Classification, Perceptron and Fisher's LDA

Xinrui Lyu

Nov 6 – 8, 2019

Classification – The setting

Goal: Assign a d-dimensional input vector $\mathbf{x}$ to one of K classes $y_k, k = 1, \ldots, K$

- divide the input space $\mathbf{X}$ into decision regions
- the decision boundaries (or surfaces) can be of any shape
- However, for mathematical simplicity, we would like to use decision boundaries of the form $\mathbf{w}^T\mathbf{x} = 0$

Classification – The setting

Generalized linear functions

In linear regression, the model for y was a linear function of input x and parameter w (weights). However, in classification, we need discrete numbers or probabilities as output. Idea: Apply some non-linear function f to $\mathbf{w}^T\mathbf{x}$: $\quad y(\mathbf{x}) = f(\mathbf{w}^T\mathbf{x})$.

Examples:

- take the sign: $y(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^T\mathbf{x})$
- sigmoidal functions like $y(\mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}} = \sigma(\mathbf{w}^T\mathbf{x})$

Important note: Even though the function $y(\mathbf{x})$ is non-linear in parameter $\mathbf{w}$, the decision surface is still linear as it is based only on the dot product.

Classification approaches

- **Discriminative**:
  - **probabilistic**: model class posterior $\mathbb{P}[Y|\mathbf{X}]$ and decide based on Bayes decision criteria (e.g. Logistic regression)
  - **non-probabilistic**: construct a discriminant function that directly assigns input $\mathbf{x}$ to a class k, without estimating underlying probability distributions (e.g. Perceptron, Fisher's LDA)
- **Generative probabilistic**: Model both class prior probabilities $\mathbb{P}[Y]$ and class-conditional probability densities $\mathbb{P}[\mathbf{X}|Y]$ to get the posterior

Linear Classifiers

Plain vanilla classifier: splitting a dataset into two classes (a *dichotomy*) using a *linear* separating hyperplane

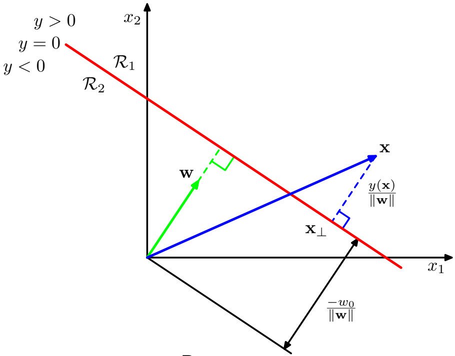
$$f(\mathbf{x}) = w_0 + \sum_{d=1}^{D} w_d x_d = w_0 + \mathbf{w}^T \mathbf{x} \tag{1}$$

- Labelled data point is pair $(\mathbf{x}_{(n)}, y_{(n)})$, with $y \in \{-1, +1\}$
- $\hat{y}_{(n)} = \mathrm{sgn}(f(\mathbf{x}_{(n)}))$ is classifier output

Extensions to $k > 2$ classes and non-linear decision surfaces often use this as the starting point.

Separating Hyperplane

Affine hyperplane geometry:



$$f(\mathbf{x}) = w_0 + \sum_{d=1}^{D} w_d x_d = w_0 + \mathbf{w}^T \mathbf{x}.$$

Homogeneous Coordinates

Two tricks to simplify notation:

1. Subsume w_0: augment feature vector $\mathbf{x} \in \mathbb{R}^D$ into $\tilde{\mathbf{x}} := (\mathbf{x}, 1) \in \mathbb{R}^{D+1}$

$$w_0 + \sum_{d=1}^{D} w_d x_d = \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}$$

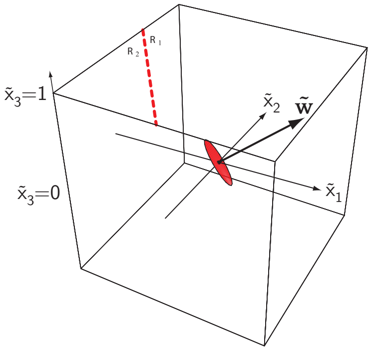
   with $\tilde{\mathbf{w}} = (\mathbf{w}, w_0)$.

2. Replace two sided test with one inequality: $\mathbf{x}_{(n)}$ correctly classified if

$$y_{(n)} f(\mathbf{x}_{(n)}) > 0.$$

Geometry of Homogeneous Coordinates

Augmented feature space:



- Data point $(x_1, x_2, 1)$ and weight vector $\tilde{\mathbf{w}}$ live in $\mathbb{R}^3$
- Shifting decision boundary in $\mathbb{R}^2$ means rotating $\tilde{\mathbf{w}}$ around origin

Points and (Hyper)-planes

End result:

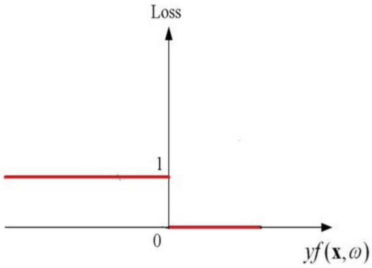
- Data point (x_1, x_2) is described by $\mathbf{x} = (x_1, x_2, 1)$ in homogeneous coordinates.
- A plane $ax_1 + bx_2 + c$ is described by a vector $\mathbf{w} = (a, b, c)$.
- Point $(x_1, x_2, 1)$ is on plane a (a, b, c) when $\mathbf{x}^T\mathbf{w} = 0$.
  If not on plane, side depends on the sign of $\mathbf{x}^T\mathbf{w}$.
- Our goal is:
  given a number of points $\{\mathbf{x_i}\}_{i=1}^{n}$ and a class corresponding to each point $\{y_i\}_{i=1}^{n}$ ($y_i \in \{-1, +1\}$), find a good separating plane.

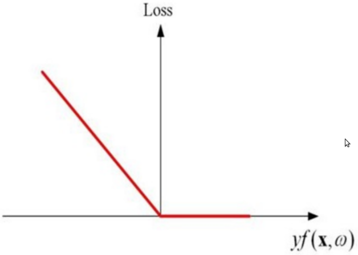
  Two solutions considered: Perceptrons, SVMs

Perceptron

How do we optimize such a classifier?

- We need a loss function, e.g. 0-1 loss or Perceptron loss





Perceptron

Perceptron algorithm minimizes the perceptron loss:

$$\hat{w} = \operatorname{argmin} \sum_{i=1}^{N} l_P(\mathbf{w}, y_i, \mathbf{x}_i)$$

$$l_P(\mathbf{w}, y_i, \mathbf{x}_i) = \max(0, -y\mathbf{w}^T\mathbf{x})$$

- this loss function is convex, differentiable almost everywhere and the gradient is zero only when the classification is correct, so we can use gradient methods
- gradient:

$$\nabla_w l_P(\mathbf{w}, y_i, \mathbf{x}_i) = \begin{cases} 0 & y\mathbf{w}^T\mathbf{x} \geq 0 \\ -y_i\mathbf{x}_i & y\mathbf{w}^T\mathbf{x} < 0 \end{cases}$$

Perceptron

Perceptron algorithm uses gradient descent or stochastic gradient descent to minimize perceptron loss:

- gradient descent: update until convergence:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla_{\mathbf{w}} \sum_{i=1}^{N} l_P(\mathbf{w}, y_i, \mathbf{x}_i)$$

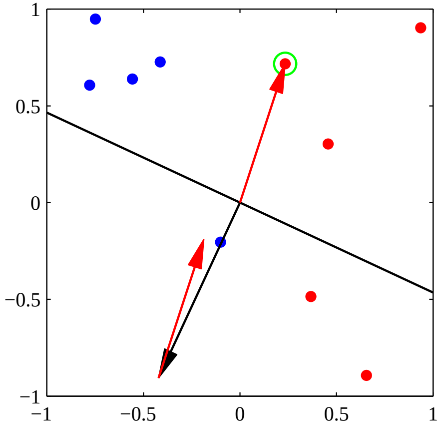
- stochastic gradient descent: for $i = 1, 2, \ldots$ pick random point $\mathbf{x}_i$, y_i from the dataset and update:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla_{\mathbf{w}} l_P(\mathbf{w}, y_i, \mathbf{x}_i)$$

- η is called learning rate

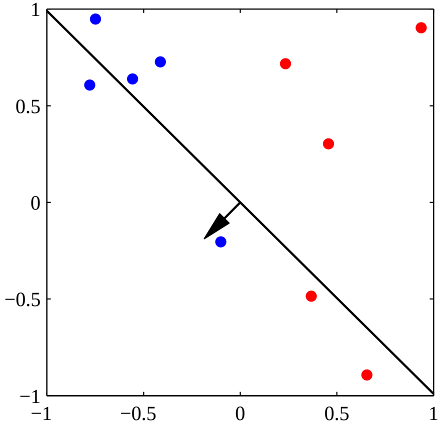
Perceptron Learning

Single sample perceptron adds $\tilde{\mathbf{x}}_{(t)}$ to $\tilde{\mathbf{w}}_{(t)}$ if misclassified:



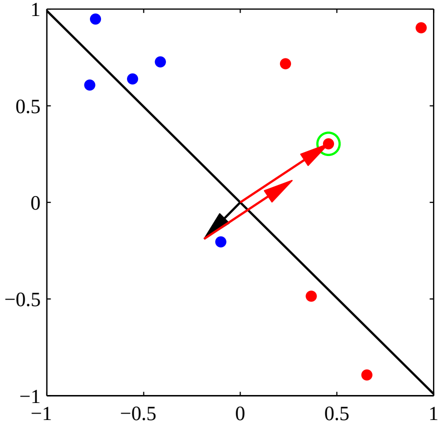
Perceptron Learning

Single sample perceptron adds $\tilde{\mathbf{x}}_{(t)}$ to $\tilde{\mathbf{w}}_{(t)}$ if misclassified:



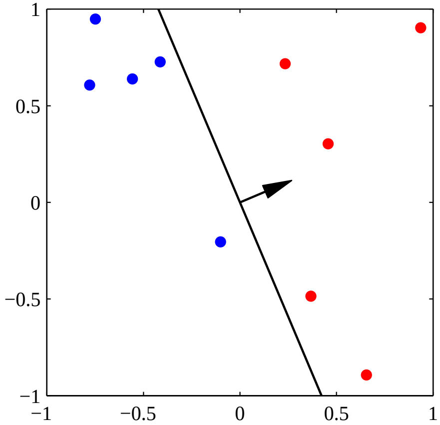
Perceptron Learning

Single sample perceptron adds $\tilde{\mathbf{x}}_{(t)}$ to $\tilde{\mathbf{w}}_{(t)}$ if misclassified:



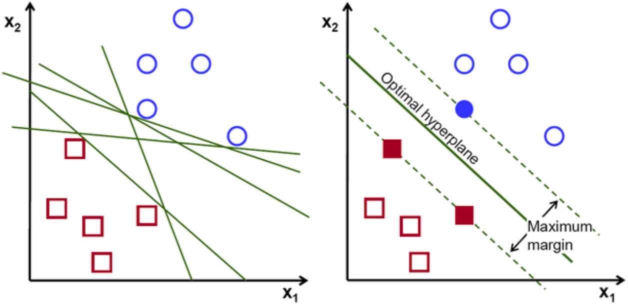
Perceptron Learning

Single sample perceptron adds $\tilde{\mathbf{x}}_{(t)}$ to $\tilde{\mathbf{w}}_{(t)}$ if misclassified:

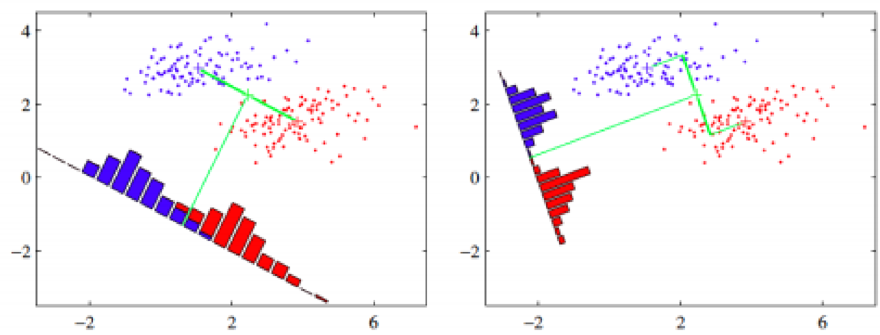


Perceptron

- Convergence of perceptron algorithm: if data is perfectly linearly separable, the perceptron will find an exact solution in a finite number of steps (otherwise will never converge)
- however, still a lot of steps, and even then, how good is the solution?..



Fisher's LDA



Fisher's LDA

Fisher's LDA maximizes the following objective:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_W \mathbf{w}},$$

where S_B is the "between classes scatter matrix", S_W is the "within classes scatter matrix".

$$S_B = \sum_c (\mu_c - \bar{\mathbf{x}})(\mu_c - \bar{\mathbf{x}})^T$$

$$S_W = \sum_c \sum_{i \in c} (\mathbf{x}_i - \mu_c)(\mathbf{x}_i - \mu_c)^T$$

Noted: $J(\mathbf{w})$ is invariant to the scaling of the vectors $\mathbf{w} \rightarrow \alpha \mathbf{w}$

Fisher's LDA

We can choose $\mathbf{w}$ such that $\mathbf{w}^T S_W \mathbf{w} = 1$.
Hence the optimization problem of the Fisher's LDA is transformed into a constrained optimzation problem:

$$\begin{aligned} \min_{\mathbf{w}} \quad & -\frac{1}{2}\mathbf{w}^T S_B \mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^T S_W \mathbf{w} = 1 \end{aligned}$$

Lagrange Multipliers, why is it interesting?

Lagrange Multipliers are a way to solve constrained optimization problems.

Consider the problem of finding

$$(x_1^*, x_2^*) = \arg\max f(x_1, x_2)$$

subject to the constraint:

$$h(x_1, x_2) = 0 \quad \text{or} \quad g(x_1, x_2) \leq 0$$

or both

Constraint optimization with equality constraints

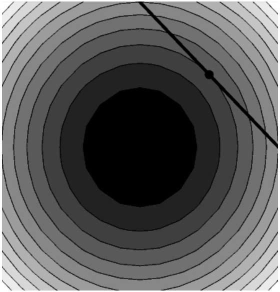
For now we focus on the easier case with 1 equality constraint.

Example

$$f(x_1, x_2) = {x_1}^2 + {x_2}^2$$

subject to

$$h(x_1, x_2) = x_1 + x_2 - 2 = 0$$

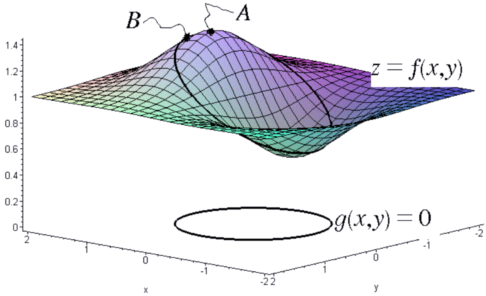


Lagrange multipliers solution: Geometrical view

Considering the variable $\mathbf{x} \in \mathbb{R}^D$, then the constraint equation

$$h(\mathbf{x}) = 0$$

represents a $(D-1)$-dimensional surface in $\mathbb{R}^D$, denoted $\mathcal{C}$.



Observation 1

At any point on the constraint surface $\mathcal{C}$, the gradient $\nabla h(\mathbf{x})$ is orthogonal to the surface.

Taking the Taylor expansion around $\mathbf{x}$:

$$h(\mathbf{x} + \epsilon) \simeq h(\mathbf{x}) + \epsilon^T \nabla h(\mathbf{x})$$

Since both $\mathbf{x}$ and $\mathbf{y} = \mathbf{x} + \epsilon$ are on the surface $\mathcal{C}$, then $h(\mathbf{x}) = h(\mathbf{x} + \epsilon) = 0$. Therefore,

$$\epsilon^T \nabla h(\mathbf{x}) \simeq 0$$

and, in the limit $\|\epsilon\| \to 0$, also

$$\epsilon^T \nabla h(\mathbf{x}) \to 0$$

Now ϵ is parallel to $\mathcal{C}$ therefore $\nabla h \perp \mathcal{C}$.

Observation 2

The optimum is $\mathbf{x}^*$ on $\mathcal{C}$ such that f is maximized.
At $\mathbf{x}^*$ it holds that

$$\nabla f(\mathbf{x}^*) \perp \mathcal{C}$$

Otherwise it would be possible to increase the value of f by moving a short distance along $\mathcal{C}$.

Deriving the Lagrange equation

From Observation $1 + 2$:

∇f is parallel (or anti-parallel) to ∇h , $\nabla f(\mathbf{x}^*) \| \nabla h(\mathbf{x}^*)$
and $\exists \lambda \neq 0$ such that

$$\nabla f(\mathbf{x}^*) + \lambda \nabla h(\mathbf{x}^*) = 0 \,. \tag{2}$$

λ is called the Lagrange Multiplier,

the Lagrangian Function is

$$L(\mathbf{x}, \lambda) := f(\mathbf{x}) + \lambda h(\mathbf{x}) \,.$$

Cool: Stationary points of L are optimum of f

Condition (2) of optimality of f is obtained by

$$\nabla_{\mathbf{x}} L = 0$$

while the constraint equation $h(\mathbf{x}) = 0$ is obtained from

$$\frac{\partial L}{\partial \lambda} = 0$$

Therefore to find the solution, do the following:

1. Write the Lagrangian $L(\mathbf{x}, \lambda)$ from $f(\mathbf{x})$ and $h(\mathbf{x})$
2. Solve the ordinary system of equations obtained from $\nabla L = 0$

Multiple equality constraints?

The same technique allows us to solve problems with more constraints by introducing more Lagrange multipliers.

Example

$$\max x^2 + y^2 + z^2$$

s.t.

$$x + y - 2 = 0$$
$$x + z - 2 = 0$$

Lagrangian:
$L(x, y, z, p, q) = x^2 + y^2 + z^2 + p(x + y - 2) + q(x + z - 2)$

Solve:

$$\begin{aligned} 0 &= \nabla_x L = 2x + p + q \\ 0 &= \nabla_y L = 2y + p \\ 0 &= \nabla_z L = 2z + q \\ 0 &= \nabla_p L = x + y - 2 \\ 0 &= \nabla_q L = x + z - 2 \end{aligned}$$

Inequality Constraints

In the case of Inequality Constraints of the following type

$$g(\mathbf{x}) \geq 0,$$

two kinds of solution are possible:

- inactive constraint, where $g(\mathbf{x}^*) > 0$;
- active constraint, where $g(\mathbf{x}^*) = 0$.

Inactive Constraint

In the case of an inactive constraint, that is when for $\mathbf{x}^*$

$$g(\mathbf{x}^*) > 0\,,$$

the function $g(\mathbf{x})$ plays no role and the only condition for optimality is

$$\nabla f(\mathbf{x}) = 0$$

This again corresponds to the stationary point of L but with $\lambda = 0$.

Active Constraint

In the case of an active constraint, when for $\mathbf{x}^*$

$$g(\mathbf{x}^*) = 0,$$

the previous analysis holds and:

$\nabla f(\mathbf{x}^*) + \lambda \nabla g(\mathbf{x}^*) = 0.$

Now however, the sign of λ is crucial.

f will be at the **maximum** when the gradient is oriented away from the region $g(x) > 0$.

Therefore $\nabla f(\mathbf{x}^*) = -\lambda \nabla g(\mathbf{x}^*)$ for $\lambda > 0$ (For maximization problems)

Inequality Constraints

For both cases of inequality constraints $\lambda g(x) = 0$.

Thus the solution to maximizing f subject to $g(x) \geq 0$ is obtained by optimizing L w.r.t x, λ under the following conditions (KKT):

$$\begin{aligned} g(x) &\geq 0 \\ \lambda &\geq 0 \\ \lambda g(x) &= 0 \end{aligned}$$

Inequality Constraints

Unfortunately when we add inequality constraints the simple condition

$$\nabla L = 0$$

is neither necessary nor sufficient to guarantee a solution to the constrained optimization problem.