SUPPORT VECTOR MACHINES

Sergey Nikolenko

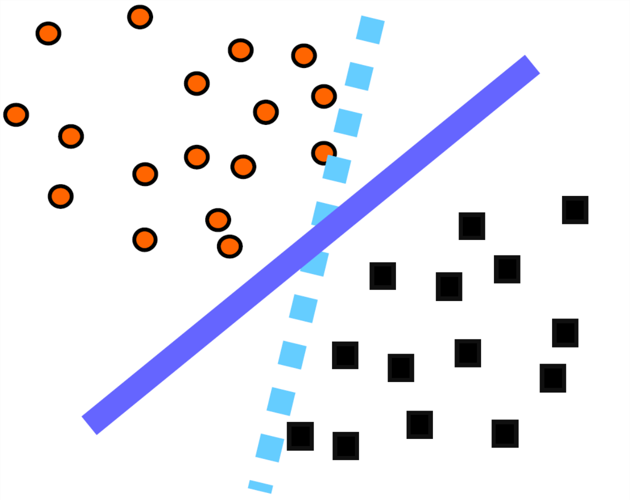
Harbour Space University, Barcelona, Spain
March 21, 2017

SVM FOR LINEAR CLASSIFICATION

- Support vector machines solve the classification problem.
- Again, each data point lies in the n–dimensional space $\mathbb{R}^n$.
- Formally, we have points x_i, $i = 1..m$, and points have labels $y_i = \pm 1$.
- The question is to separate the data with an $(n-1)$–dimensional hyperplane and find that hyperplane.
- Is that all?

- Not quite; we also want to separate with this hyperplane *as well as possible*.
- I.e., the two separated classes should be as far as possible from the hyperplane.
- Practical too: then small disturbances in the hyperplane won't gurt anything.

EXAMPLE



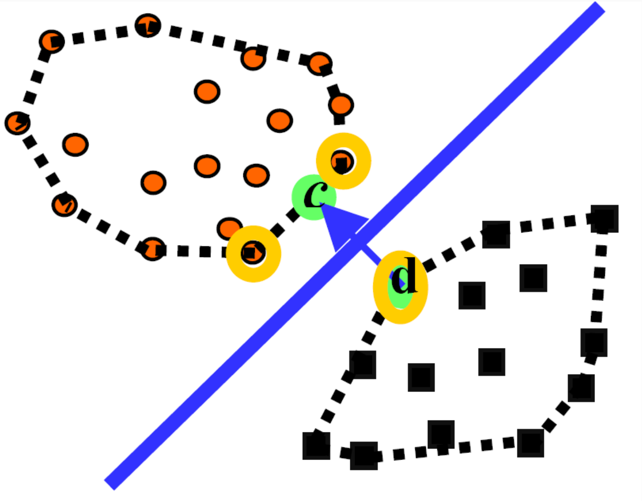
- Find two points in convex hulls of the data and use the perpendicular bisector between them.
- Formally this turns into a quadratic optimization problem:

$$\min_{\alpha}\left\{ ||c-d||^2, \text{ where } c=\sum_{y_i=1}\alpha_i x_i, d=\sum_{y_i=-1}\alpha_i x_i \right\}$$

$$\text{given that } \sum_{y_i=1}\alpha_i = \sum_{y_i=-1}\alpha_i = 1, \alpha_i \geq 0.$$

- We can solve this with general optimization algorithms.

EXAMPLE



- Or we could maximize the *margin* between two parallel support planes and then use the parallel one in the middle.
- A *support* hyperplane for a set of points X is a hyperplane such that all points from X lie on the same side of this hyperplane.
- Formally speaking, the distance from point $\mathbf{x}$ to hyperplane $y(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + w_0 = 0$ equals $\frac{|y(\mathbf{x})|}{\|\mathbf{w}\|}$.

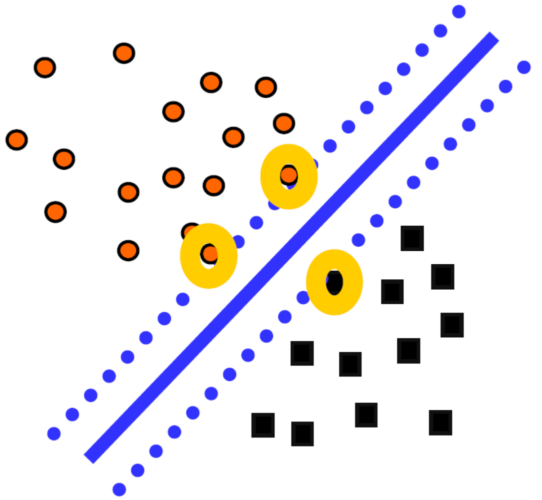
- Distance from point $\mathbf{x}$ to hyperplane $y(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + w_0 = 0$ equals $\frac{|y(\mathbf{x})|}{\|\mathbf{w}\|}$.
- All points have correct classification: $t_n y(\mathbf{x}_n) > 0$ $(t_n \in \{-1, 1\})$.
- And we want to find

$$\arg\max_{\mathbf{w}, w_0} \min_n \frac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|} = \\ = \arg\max_{\mathbf{w}, w_0} \left\{ \frac{1}{\|\mathbf{w}\|} \min_n \left[t_n (\mathbf{w}^\top \mathbf{x}_n + w_0) \right] \right\}.$$

MAXIMIZING THE MARGIN

- $\arg\max_{\mathbf{w},w_0} \left\{ \frac{1}{\|\mathbf{w}\|} \min_n \left[t_n(\mathbf{w}^\top \mathbf{x}_n + w_0) \right] \right\}$. That's hard!
- But we can renormalize $\mathbf{w}$!
- Let's renormalize so that $\min_n \left[t_n(\mathbf{w}^\top \mathbf{x}_n + w_0) \right] = 1$.

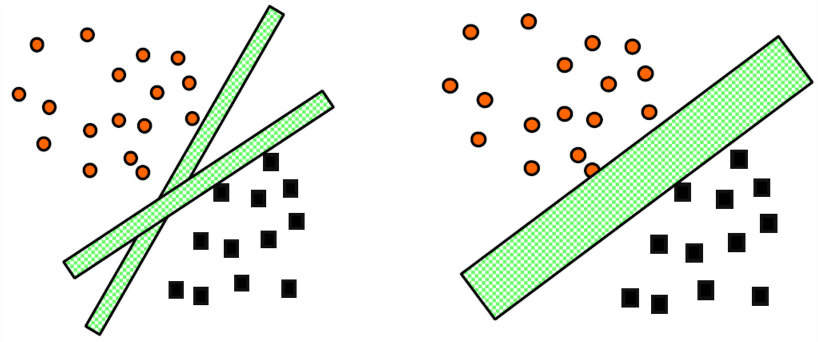
EXAMPLE



- We also get a quadratic programming problem:

$$\min_{\vec{w},b}\left\{\frac{1}{2}||\mathbf{w}||^2\right\} \text{ given that } t_n(\mathbf{w}^\top\mathbf{x}_n + w_0) \geq 1.$$

- We get good results. SVMs often find *stable* solutions, which solves overfitting to a large extent and leads to better predictions.
- In a sense, solutions with “thick” hyperplanes between the data contain more information than “thin” ones because there are fewer “thick” ones.



- Let's recall what dual problems are.
- Direct optimization problem:

$$\min \{f(x)\} \text{ given that } h(x) = 0,\ g(x) \leq 0,\ x \in X.$$

- For the dual problem we introduce parameters λ corresponding to equalities; μ, to inequalities.

- Direct optimization problem:

$$\min\left\{f(x)\right\} \text{ given that } h(x)=0,\ g(x)\leq 0,\ x\in X.$$

- Dual optimization problem:

$$\begin{aligned}\min\left\{\phi(\lambda,\mu)\right\} &\text{ given that } \mu\geq 0,\\ &\text{where } \phi(\lambda,\mu)=\inf_{x\in X}\left\{f(x)+\lambda^\top h(x)+\mu^\top g(x)\right\}.\end{aligned}$$

- Then, if $(\bar{\lambda}, \bar{\mu})$ is an admissible solution of the dual problem, and $\bar{x}$ is an admissible solution of the direct problem, then

$$\phi(\bar{\lambda}, \bar{\mu}) = \inf_{x \in X} \left\{ f(x) + \bar{\lambda}^\top h(x) + \bar{\mu}^\top g(x) \right\} \leq \\ \leq f(\bar{x}) + \bar{\lambda}^\top h(\bar{x}) + \bar{\mu}^\top g(\bar{x}) \leq f(\bar{x}).$$

- This is called *weak duality* (only $\leq$), but equality also holds in many cases.

- For linear programming the direct problem is

$$\min c^\top x \text{ given that } Ax = b,\ x \in X = \{x \leq 0\}.$$

- Then the dual problem is

$$\begin{aligned}\phi(\lambda) = \inf_{x\geq 0} \{c^\top x + \lambda^\top(b - Ax)\} = \\ = \lambda^\top b + \inf_{x \geq 0} \{(c^\top - \lambda^\top A)x\} = \\ = \begin{cases} \lambda^\top b, & \text{if } c^\top - \lambda^\top A \geq 0, \\ -\infty & \text{otherwise.} \end{cases}\end{aligned}$$

- For linear programming the direct problem is

$$\min\{c^\top x\} \text{ given that } Ax = b,\ x \in X = \{x \leq 0\}.$$

- Dual problem:

$$\max\{b^\top \lambda\} \text{ given that } A^\top \lambda \leq c,\ \lambda \text{ are unbounded.}$$

- For quadratic programming the direct problem is

$$\min\left\{\frac{1}{2}x^\top Qx + c^\top x\right\} \text{ given that } Ax \le b,$$

  where Q is a positive semidefinite matrix (i.e., $x^\top Qx \ge 0$ for every x).
- Dual problem (check!):

$$\max\left\{\frac{1}{2}\mu^\top D\mu + \mu^\top d - \frac{1}{2}c^\top Q^{-1}c\right\} \text{ given that } c \ge 0,$$

  where $D = -AQ^{-1}A^\top$ (negative definite matrix),
  $d = -b - AQ^{-1}c.$

- In the case of SVMs we introduce Lagrange multipliers:

$$L(\mathbf{w}, w_0, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_n \alpha_n \left[t_n(\mathbf{w}^\top \mathbf{x}_n + w_0) - 1\right], \quad \alpha_n \geq 0.$$

- Taking derivatives w.r.t. $\mathbf{w}$ and w_0, we equate to zero and get

$$\mathbf{w} = \sum_n \alpha_n t_n \mathbf{x}_n,$$
$$0 = \sum_n \alpha_n t_n.$$

- Substituting into $L(\mathbf{w}, w_0, \alpha)$, we get

$$L(\alpha) = \sum_n \alpha_n - \frac{1}{2} \sum_n \sum_m \alpha_n \alpha_m t_n t_m \left(\mathbf{x}_n^\top \mathbf{x}_m\right)$$

$$\text{given that } \alpha_n \geq 0, \sum_n \alpha_n t_n = 0.$$

- This is the dual problem which we use in SVMs.

- For prediction we look at the sign of $y(\mathbf{x})$:

$$y(\mathbf{x}) = \sum_{n=1}^{N} \alpha_n t_n \mathbf{x}^\top \mathbf{x}_n + w_0.$$

- So the predictions depend on all points $\mathbf{x}_n$?!..

- …not. :) KKT (Karush–Kuhn–Tucker) conditions:

$$\begin{aligned} \alpha_n &\geq 0, \\ t_n y(\mathbf{x}_n) - 1 &\geq 0, \\ \alpha_n \left(t_n y(\mathbf{x}_n) - 1 \right) &= 0. \end{aligned}$$

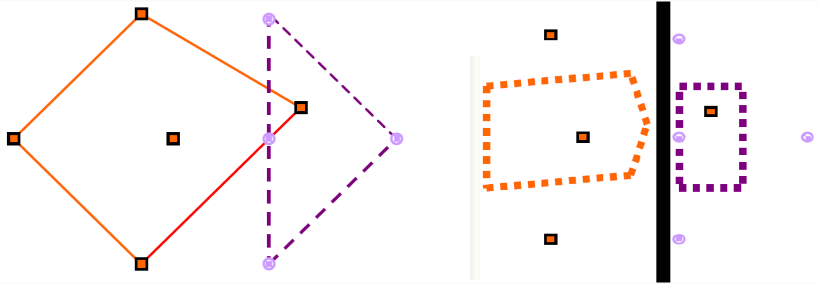
- i.e., the actual prediction depends on a small number of *support* vectors for which $t_n y(\mathbf{x}_n) = 1$ (they are exactly at the boundary of the separating surface).

- All these methods work when the data are actually linearly separable.
- What if they are not? At least a little?
- First question: what do we do in the first approach, with convex hulls?

- We can consider *reduced* convex hulls where coefficients are bounded stronger than just by 1:

$$c = \sum_{y_i=1} \alpha_i x_i, \quad 0 \le \alpha_i \le D.$$

- Then for sufficiently small D reduced convex hulls will be disjoint, and we can find the optimal hyperplane between them.
- The reductions are much stronger around single outliers than in dense regions.



FOR THE SUPPORT VECTORS

- For the support vectors we also have to change something. What exactly?

FOR THE SUPPORT VECTORS

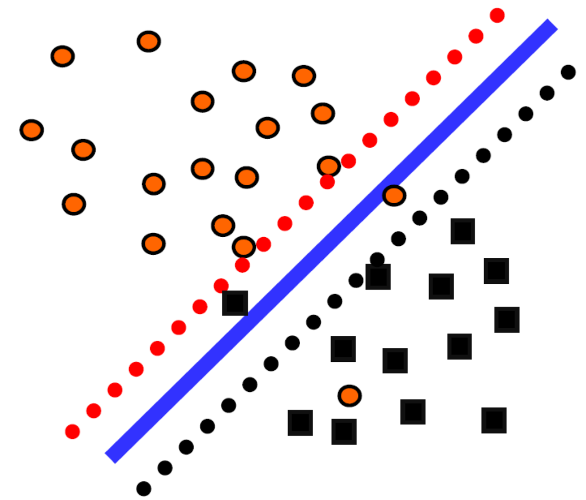
- For the support vectors we also have to change something. What exactly?
- We add to the objective function a nonnegative new error term, called *slack*:

$$\min_{\vec{w},w_0} \left\{ ||\vec{w}||^2 + C \sum_{i=1}^{m} z_i \right\}$$

given that $t_i(\vec{w} \cdot \vec{x}_i - w_0) + z_i \geq 1$.

- This is the direct problem…

EXAMPLE



- ...and this is the dual:

$$\min_{\alpha}\left\{\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m}t_i t_j \alpha_i \alpha_j \left(\vec{x}_i\cdot\vec{x}_j\right)-\sum_{i=1}^{m}\alpha_i, \right.$$

$$\left. \text{where } \sum_{i=1}^{m}t_i\alpha_i=0, \quad 0\le\alpha_i\le C.\right\}$$

- This is most often used in SVM theory.
- The only difference from the linearly separable case is the upper bound C on α_j, i.e., on the influence of every point.

- Support vector machines are a great fit for linear classification.
- But we need to solve a quadratic programming problem, which may be prohibitively complex.
- Practical note: usually requires normalization/whitening of the data.

- Another look at SVM: what is the basic classification problem?
- The goal is to minimize empirical risk, i.e., the number of wrong answers:

$$\sum_{n} [y_i \neq t_i] \to \min_{\mathbf{w}}.$$

- And if the function is linear with parameters $\mathbf{w}$, w_0, it is equivalent to

$$\sum_{n} \left[t_i \left(\mathbf{x}_n^\top \mathbf{w} - w_0\right) < 0\right] \to \min_{\mathbf{w}}.$$

- We call the value $M_i = \mathbf{x}_n^\top \mathbf{w} - w_0$ the *margin*.
- Hard to optimize directly...

- ...so we replace with an upper bound:

$$\sum_n [M_i < 0] \leq \sum_n (1 - M_i) \to \min_{\mathbf{w}} .$$

- And add a regularizer for stability:

$$\sum_n [M_i < 0] \leq \sum_n (1 - M_i) + \frac{1}{2C} \|\mathbf{w}\|^2 \to \min_{\mathbf{w}} .$$

- Et voila: we've got the SVM problem again!

Thank you for your attention!