mathbf{w}$ to zero, we have

$$\nabla \mathcal{J}_s = 0 \Rightarrow \mathbf{X}\mathbf{X}^T \mathbf{w} = \mathbf{X}\mathbf{b} \Rightarrow \mathbf{w} = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{b} = \mathbf{X}^{\dagger}\mathbf{b}$$

From LDA, we saw that when N < D then the rank(A)=N.
If the data set has the number of samples are higher than dimension then matrix can be inverted.

Non-linearly separable case

In order to determine the optimal $\mathbf{w}$, we optimize (minimize) for

If X are linearly separable Js will be 0, otherwise it will larger than 0.

$$\mathcal{J}_s = \sum_{i=1}^{N} (\mathbf{w}^T \mathbf{x}_i - b_i)^2 = \|\mathbf{X}^T \mathbf{w} - \mathbf{b}\|_2^2$$

In the case where X is square matrix, we can prove that the pseudo inverse is the same as the inverse of the matrix

Setting the derivative w.r.t. $\mathbf{w}$ to zero, we have

$$\nabla \mathcal{J}_s = 0 \Rightarrow \mathbf{X}\mathbf{X}^T \mathbf{w} = \mathbf{X}\mathbf{b} \Rightarrow \mathbf{w} = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{b} = \mathbf{X}^\dagger \mathbf{b}$$

In case we cannot apply the inverse, we use the pseudo-inverse because the matrix may not be a square matrix

$$\mathbf{X}^\dagger = \lim_{\epsilon \to 0} (\mathbf{X}\mathbf{X}^T + \epsilon \mathbf{I})^{-1}\mathbf{X}$$

Adding a small number epsilon makes the columns independent. Check the pictures

Non-linearly separable case

Thus, the optimal weight vector **w** is given by

$$\mathbf{w} = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{b} = \mathbf{X}^\dagger\mathbf{b}$$

How to select the values of **b**?

Non-linearly separable case

Thus, the optimal weight vector **w** is given by

$$\mathbf{w} = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{b} = \mathbf{X}^{\dagger}\mathbf{b}$$

How to select the values of **b**?

w can be calculated using the above expression, or it can be calculated by following an iterative optimization process

Non-linearly separable case

When the N or D is large then we use the iterative approaches.

Algorithm 10: Iterative LMS

1: Initialize the parameters $\mathbf{w}$, $\eta(\cdot)$, $\mathbf{b}$, θ, $t = 0$
2: **Do** $t \leftarrow t + 1$
3: $\quad \mathbf{w} \leftarrow \mathbf{w} - \eta(t)\mathbf{X}(\mathbf{X}^T\mathbf{w} - \mathbf{b})$
4: **until** $\eta(t)\mathbf{X}(\mathbf{X}^T\mathbf{w} - \mathbf{b}) < \theta$

Notice that the optimal w is the same because our optimisation function is quadratic.

Algorithm 11: Sample-based iterative LMS

1: Initialize the parameters $\mathbf{w}$, $\eta(\cdot)$, $\mathbf{b}$, θ, $t = 0$
2: **Do** $t \leftarrow t + 1$
3: $\quad$ Select (randomly) a vector $\mathbf{x}_i$
4: $\quad \mathbf{w} \leftarrow \mathbf{w} - \eta(t)(\mathbf{w}^T\mathbf{x}_i - b_i)\mathbf{x}_i$
5: **until** $\eta(t)(\mathbf{w}^T\mathbf{x}_i - b_i)\mathbf{x}_i < \theta$

Generalized Linear Discriminant Functions

In all the above methods, the decision function has the form

$$g(\mathbf{x}) = w_0 + \sum_{d=1}^{D} w_d x_d$$

And corresponds to a linear discriminant rule (hyperplane).

However, most classification problems cannot be solved linearly. How can we extend what we have learnt until now to define nonlinear decision functions?

Generalized Linear Discriminant Functions

This can be done in two ways:

1. Changing the form of the decision function
2. Changing the data representation

Generalized Linear Discriminant Functions

We can change the decision function by adding more terms involving products of pairs of elements of x

$$g(\mathbf{x}) = w_0 + \sum_{d=1}^{D} w_d x_d \sum_{d=1}^{D} \sum_{l=1}^{D} Q_{dl} x_d x_l = w_0 + \mathbf{w}^T \mathbf{x} + \mathbf{x}^T \mathbf{Q} \mathbf{x}$$

What is the decision function shape in this case?

Generalized Linear Discriminant Functions

We can change the decision function by adding more terms involving products of pairs of elements of x

$$g(\mathbf{x}) = w_0 + \sum_{d=1}^{D} w_d x_d + \sum_{d=1}^{D}\sum_{l=1}^{D} Q_{dl} x_d x_l = w_0 + \mathbf{w}^T\mathbf{x} + \mathbf{x}^T\mathbf{Q}\mathbf{x}$$

where $\mathbf{Q} \in \mathbb{R}^{D\times D}$ is an additional weight matrix. Since $x_d x_l = x_l x_d$, $Q_{dl} = Q_{ld}$, i.e. $\mathbf{Q}$ is a symmetric matrix. Since $\mathbf{Q}$ is a parameter of the classifier, we can define it to be a symmetric and nonsingular matrix.

Q is fixed

What is the decision function shape in this case?

Generalized Linear Discriminant Functions

We can change the decision function by adding more terms involving products of pairs of elements of x

$$g(\mathbf{x}) = w_0 + \sum_{d=1}^{D} w_d x_d + \sum_{d=1}^{D}\sum_{l=1}^{D} Q_{dl} x_d x_l = w_0 + \mathbf{w}^T\mathbf{x} + \mathbf{x}^T\mathbf{Q}\mathbf{x}$$

What is the decision function shape in this case? The properties of the classifier are then defined by the properties of the matrix $\tilde{\mathbf{W}} = \frac{1}{(\mathbf{w}^T\mathbf{Q}^{-1}\mathbf{w}-4w_0)}\mathbf{Q}$. If $\tilde{\mathbf{W}}$ is a positive scaled version of the identity matrix, the decision function cooresponds to a hypersphere in $\mathbb{R}^D$. If $\tilde{\mathbf{W}}$ is positive definite, the decision function corresponds to a hyperellipsoid. If $\tilde{\mathbf{W}}$ is not positive semi-definite (i.e. some of its eigenvalues are positive and others negative), the decision function corresponds to a hyperhyperboloid.

This is not used in practical. Just to show that is possible to use

Generalized Linear Discriminant Functions

In order to obtain a nonlinear decision function, we can also define a nonlinear function, mapping the original data representations **x** to another one **φ**.

Examples:

$$\phi = [1 \;\; x \;\; x^2]$$

Generalized Linear Discriminant Functions

In order to obtain a nonlinear decision function, we can also define a nonlinear function, mapping the original data representations **x** to another one **φ**.

Examples:

$$\phi = [1 \;\; x \;\; x^2]$$

$$\phi = [1 \;\; x_1 \;\; x_2 \;\; x_1 x_2]^T$$

Can you imagine other mappings?

Generalized Linear Discriminant Functions

In order to obtain a nonlinear decision function, we can also define a nonlinear function, mapping the original data representations **x** to another one **φ**.

Examples:

$$\phi = [1 \;\; x \;\; x^2]$$

$$\phi = [1 \;\; x_1 \;\; x_2 \;\; x_1 x_2]^T$$

Defining a linear decision function (hyperplane) using **φ**

$$g(\mathbf{y}) = \mathbf{w}^T \phi$$

corresponds to a nonlinear decision function for x.

Extension to multi-class classification

Until now we have seen how to optimize the parameters of a binary classifier.

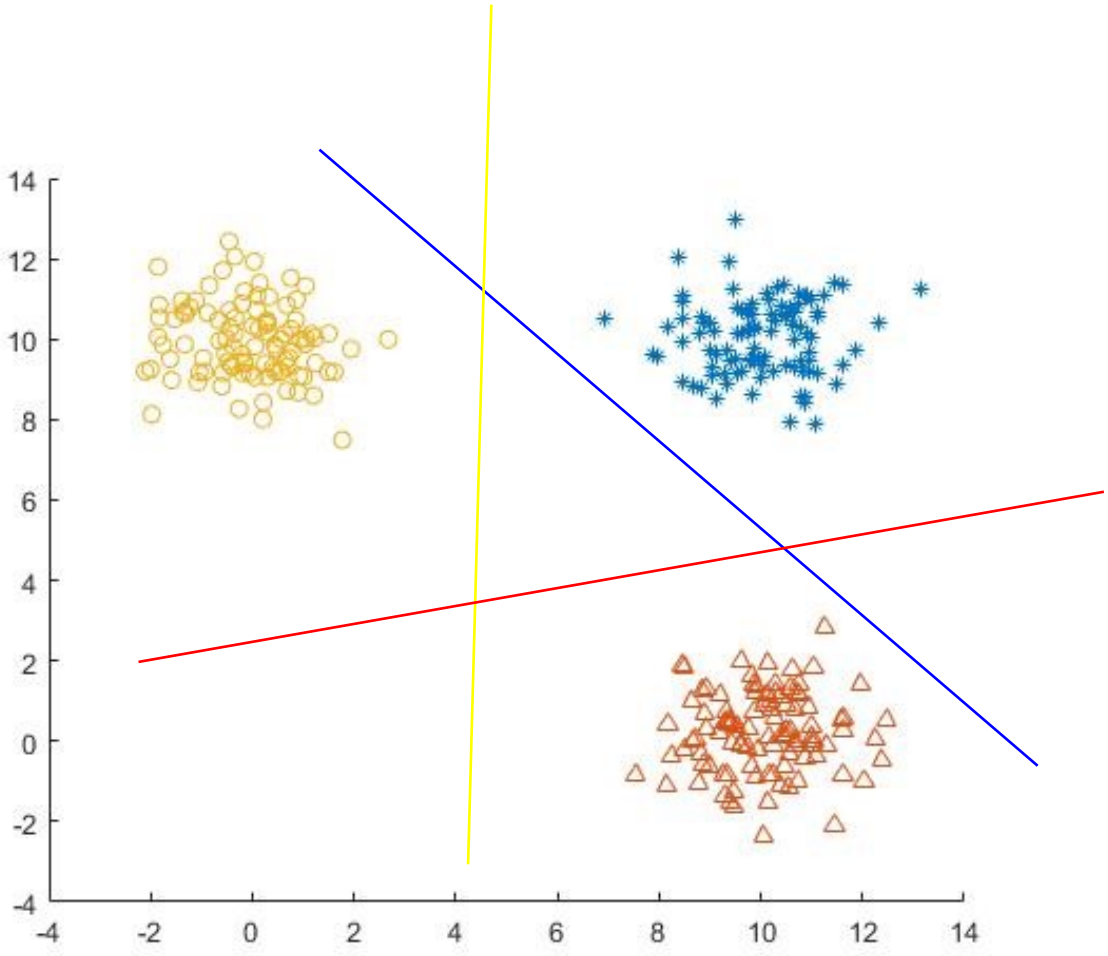
We can use multiple binary classifiers in order to solve multi-class classification problems.

There are two schemes:

1. One-versus-Rest classification
2. One-versus-One classification

Extension to multi-class classification

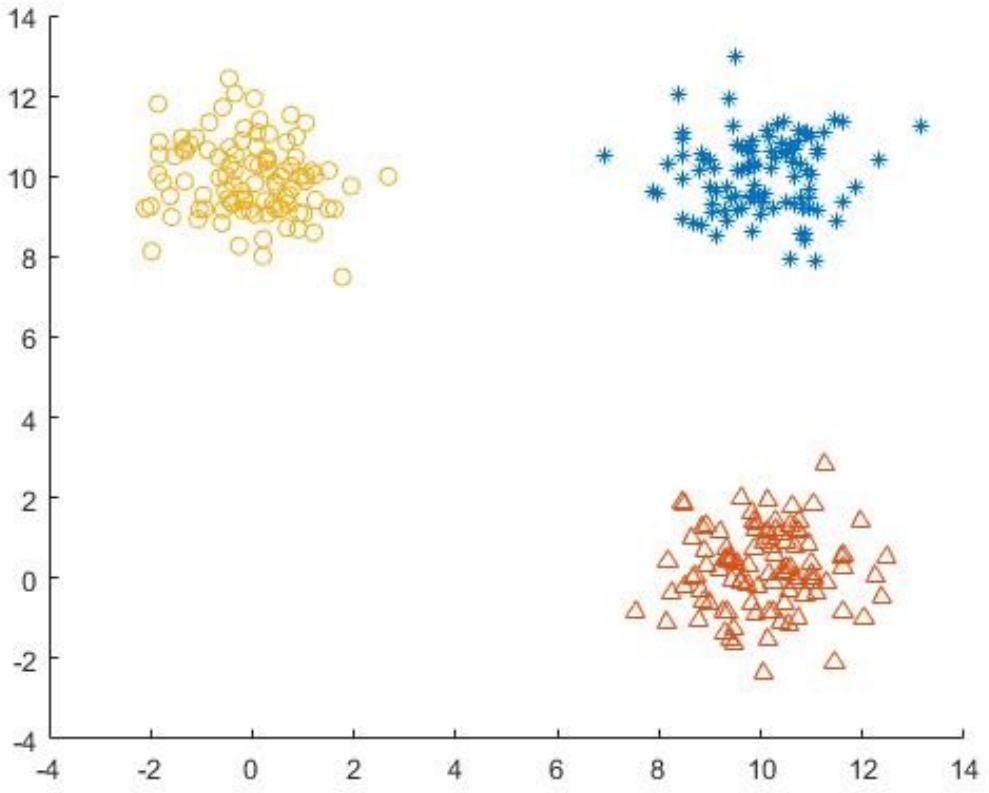
One-versus-Rest



Extension to multi-class classification

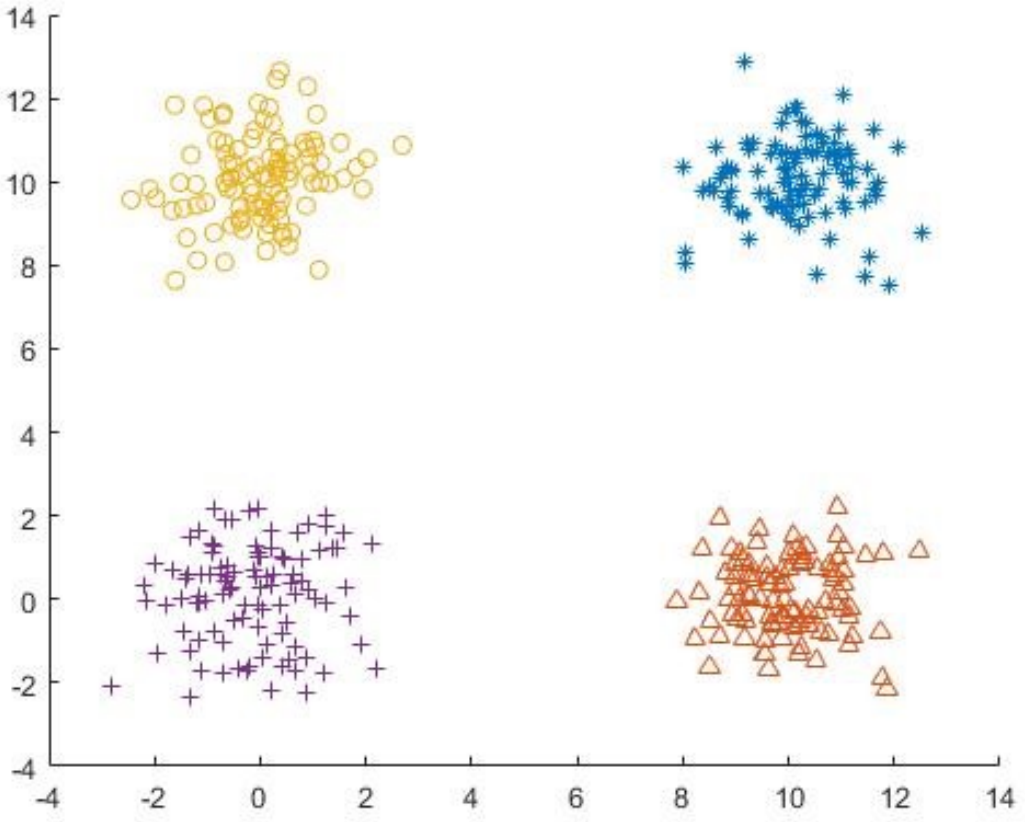
One-versus-One

We create pairs of decision function.



Extension to multi-class classification

One-versus-Rest



Extension to multi-class classification

One-versus-One