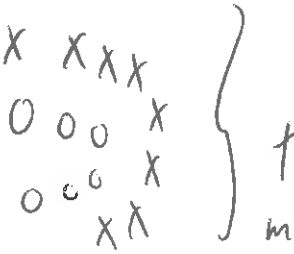
Lecture #20: November 27, 2018

- How can we do SVM for non-linearly boundaries?



No hyperplane does a very good job of separating the classes... if we could use a curved boundary, then maybe things would be possible

- Recall the Kernel trick from our discussion of kernel PCA: we want to find a feature map $\phi : \mathbb{R}^d \to \mathbb{R}^D$ that discriminates in the way we want, i.e. separates the classes.

ex: For example above, $\phi(x,y) = (x, y, x-y^2)$ would probably work, since the two classes live on different parabolas.

- If D is big, ie. we need to embed in a large number of dimensions, things may be slow computationally. To get around this, we never consider ϕ explicitly but rather consider a kernel $K(x,y) = \phi(x)^T \phi(y)$, which allows to compute the distances according to ϕ as:

exercise: $\|\phi(x) - \phi(y)\|_2 = \sqrt{K(x,x) - 2K(x,y) + K(y,y)}$

The kernel $K(x,y) = \phi(x)^T \phi(y)$ is <u>symmetric</u> ($K(x,y) = K(y,x)$) and <u>non-negative</u> ($K(x,y) \geq 0$ for all x). and <u>positive semidefinite</u> (

$\sum_{i,j=1}^{n} \alpha_i \alpha_j K(x_i, x_j) \geq 0$ for all $\{\alpha_i\}_{i=1}^n \subset \mathbb{R}$, $\{x_i\}_{i=1}^n \subset \mathbb{R}^d$)

<u>Exercise</u>: Positive semidefiniteness as above is related to matrix positive semidefiniteness as

K is psd $\Leftrightarrow$ the matrix $\left(K(x_i, x_j)\right)_{i,j=1,\dots,n}$ is psd as a matrix.

- In fact, these properties completely characterize kernels!

<u>Theorem (Moore-Aronszajn)</u>: Let $K: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be symmetric and psd. Then $K(x,y) = \langle \phi(x), \phi(y) \rangle$ for some $\phi: \mathbb{R}^d \to \mathcal{F}$, where $\mathcal{F}$ is a (possibly infinite dimensional) feature space.

- Common kernel is the Gaussian kernel $K(x,y) = \exp(-\|x-y\|_2^2/\sigma^2)$, some $\sigma > 0$.

<u>Remark</u>: The $\mathcal{F}$ in the MAT may be infinite dimensional, so we don't want to think about writing out ϕ explicitly. Better to use the kernel trick and only touch K.

So, how to incorporate kernels into SVM? Let's replace x_i with $\phi(x_i)$, and see if we can ultimately bring the kernel $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ into the picture.

Hard Margin SVM: original space | feature space

$$(w^*, b^*) = \arg\min_w \|w\|_2^2 \text{ s.t. } y_i(w^T x_i + b) \geq 0, \; i = 1, \dots, n$$

$$(w^*, b^*) = \arg\min_w \|w\|_2^2 \text{ s.t. } y_i(w^T \phi(x_i) + b) \geq 1, \; i = 1, \dots, n \quad (\star)$$

Exercise: If (w^*, b^*) solve $(\star)$, then $w^* = \sum_{i=1}^n \alpha_i \phi(x_i)$, some $\{\alpha_i\}_{i=1}^n \subset \mathbb{R}$.

Hence, $(\star)$ may be re-cast as

$$(\alpha_1^*, \dots, \alpha_n^*, b^*) = \arg\min_{(\alpha_1, \dots, \alpha_n, b)} \left\| \sum_{j=1}^n \alpha_j \phi(x_j) \right\|_2^2 \text{ s.t.}$$

$(\star\star)$

$$y_i\left(w^T \left[\sum_{j=1}^n \alpha_j \phi(x_j)\right] \phi(x_i) + b\right) \geq 1, \; i = 1, \dots, n.$$

Let $K = (K(x_i, x_j))_{i,j=1,\dots,n} \in \mathbb{R}^{n \times n}$ and $\alpha = (\alpha_1, \dots, \alpha_n) \in \mathbb{R}^{n \times 1}$. Then we may write $(\star\star)$ compactly as

$$(\alpha_1^*, \dots, \alpha_n^*, b^*) = (\alpha^*, b) = \arg\min_{(\alpha, b)} \alpha^T K \alpha \text{ s.t.}$$

$$-(K\alpha)_i - b \leq -1, \; y_i = 1$$

$$-(K\alpha)_i + b \leq -1 \text{ if } y_i = -1.$$

- Once the $\{\alpha_i\}_{i=1}^{\hat{n}}, b$ are learned, new points $\hat{x}$ (call it x) are labeled as

1 if $\sum_{i=1}^{\hat{n}} \alpha_i K(x_i, x) + b > 0$

-1 if $\sum_{i=1}^{\hat{n}} \alpha_i K(x_i, x) + b < 0$.

//

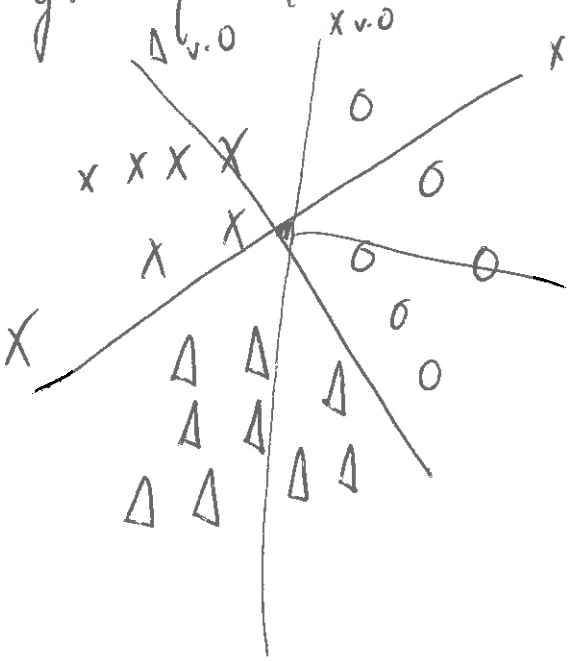
- What if we have multiple classes we want to separate with SVM?

- Natural approach: for each of the K classes, train a classifier against each other class. This is the so-called <u>one-versus-one</u> SVM.

- More precisely, let $k_1, k_2 \in \{1, \dots, K\}$ be distinct labels. Build a classifier SVM by handling the two class problem with data $\{x_i \mid g_i = k_1 \text{ or } k_2\}$.

- Problem: hard to get unique labels for new points:

ex (3 class)



what to do about points that are here?

- An alternative approach is to do a one-versus-all classification scheme for each class, then give a point the label that maximizes the margin among the K one versus all classifiers.

- More precisely, for each $k \in \{1, \dots, K\}$, let $C_1 = \{x_i \mid y_i = k\}$
$C_2 = \{x_i \mid y_i \neq k\}$.

Train an SVM to distinguish C_1 from C_2, and let w_k, b_k be the corresponding hyperplane parameters. This yields a family of SVM with parameters $\{(w_k, b_k)\}_{k=1}^{K}$.

- One then chooses the class that maximizes margin:

$$y(x) = \arg\max_{k=1,\dots,K} w_k^T x + b_k.$$

- Of course, this would all work for kernel SVM as well.

- So far, we have seen two ways to approach classification:
  1.) K-NN algorithms focus on local similarities, but may be poor if local similarities are not enough and global structure is needed.

2.7 SVM: fit a hyperplane or warped hyperplane between classes.

- These methods are rather principled, easy to understand in terms of decisions made, and rely on a small number of parameters.
- We will end the course with neural networks / deep learning, which are harder to understand and have a lot of parameters, but can work amazingly well.