Today → Maximum Margin Classifier | Support vector Machines

Supervised learning

→ max min formulation

→ Constrained formulation

→ lagrangian function (KKT conditions)

→ Solution through the Hinge loss

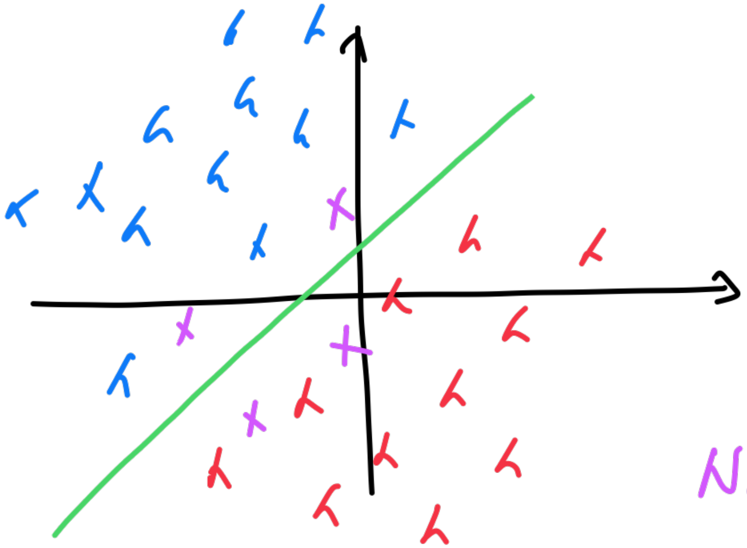
Unsupervised learning

→ Clustering → Combinatorial approaches (including Kmeans/Kmedoid)

→ Bump Hunting approaches (including A priori algorithm)

Maximum Margin Classifier

Question: Given a binary classification problem where should one position our classifier for it to be as robust as possible toward new (unknown) examples?



One idea is to try to position the discriminant right in between the 2 classes.

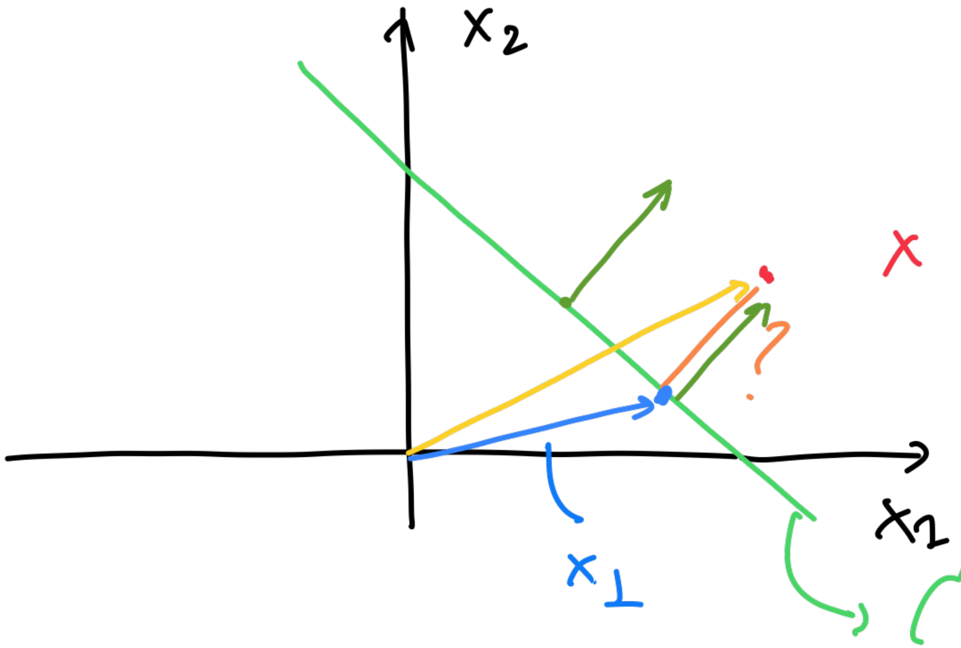
New examples are likely to be "close" to their corresponding classes and such a position is thus likely going to reduce the misclassification risk.

→ Putting the discriminant right in between the 2 classes is equivalent to maximizing the distance between the discriminant and the 2 classes

Which can be done by maximizing the distance between the discriminant and the closest point from each class.

How can one express this distance ?

$$x = x_\perp + r \frac{\vec{\beta}}{\|\beta\|} \qquad (*)$$



For any plane of the form $\underbrace{\beta_2 x_1 + \beta_2 x_2 + \beta_0}_{y(x)} = 0$

the normal to the plane is $[\beta_1, \beta_2]$

$$y(x) = \vec{\beta}^T x + \beta_0 \qquad \vec{\beta} = [\beta_1, \beta_2]$$

substituting (*)

$$y(x) = \vec{\beta}^T x + \beta_0$$

y(x) is prediction from the plane

$$= \vec{\beta}^T x_\perp + \vec{\beta}^T r \frac{\beta^T}{\|\beta\|} + \beta_0$$

$$= \vec{\beta}^T x_\perp + \frac{\|\beta\|^2}{\|\beta\|} r + \beta_0$$

= 0 as $x_\perp$ belongs to the plane $\beta_1 x_1 + \beta_2 x_2 + \beta_0 = 0$

We are then left with

$$y(x) = \|\beta\| r$$

SIGNED DISTANCE

$$\boxed{r = \frac{y(x)}{\|\beta\|}} \quad (*)$$

($y(x)$ can be both positive or negative depending on whether x lies above or below the plane.

if we consider a classification problem where the points above the plane are given a +1 target and the points below are given a -1 target, then the "unsigned" distance of a point x to the plane is given by multiplying r by the target $t^{(i)}$ of x

$$\mathrm{dist}(x, \text{plane}) = r\,t(x) \underbrace{=}_{(*)} \frac{y(x)}{\|\beta\|}\, t(x)$$

Given this expression for the distance, in order to learn a robust classifier, we can just find the points that are the closest to the plane and then "push" the plane as far as possible from those points.

Concretely

$$\beta^*, \beta_0^* = \underbrace{\max_{\beta, \beta_0} \underbrace{\min_{x} \frac{y(x)\, t(x)}{\|\beta\|}}_{\text{finding the distance to the closest point}}}_{\text{push the plane as far as possible away from that point}}$$

⚠ here $\vec{\beta} = (\beta_1, \beta_2)$
(not including β_0)

MAXIMUM MARGIN CLASSIFIER

$$\boxed{\beta^*, \beta_0^* = \underset{\vec{\beta}, \beta_0}{\operatorname{argmax}} \; \min_{x} \frac{(\beta_1 x_1 + \beta_2 x_2 + \beta_0)\, t(x)}{\|(\beta_1, \beta_2)\|}}$$

(**)

Note that for a 2D space (x_1, x_2)

the plane is the set of points (x_1, x_2) satisfying $\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$

which can also read as $x_2 = -\frac{\beta_1 x_1 - \beta_0}{\beta_2}$

(**) is not easy to solve as (1) it involves a ratio of functions in the optimization variables

(2) it involves competing maximization and minimization processes

In order to simplify this formulation we can however use the fact that the ratio that appears in the objective in (**) is invariant under any rescaling of $\vec{\beta}$

letting $\beta = \alpha \beta$, we get $$\frac{(\alpha \beta_1 x_2 + \alpha \beta_2 x_2 + \alpha \beta_0)\, t(x)}{\|(\alpha \beta_1, \alpha \beta_2)\|}$$

$$= \frac{\alpha (\beta_1 x_1 + \beta_2 x_2 + \beta_0)\, t(x)}{\alpha \|(\beta_1, \beta_2)\|}$$

in particular this means we can fix the scaling α and optimize over all vectors β satisfying that fixed scaling

One possible choice is to choose the scaling such that for the closest point to the plane, we get

$$(\beta_0 + \beta_1 x_1 + \beta_2 x_2)\, t(x) = 1$$

In other words, we fix the scaling ambiguity by requiring that for the closest point to the plane, the distance to the plane should be $\frac{1}{\|\beta\|}$ (or equivalently the numerator should be 1)

Following from this choice, all the training examples must then satisfy $(\beta_0 + \beta_1 x_1 + \beta_2 x_2)\, t(x) \geq 1$

Our original formulation now reduces to a constrained formulation (a.k.a canonical formulation)

$$(*) \quad \underset{\vec{\beta},\, \beta_0}{\operatorname{argmax}} \; \boxed{\frac{1}{\|\beta\|}}$$

$$\text{s.t} \quad \underbrace{(\beta_0 + \beta_1 x_2 + \beta_2 x_2)\, t(x) \geq 1}$$

DISTANCE OF CLOSEST POINT TO THE PLANE

CANONICAL FORMULATION

LINEAR CONSTRAINTS (all the training examples must be at a distance larger than $\frac{1}{\|\beta\|}$ (closest distance)

(*) can be further simplified by squaring the $\|\beta\|$ and replacing the maximization with a minimization

$$\beta^*, \beta_0^* = \underset{\vec{\beta},\, \beta_0}{\operatorname{argmin}} \; \|\beta\|^2 \quad (*)$$

$$\text{s.t} \quad (\beta_0 + \beta_1 x_1 + \beta_2 x_2)\, t(x) \geq 1$$

for all x in training

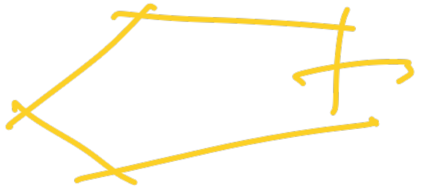
Given (*) we can analyze the solution(s) by relying on the Lagrangian function and the associated Karush-Kuhn Tucker conditions

Given the constrained problem (*) the lagrangian is defined by introducing a set of (lagrange) multipliers (one for each constraint) $\lambda_1, \lambda_2, \dots, \lambda_N \geq 0$ as $\boxed{L}$

$$\mathcal{L}(\vec{\beta}, \beta_0, \lambda) = \|\beta\|^2 - \sum_{i=2}^{N} \lambda_i \Big(\overbrace{(\beta_0 + \beta_1 x_2^{(i)} + \beta_2 x_2^{(i)}) t(x)}^{\geq 0} - 1 \Big)$$

Under appropriate conditions our $\underbrace{\text{Original}}_{\|\beta\|^2}$ constrained formulation (*) is equivalent to

$$\underset{\vec{\beta}, \beta_0}{\text{argmin}} \; \underset{\lambda}{\max} \; \mathcal{L}(\vec{\beta}, \beta_0, \lambda)$$



feasible set

Ⓐ For any β which violate at least one of the constraints we have $((\beta_0+\beta_1 x_1^{(i)}+\beta_2 x_2^{(i)})t(x)-1)<0$

and hence $-\lambda_i((\beta_0+\beta_1 x_1^{(i)}+\beta_2 x_2^{(i)})t(x)-1)\geq 0$

and can be increased to $+\infty$ by growing λ_i

in this Ⓐ setting we thus have $\max\limits_{\lambda} \mathcal{L}(\vec{\beta},\beta_0,\lambda)=+\infty$

Ⓑ On the contrary, for a β that satisfies all the constraints (i.e which lies inside the feasible set) we have

$$((\beta_0+\beta_1 x_1^{(i)}+\beta_2 x_2^{(i)})t(x)-1)>0$$

hence $-\lambda_i((\beta_0+\beta_1 x_1^{(i)}+\beta_2 x_2^{(i)})t(x)-1)<0$

and the best the max over λ_i can do is to bring that quantity to zero in this Ⓑ setting, we thus have

$$\max_{\lambda} \mathcal{L}(\vec{\beta}, \beta_0, \lambda) = \|\beta\|^2$$

Grouping Ⓐ + Ⓑ we see that

$$\arg\min_{\vec{\beta}, \beta_0} \max_{\lambda_i} \mathcal{L}(\vec{\beta}, \beta_0, \lambda) = \arg\min_{\vec{\beta}, \beta_0} \begin{cases} +\infty & \text{if } \vec{\beta}, \beta_0 \text{ are not feasible} \\ \|\beta\|^2 & \text{if } (\beta_0 + \beta_1 x_1^{(i)} + \beta_2 x_2^{(i)})\, t(x^{(i)}) \geq 1 \quad \forall\, x^{(i)} \end{cases}$$

which is exactly equivalent to our original (i.e constrained) formulation

Now that we have the lagrangian function we can turn to the following theorem (known as the Karush-Kuhn-Tucker conditions) which connect the solutions of the constrained problem with the lagrangian function $\mathcal{L}$.

Theorem (KKT conditions for linearly constrained problems)

Consider a constrained problem

$$(*) \begin{cases} \min \ f(\beta) \\ \text{s.t} \quad a_i^T \beta \leq b_i \quad i = 1, \dots, N \end{cases}$$

Assume that $f(\beta)$ is convex and continuously differentiable over $\mathbb{R}^{D+1}$. Let β^* be a feasible point for $(*)$ (i.e β^* satisfies $a_i^T \beta^* \leq b_i \ \forall i$)

then β^* is an optimal solution of the constrained problem $(*)$ if and only if there exists a set of multiplers $\lambda_1^*, \dots, \lambda_N^* \geq 0$ such that

$$\nabla f(\beta^*) + \sum_{i=1}^{N} \lambda_i^* \vec{a}_i = 0 \quad \text{and} \quad \lambda_i^*(a_i^T \beta - b_i) = 0 \quad \forall\, i \in \{1, \dots, N\}$$

using the KKT conditions above, and applying these conditions to $(*)$ we find that our classifier has to satisfy the equation

$$\frac{\partial L}{\partial \beta_1} = 2\beta_1 - \sum_{i=2}^{N} \lambda_i^* t(x^{(i)}) x_1^{(i)} = 0 \quad *$$

$$\frac{\partial L}{\partial \beta_2} = 2\beta_2 - \sum_{i=1}^{N} \lambda_i^* t(x^{(i)}) x_2^{(i)} = 0 \quad **$$

Compactly we thus have $\nabla_{(\beta_1,\beta_2)} L = 2\beta - \sum_{i=1}^{N} \lambda_i^* \vec{x}^{(i)} t(x^{(i)})$

Doing the same for β_0 (which does not appear in $\|\beta\|^2$)

we get

$$\boxed{\frac{\partial L}{\partial \beta_0} = -\sum_{i=1}^{N} \lambda_i^* t(x^{(i)}).1 = 0 \quad (***)}$$

Solving (*), (**) & (***) for β_1, β_2 and substituting in the lagrangian Ⓛ we get a formulation (maximization problem) that only depends on the λ_i's known as the dual (as opposed to the primal formulation (*))

this formulation is a quadratic optimization problem

Both the primal and dual formulations can be solved efficiently (e.g through CVX see recitation).