Linear methods of classification

Victor Kitov

Table of contents

Linear discriminant functions

- Classification of two classes ω_1 and ω_2.
- Linear discriminant function:

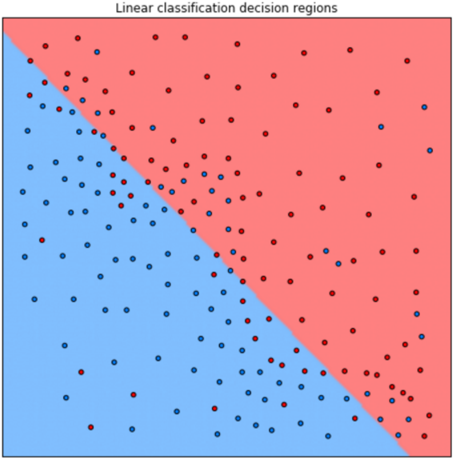
$$g(x) = w^T x + w_0$$

- Decision rule:

$$x \to \begin{cases} \omega_1, & g(x) \geq 0 \\ \omega_2, & g(x) < 0 \end{cases}$$

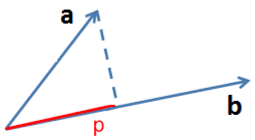
- Decision boundary $B = \{x : g(x) = 0\}$ is linear.

Example: decision regions



Reminder

1. $a=[a^1,...a^D]^T$, $b=[b^1,...b^D]^T$
2. Scalar product $\langle a,b\rangle=a^Tb=\sum_{d=1}^{D}a_d b_b$
3. $a\perp b$ means that $\langle a,b\rangle=0$
4. Norm $\|a\|=\sqrt{\langle a,a\rangle}$
5. Distance $\rho(a,b)=\|a-b\|=\sqrt{\langle a-b,a-b\rangle}$



- $p=\langle a,\frac{b}{\|b\|}\rangle$ - signed projection
- $|p|=\left|a,\frac{b}{\|b\|}\right|$- unsigned length

- Comments:
  - in 2,4 and 5 most commonly used definitions of scalar product, norm and daistace are given
  - in general scalar product, norm and distance may have alternative definitions.

Properties

- Consider arbitrary
$$x_A, x_B \in B \Rightarrow \begin{cases} g(x_A) = w^T x_A + w_0 = 0 \\ g(x_B) = w^T x_B + w_0 = 0 \end{cases}$$
so $w^T(x_A - x_B) = 0$ and $w \perp B$.

Distance form origin

- Distance from the origin to B is equal to absolute value of the projection of $x \in B$ on $\frac{w}{\|w\|}$:

$$\langle x, \frac{w}{\|w\|} \rangle = \frac{\langle x, w \rangle}{\|w\|} = \{w^T x + w_0 = 0\} = -\frac{w_0}{\|w\|}$$

- So $\rho(0, B) = \frac{w_0}{\|w\|}$, and w_0 determines the offset from the origin.

Distance from x to B

Denote $x_\perp$ - the projection of x on B, and $r = \langle \frac{w}{\|w\|}, x - x_\perp \rangle$ - the signed length of the orthogonal complement of x on B:

$$x = x_\perp + r\frac{w}{\|w\|}$$

After multiplication by w and addition of w_0:

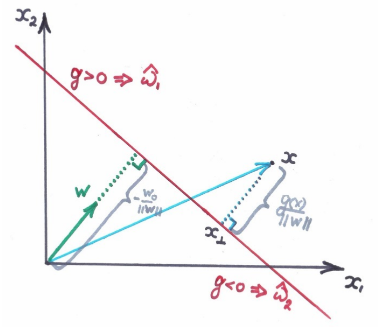
$$w^T x + w_0 = w^T x_\perp + w_0 + r\frac{\langle w, w \rangle}{\|w\|}$$

Using $w^T x + w_0 = g(x)$ and $w^T x_\perp + w_0 = 0$, we obtain:

$$r = \frac{g(x)}{\|w\|}$$

So from one side of the hyperplane $r > 0 \Leftrightarrow g(x) > 0$, and from the other side of the hyperplane $r < 0 \Leftrightarrow g(x) < 0$.

Illustration



Linear decision rule:

$$\widehat{c}(x) = \begin{cases} \omega_1, & g(x) > 0 \\ \omega_2, & g(x) < 0 \end{cases}$$

Decision boundary: $g(x) = 0$, confidence of decision: $|g(x)| / \|w\|$.

Multiple classes classification

- Classification among $\omega_1, \omega_2, ...\omega_C$.
- Use C discriminant functions $g_c(x) = w_c^T x + w_{c0}$
- Decision rule:

$$\widehat{c}(x) = \arg\max_c g_c(x)$$

- Decision boundary between classes ω_i and ω_j is linear:

$$\left(w_i - w_j\right)^T x + \left(w_{i0} - w_{j0}\right) = 0$$

- *Decision regions are convex.*

Table of contents

Linear discriminant functions

- Consider binary classification of classes ω_1 and ω_2.
- Denote classes ω_1 and ω_2 with $y=+1$ and $y=-1$.
- Linear discriminant function: $g(x) = w^T x + w_0$,

$$\widehat{\omega} = \begin{cases} \omega_1, & g(x) \geq 0 \\ \omega_2, & g(x) < 0 \end{cases}$$

- Decision rule: $y = \operatorname{sign} g(x)$.
- Define constant feature $x_0 \equiv 1$, then $g(x) = w^T x = \langle w, x \rangle$ for $w = [w_0, w_1, ..w_D]^T$.
- Define the margin $M(x,y) = g(x)y$
    - $M(x,y) \geq 0$ <=> object x is correctly classified as y
    - $|M(x,y)|$ - confidence of decision

Weights selection

- Target: minimization of the number of misclassifications Q:

$$Q(w|X) = \sum_n \mathbb{I}[M(x_n, y_n|w) < 0] \to \min_w$$

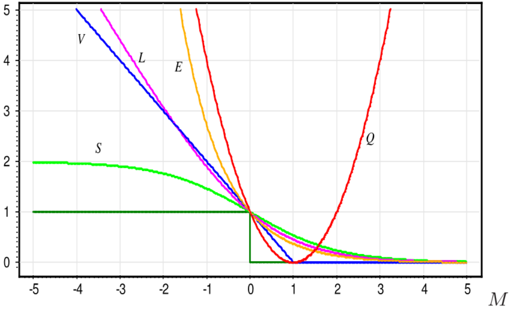
- Problem: standard optimization methods are inapplicable, because $Q(w, X)$ is discontinuous.
- Idea: approximate loss function with smooth function $\mathcal{L}$:

$$\mathbb{I}[M(x_n, y_n|w) < 0] \le \mathcal{L}(M(x_n, y_n|w))$$

Approximation of the target criteria

We obtain the upper boundary on the empirical risk:

$$Q(w|X) = \sum_n \mathbb{I}[M(x_n, y_n|w) < 0]$$
$$\leq \sum_n \mathcal{L}(M(x_n, y_n|w)) = F(w)$$



$$Q(M) = (1-M)^2$$
$$V(M) = (1-M)_+$$
$$S(M) = 2(1+e^M)^{-1}$$
$$L(M) = \log_2(1+e^{-M})$$
$$E(M) = e^{-M}$$

Table of contents

Optimization

- Optimization task to obtain the weights:

$$F(w) \quad = \quad \sum_{i=1}^{N} \mathcal{L}(\langle w, x_i \rangle y_i) \to \min_w$$

- Gradient descend algorithm:

<u>INPUT</u>:
η - parameter, controlling the speed of convergence
stopping rule

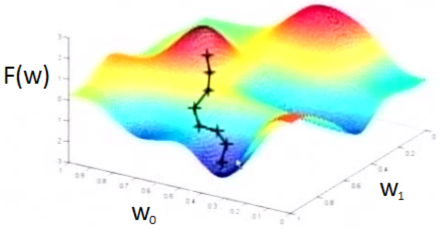
<u>ALGORITHM</u>:
initialize w_0 randomly
while stopping rule is not satisfied:

$w_{n+1} \leftarrow w_n - \eta \frac{\partial F(w_n)}{\partial w}$

$n \leftarrow n + 1$

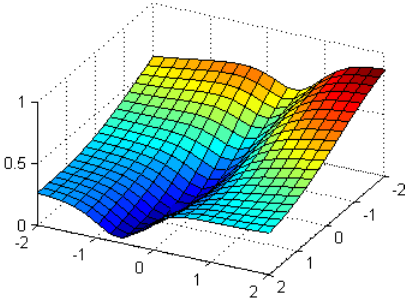
Gradient descend

- Possible stopping rules:
  - $|w_{n+1} - w_n| < \varepsilon$
  - $|F(w_{n+1}) - F(w_n)| < \varepsilon$
  - $n > n_{max}$
- Suboptimal method of minimization in the direction of the greatest reduction of $F(w)$:



Recommendations for use

- Convergence is faster for normalized features
  - feature normalization solves the problem of «elongated valleys»



Convergence acceleration

Stochastic gradient descend method

set the initial approximation w_0
calculate $\widehat{F} = \sum_{i=1}^{n} \mathcal{L}(M(x_i, y_i|w_0))$
iteratively until convergence $\widehat{Q}_{approx}$:

1. select random pair (x_i, y_i)
2. recalculate weights: $w_{n+1} \leftarrow w_n - \eta_n \mathcal{L}'(\langle w_n, x_i \rangle y_i) x_i y_i$
3. estimate the error: $\varepsilon_i = \mathcal{L}(\langle w_{n+1}, x_i \rangle y_i)$
4. recalculate the loss $\widehat{F} = (1-\alpha)\widehat{F} + \alpha \varepsilon_i$
5. $n \leftarrow n + 1$

Variants for selecting initial weights

- $w_0 = w_1 = ... = w_D = 0$
- For logistic $\mathcal{L}$ (because the horizontal asymptotes):
    - randomly on the interval $[-\frac{1}{2D}, \frac{1}{2D}]$
- For other functions $\mathcal{L}$:
    - randomly
- $w_i = \frac{cov[x^i, y]}{var[x^i]}$ (these are regression weights, given that x^i are uncorrelated).

Discussion of SGD

Advantages

- Easy to implement
- Works online
- A small subset of learning objects may be sufficient for accurate estimation

Discussion of SGD

Advantages

- Easy to implement
- Works online
- A small subset of learning objects may be sufficient for accurate estimation

Drawbacks

- Suboptimal - converges to local optimum
- Needs selection of η_n:
  - too big: divergence
  - too small: very slow convergence
- Overfitting possible for large D and small N
- When $\mathcal{L}(u)$ has left horizontal asymptotes (e.g. logistic), the algorithm may «get stuck» for large values of $\langle w, x_i \rangle$.

Examples

Quadratic loss: $\mathcal{L}(M) = (M-1)^2$
Perceptron of Rosenblatt: $\mathcal{L}(M) = [-M]_+$.

Table of contents

Regularization for SGD

- L_2-regularization for upperbound approximation:

$$F^{reg}(w) = F(w) + \lambda \left\| w \right\|^2$$

- SGD weights modification: $w \leftarrow w(1-\eta 2\lambda) - \eta \frac{\partial}{\partial w} F(w)$

Regularization

- Useful technique to control the complexity of the model, can be applied to any algorithm.

$$F^{reg}(w) = F(w) + \lambda R(w)$$

- λ is the parameter controlling strength of regularization = model complexity.
- Examples:

$$\begin{aligned} R(w) &= ||w||_1 = \sum_{d=1}^{D} |w^d| \\ R(w) &= \|w\|_2^2 = \sum_{d=}^{D} \left(w^d\right)^2 \\ R(w) &= \alpha \|w\|_1 + (1-\alpha) \|w\|_2 , \alpha \in [0, 1] \end{aligned}$$

L_1 norm

- $||w||_1$ regularizer will do feature selection.
- Consider

$$Q(w) = F(w) + \lambda \sum_{d=1}^{D} |w_d|$$

- if $\lambda > \sup_w \left| \frac{\partial F(w)}{\partial w_i} \right|$, then it becomes optimal to set $w_i = 0$
- For smaller C more inequalities will become active.

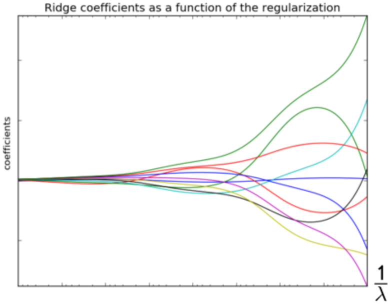
L_2 norm

- $||w||_1$ regularizer will do feature selection.
- Consider $R(w) = \|w\|_2^2 = \sum_d w_d^2$

$$Q(w) = F(w) + \lambda \sum_{d=1}^{D} w_d^2$$

- $\frac{\partial R(w)}{\partial w_i} = 2w_i \to 0$ when $w_i \to 0$.

Illustration



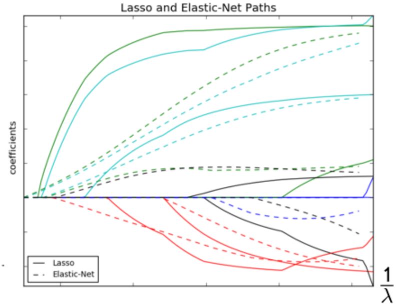


Table of contents

Binary classification

- Linear classifier:

$$score(\omega_1|x) = w^T x$$

- +relationship between score and class probability is assumed:

$$p(\omega_1|x) = \sigma(w^T x)$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ - sigmoid function

Binary classification: estimation

Using the property $1-\sigma(z)=\sigma(-z)$ obtain that

$$p(y=+1|x)=\sigma(w^T x)\Longrightarrow p(y=-1|x)=\sigma(-w^T x)$$

So for $y\in\{+1,-1\}$

$$p(y|x)=\sigma(y\langle w,x\rangle)$$

Therefore ML estimation can be written as:

$$\prod_{i=1}^{N}\sigma(\langle w,x_i\rangle y_i)\to\max_w$$

Loss function for 2-class logistic regression

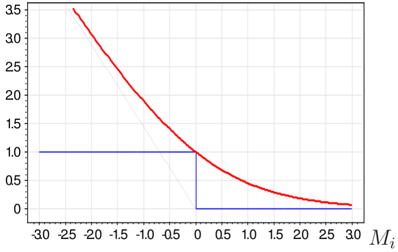
For binary classification $p(y|x) = \sigma(\langle w, x\rangle y)$ $w = [\beta_0', \beta]$, $x = [1, x_1, x_2, ...x_D]$.

Estimation with ML:

$$\prod_{i=1}^{n} \sigma(\langle w, x_i\rangle y_i) \to \max_w$$

which is equivalent to

$$\sum_i^n \ln(1 + e^{-\langle w, x_i\rangle y_i}) \to \min_w$$



It follows that logistic regression is linear discriminant estimated with loss function $\mathcal{L}(M) = \ln(1 + e^{-M})$.

Multiple classes

Multiple class classification:

$$\begin{cases} score(\omega_1|x) = w_1^T x \\ score(\omega_2|x) = w_2^T x \\ \cdots \\ score(\omega_C|x) = w_C^T x \end{cases}$$

+relationship between score and class probability is assumed:

$$p(\omega_c|x) = softmax(w_c^T x | x_1^T x, ... x_C^T x) = \frac{exp(w_c^T x)}{\sum_i exp(w_i^T x)}$$

Multiple classes

Weights ambiguity:

w_c, $c = 1, 2, ...C$ defined up to shift v:

$$\frac{exp((w_c - v)^T x)}{\sum_i exp((w_i - v)^T x)} = \frac{exp(-v^T x) exp(w_c^T x)}{\sum_i exp(-v^T x) exp(w_i^T x)} = \frac{exp(w_c^T x)}{\sum_i exp(w_i^T x)}$$

To remove ambiguity usually $v = w_C$ is subtracted.

Estimation with ML:

$$\begin{cases} \prod_{n=1}^N softmax(w_{y_n}^T x_n | x_1^T x, ... x_C^T x) \to \max_{w_1, ... w_C - 1} \\ w_C = \mathbf{0} \end{cases}$$

Table of contents

Problem statement

- Standard linear classification decision rule

$$\widehat{c} = \begin{cases} 1, & w^T x \geq -w_0 \\ 2, & w^T x < w_0 \end{cases}$$

  is equivalent to

  1. dimensionality reduction to 1-dimensinal space (defined by w)
  2. making classification in this space

- Idea of Fisher's LDA: find direction, giving most discriminative projections.

Possible realization

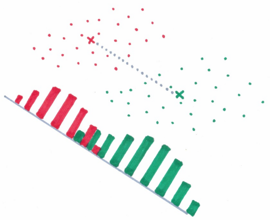
- Classification between ω_1 and ω_2.
- Define $C_1 = \{i : x_i \in \omega_1\}, \quad C_2 = \{i : x_i \in \omega_2\}$ and

$$m_1 = \frac{1}{N_1}\sum_{n\in C_1} x_n, \quad m_2 = \frac{1}{N_1}\sum_{n\in C_2} x_n$$

$$\mu_1 = w^T m_1, \quad \mu_2 = w^T m_2$$

Naive solution:

$$\begin{cases}(\mu_1-\mu_2)^2 \to \max_w \\ \|w\| = 1\end{cases}$$

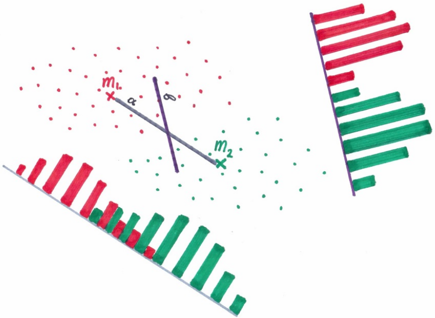


Fisher's LDA

- Define projected within class variances:

$$s_1 = \sum_{n \in C_1} (w^T x_n - w^T m_1)^2, \quad s_2 = \sum_{n \in C_2} (w^T x_n - w^T m_2)^2$$

- Fisher's LDA criterion: $\frac{(\mu_1 - \mu_2)^2}{s_1^2 + s_2^2} \to \max_w$



Equivalent representation

$$
\begin{aligned}
\frac{(\mu_1-\mu_2)^2}{s_1^2+s_2^2} &= \frac{\left(w^T m_1 - w^T m_2\right)^2}{\sum_{n\in C_1}(w^T x_n - w^T m_1)^2 + \sum_{n\in C_2}(w^T x_n - w^T m_2)^2} \\
&= \frac{\left[w^T(m_1-m_2)\right]^2}{\sum_{n\in C_1}\left[w^T(x_n-m_1)\right]^2 + \sum_{n\in C_2}\left[w^T(x_n-m_1)\right]^2} \\
&= \frac{w^T(m_1-m_2)(m_1-m_2)^T w}{w^T\left[\sum_{n\in C_1}(x_n-m_1)(x_n-m_1)^T + \sum_{n\in C_2}(x_n-m_2)(x_n-m_2)^T\right] w} \\
&= \frac{w^T S_B w}{w^T S_W w}
\end{aligned}
$$

$$
\begin{aligned}
S_B &= (m_1-m_2)(m_1-m_2)^T, \\
S_W &= \sum_{n\in C_1}(x_n-m_1)(x_n-m_1)^T + \sum_{n\in C_2}(x_n-m_2)(x_n-m_2)^T
\end{aligned}
$$

Fisher's LDA solution

$Q(w) = \frac{w^T S_B w}{w^T S_W w} \to \max_w$

Using property that $\frac{d}{dw}\left(w^T A w\right) = 2Aw$ for any $A \in \mathbb{R}^{KxK}$, $A^T = A$

$$\frac{dQ(w)}{dw} \propto 2S_B w\left[w^T S_W w\right] - 2\left[w^T S_B w\right] S_W w = 0$$

which is equivalent to

$$\left[w^T S_W w\right] S_B w = \left[w^T S_B w\right] S_W w$$

So

$$w \propto S_W^{-1} S_B w \propto S_W^{-1}(m_1 - m_2)$$