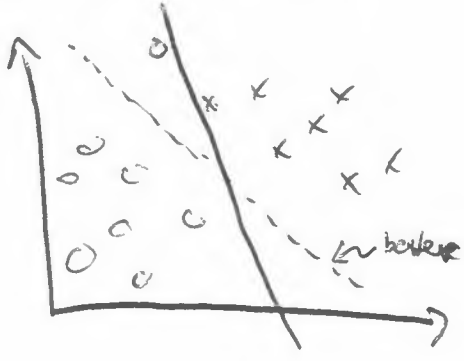
- SVM (Support Vector Machines)

- finds separating, "soft", hyperplane w/ largest margin

- can kernelize to make non-linear

- typically used for classification, $y = \{-1, 1\}$, but can also formulate as regression

soft margin fixes the non-linearly separable cases, and the case where 1 or 2 errant training points drastically change the decision boundary



The problem is posed as a convex optimization problem:

$$\max_{w, b, \xi, M} \; M \quad \leftarrow M = \text{margin}$$

$$\text{s.t.} \quad \|W\|_2^2 = 1 \quad \leftarrow \text{ensures that } b + w^T x^{(i)} \text{ gives signed distance to hyperplane}$$

$$\overbrace{y^{(i)}(b + w^T x^{(i)})}^{\leftarrow \text{signed distance from plane * class}} \geq M(1 - \xi_i) \qquad \forall \; i = 1, \ldots, m$$

$\leftarrow$ ensures correct classification for violations of margin

$\leftarrow$ allows for slack

$$\xi_i \geq 0 \qquad \forall \; i = 1, \ldots, m$$

$$\sum_{i=1}^{m} \xi_i \leq C$$

$\leftarrow$ "soft" budget

- actual optimization objective is a bit different
- can also rephrase w/ ERM using the hinge loss (no penalty for correct prediction, linear penalty for incorrect score) w/ l_2 regularization

- After working out the dual form of the problem, we notice that the entire objective function, as well as the prediction function can be written in terms of inner products of the form: $\langle x^{(i)}, x^{(j)} \rangle$ $(= x^{(i)T} x^{(j)})$
thus, for more model expressiveness we can use a feature map $\phi: \mathbb{R}^n \to \mathbb{R}^d$ $(d > n)$
so that all occurences of $\langle x^{(i)}, x^{(j)} \rangle$ become $\langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$.

Kernel Trick

- given a function $K: \mathbb{R}^n \times \mathbb{R}^n \to [0, \infty)$ (~~$\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$~~), if $\exists$ a feature map, $\phi: \mathbb{R}^n \to \mathbb{R}^d$, s.t. $K(x,y) = \langle \phi(x), \phi(y) \rangle_{\mathbb{R}^d}$
$\forall\ x, y \in \mathbb{R}^n$, then K is said to be a valid kernel.
- thus we can replace all occurences of $\langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$ w/ $K_{ij} = K(x^{(i)}, x^{(j)})$
- this has space + time savings
  - don't need to explicitly store $\phi(x^{(i)})$ (which could be infinitely dimensional)
  - evaluating $K(x^{(i)}, x^{(j)})$ is also typically $O(n)$ time, whereas computing $\langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$

is much worse

- Can kernelize all ERM objectives of the form:

$$R(\|w\|) + \sum_{i=1}^{m} L(y^{(i)}, w^T x^{(i)})$$

(Representer thm) (R is a nondecreasing function)

- Can obtain valid kernel functions with Mercer theorem.

Pros

- mathematically appealing, and is clear why it works well (i.e., has low generalization error) as compared to say the GBM algo
- memory efficient and predictions are fast, since prediction only depends on $\alpha^{(i)}$'s of "support vectors"

Cons

- doesn't provide probability estimates (only scores)
- may take long to train w/ lots of data since entire $m \times m$ kernel matrix must be computed