Statistical Learning V

1. Support Vector Machines
   a) Optimal separating hyperplane
   b) SVMs
   c) Interpretation as penalized ERM with hinge loss
   d) Dual Formulation
   e) Kernel trick
   f) Derivation of dual formulation

Unsupervised Learning:
2. K-Means
3. EM for Gaussian Mixtures
(studying 3 is optional)

Vapnik & Chervonenkis: foundational work on statistical learning starting in sixties and seventies, leading to SVMs ~1990.

Vapnik, 1998: "solve the problem directly and never solve a more general problem as an intermediate step"

1. SVMs

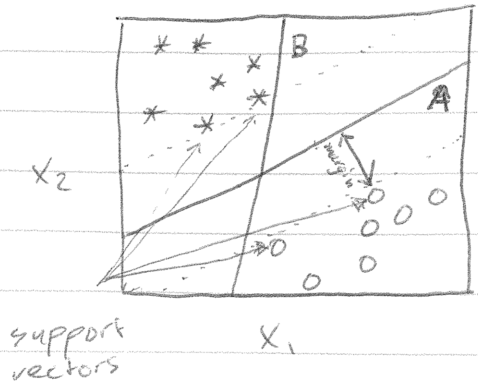
a) Optimal separating hyperplane

	estimate
generative:	$P(X,Y)$
discriminative:	$P(Y\mid X)$
now:	decision boundary directly

Assume: 2 classes, linearly separable

$$\binom{Y_1}{X_1}, \dots, \binom{Y_N}{X_N} \qquad Y \in \{-1, +1\}$$

Linear Model: classifiers that compute $x^T\beta + \beta_0$ ← writing intercept separately
and return its sign



Which decision boundary do you like best? A or B?

A has larger "margin" = distance to nearest point

Fig. 12.1, left

"Optimal" separating hyperplane maximizes the margin.

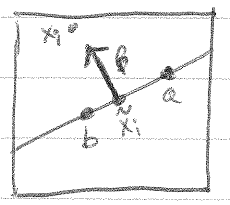
<u>support vectors</u>: points <u>on</u> the margin

decision boundary: $\{x : x^T\beta + \beta_0 = 0\}$

Let $f(x) = x^T\beta + \beta_0$. Then distance of x_i to decision boundary is $\frac{|f(x_i)|}{\|\beta\|}$

<u>margin</u>: $\frac{y_i \cdot f(x_i)}{\|\beta\|}$ is distance if x_i classified correctly, is negative distance if classified incorrectly

<u>Optional</u>: for any a, b on decision boundary:



$(b-a)^T\beta = 0$, so $\frac{\beta}{\|\beta\|}$ is normal vector to the decision boundary of length 1.

suppose $\hat{x}_i$ is projection of x_i onto decision boundary. Then

$$x_i = \hat{x}_i + \alpha \cdot \frac{\beta}{\|\beta\|}$$

$|\alpha|$ is distance of x_i from decision boundary

$$\alpha \frac{\beta}{\|\beta\|} = x_i - \hat{x}_i$$

$$\alpha \cdot \|\beta\| = (x_i - \hat{x}_i)^T\beta \quad \text{(because } \beta^T\beta = \|\beta\|^2\text{)}$$

$$= x_i^T\beta - \hat{x}_i^T\beta$$

$$= x_i^T\beta + \beta_0 \quad \text{(because } \hat{x}_i \text{ on decision boundary)}$$

$$= f(x_i)$$

$$\alpha = \frac{f(x_i)}{\|\beta\|}$$

to maximize the margin M:

$$\max_{\beta, \beta_0, M} \quad M$$

subject to: $\dfrac{y_i (x_i^T \beta + \beta_0)}{\|\beta\|} \geq M$ for $i = 1, \dots, N$

margin of $\binom{y_i}{x_i}$

Same decision boundary and margin if we multiply β, β_0 by a constant, so can always choose this constant such that $\|\beta\| = \frac{1}{M}$:

$$\max_{\substack{\beta, \beta_0, M \\ \|\beta\| = \frac{1}{M}}} \quad M$$

s.t. $\dfrac{y_i (x_i^T \beta + \beta_0)}{\cancel{\|\beta\|}} \geq M \cdot \|\beta\| \quad \forall i$

$$\max_{\beta, \beta_0} \quad \frac{1}{\|\beta\|}$$

s.t. $y_i (x_i^T \beta + \beta_0) \geq 1 \quad \forall i$

solution achieved by same parameters β, β_0

convex function $\rightarrow$ $\min_{\beta, \beta_0} \frac{1}{2} \|\beta\|^2$

linear inequality constraints $\rightarrow$ s.t. $y_i (x_i^T \beta + \beta_0) \geq 1 \quad \forall i$

b) SVMs

What if classes are <u>not</u> linearly separable?

Greek letter "xi"

Fig. 12.1, right: introduce slack variable $\xi_i \geq 0$ for each data point $i = 1, \ldots, N$

$$\max_{\beta, \beta_0, M, \xi_i} \quad M$$

subject to: $\xi_i \geq 0, \quad \dfrac{y_i (x_i^T \beta + \beta_0)}{\|\beta\|} \geq M(1 - \xi_i) \quad i = 1, \ldots, N$

$$\sum_{i=1}^{N} \xi_i \leq t$$

parameter of alg. ← Assume we take large enough to have a solution

<u>support vectors</u>: all points <u>inside</u> the margin

$$\min_{\beta, \beta_0, \xi_i} \quad \tfrac{1}{2}\|\beta\|^2$$

s.t. $\quad \xi_i \geq 0, \quad y_i (x_i^T \beta + \beta_0) \geq 1 - \xi_i \quad i = 1, \ldots, N$

$$\sum_{i=1}^{N} \xi_i \leq t$$

equivalent

$$\min_{\beta, \beta_0, \xi_i} \quad \tfrac{1}{2}\|\beta\|^2 + \underset{\text{Parameter}}{C} \cdot \sum_{i=1}^{N} \xi_i \qquad (*)$$

s.t. $\quad \xi_i \geq 0, \quad y_i (x_i^T \beta + \beta_0) \geq 1 - \xi_i \quad i = 1, \ldots, N$

c) Interpretation as penalized ERM

see slides 4

$L(y_i, f(x_i)) = \max\{0, 1 - y_i \cdot f(x_i)\}$ is "hinge loss"

$$\min_{\beta,\beta_0,\xi_i} \quad \tfrac{1}{2}\|\beta\|^2 + C\sum_{i=1}^N \xi_i$$

$$\text{s.t.} \quad \underbrace{\xi_i \geq 0 \;,\; \xi_i \geq 1 - y_i(x_i^T\beta+\beta_0)}_{\xi_i \geq L(y_i,\, x_i^T\beta+\beta_0)} \qquad i=1,\dots,N$$

$$\min_{\beta,\beta_0} \quad \sum_{i=1}^N L(y_i, x_i^T\beta+\beta_0) + \frac{1}{2C}\|\beta\|^2$$

ERM for hinge loss with L_2-penalty, $\lambda = \frac{1}{2C}$.

d) Dual Formulation

$$\max_{\alpha_1,\dots,\alpha_N} \quad \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N\sum_{k=1}^N \alpha_i\alpha_k y_i y_k \langle x_i, x_k\rangle \qquad = x_i^T x_k$$

subject to $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^N \alpha_i y_i = 0$

Then solution to (*) is:

$$\hat\beta = \sum_{i=1}^N \hat\alpha_i y_i x_i$$

← very different, but reminiscent of nearest neighbor, because also defined in terms of training data

$\hat\alpha_i = 0$ for x_i outside margin

$0 < \hat\alpha_i < C$ for x_i on margin

$\hat\alpha_i = C$ for x_i inside margin

solve $\hat\beta_0$ from $\hat\alpha_i[y_i(x_i^T\hat\beta+\hat\beta_0)-1]=0$
for any i s.t. $0<\hat\alpha_i<C$

e) Kernel trick

Fig. 2.5: how to learn something like this with linear classifier?

map features x to a larger set of features $h(x)$!
e.g.

$$h\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_1 \cdot x_2 \\ x_1^2 \\ x_2^2 \end{pmatrix}$$

Then $\langle x_i, x_k \rangle$ in dual formulation becomes $\langle h(x_i), h(x_k) \rangle$

kernel trick: don't need to specify h,
only need to know the kernel function

really nice if this were a simple function..., so turn things around and start by defining $K(x_i, x_k)$ → $K(x_i, x_k) = \langle h(x_i), h(x_k) \rangle$ ← measure of similarity, larger if x_i, x_k more similar.

$h(x)$ may even be infinite-dimensional!

Examples: $K(x, x') = (1 + \langle x, x' \rangle)^d$: d-th degree polynomial

↳ e.g. $d = 2$, $x \in \mathbb{R}^2$:

$$K(x,x') = (1 + x_1 x_1' + x_2 x_2')^2$$
$$= \langle h(x), h(x') \rangle$$

for

$$h(x) = \begin{pmatrix} 1 \\ \sqrt{2}\, x_1 \\ \sqrt{2}\, x_2 \\ x_1^2 \\ x_2^2 \\ \sqrt{2}\, x_1 x_2 \end{pmatrix}$$

radial basis: $K(x,x') = e^{-\gamma \|x - x'\|^2}$

neural network: $K(x,x') = \tanh(a\langle x, x' \rangle + b)$ $\qquad \tanh(z) = \dfrac{e^z - e^{-z}}{e^z + e^{-z}}$

If K satisfies certain technical conditions (symmetric, positive definite) then there always exists some mapping h s.t. $K(x,x') = \langle h(x), h(x') \rangle$.

Classifying a new x:

$$\hat{f}(x) = h(x)^T\hat{\beta} + \hat{\beta}_0 = h(x)^T \sum_{i=1}^N \alpha_i y_i h(x_i) + \hat{\beta}_0$$

$$= \sum_{i=1}^N \alpha_i y_i \langle h(x), h(x_i)\rangle + \hat{\beta}_0$$

$$= \sum_{i=1}^N \alpha_i y_i K(x, x_i) + \hat{\beta}_0$$

Again no need to specify h; only need to know kernel.

Fig. 12.3

f.) Derivation of Dual Formulation ← optional!

$$\min_{\beta, \beta_0, \xi_i} \quad \tfrac{1}{2}\|\beta\|^2 + C\cdot\sum_{i=1}^N \xi_i$$

$$\text{s.t.} \quad \xi_i \geq 0, \quad y_i(x_i^T\beta + \beta_0) - (1-\xi_i) \geq 0 \qquad i = 1, \dots, N$$

$$\min_{\beta, \beta_0, \xi_i} \ \sup_{\alpha_i, \mu_i \geq 0} \ \underbrace{\tfrac{1}{2}\|\beta\|^2 + C\sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i(y_i(x_i^T\beta + \beta) - (1-\xi_i)) - \sum_{i=1}^N \mu_i \xi_i}_{A}$$

(If β, β_0, ξ_i violate constraints, then α_i or μ_i becomes ∞ and $A = \infty$, so β, β_0, ξ_i only achieve minimum while satisfying constraints. And then α_i, μ_i become 0, so the constraints ~~[illegible]~~ drop away and we are minimizing the previous objective.)

$$\cancel{\nabla_\beta A = 0 \Rightarrow \beta = \sum_{i=1}^N \alpha_i y_i x_i} \qquad \cancel{\nabla_{\beta_0} A = 0 \Rightarrow \sum_{i=1}^N \alpha_i y_i = 0}$$

$$\cancel{\nabla_{\xi_i} A = 0 \Rightarrow \mu_i = C - \alpha_i}$$

~~plugging~~

$$\min_{\beta, \beta_0, \xi_i} \sup_{\alpha_i, \mu_i \geq 0} A = \sup_{\alpha_i, \mu_i \geq 0} \underbrace{\min_{\beta, \beta_0, \xi_i} A}$$

by convex optimization theory (Slater's condition)

can solve this

$$\nabla_\beta A = 0 \Rightarrow \beta = \sum_{i=1}^{N} \alpha_i y_i x_i \qquad \nabla_{\beta_0} A = 0 \Rightarrow \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\nabla_{\xi_i} A = 0 \Rightarrow \mu_i = C - \alpha_i$$

Plugging these in gives dual formulation.

Unsupervised Learning

1) Intro

Supervised: $T = \binom{Y_1}{X_1}, \ldots, \binom{Y_N}{X_N}$ ← can evaluate every method by EPE

Unsupervised: $T = \{X_1, \ldots, X_N$

What can we do?

Clustering: split data into groups of points (clusters) that are similar

1. What do you mean by "similar"?
2. What kind of clusters are you looking for?

K-means

① Squared Euclidean distance between $x_i, x_{i'}$:
$$d(x_i, x_{i'}) = \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

② Find K clusters s.t. distance to mean within each cluster is small.

$C(i) \in \{1, \ldots, K\}$ is cluster assigned to x_i.

$$\text{minimize} \quad \sum_{k=1}^{K} \sum_{i: C(i)=k} d(x_i, \mu_k) \qquad (*)$$

where μ_k = mean of points in cluster k

NB. Mistake in book! (multiplies by N_k in each cluster)

Algorithm:
1. Initialize cluster assignment C
2. Set μ_k to current mean in cluster k
3. Given means $\mu_1, \ldots, \mu_K$, assign each x_i to nearest mean to get new C
4. repeat from 2 until no changes in C.

- Converges to local minimum
- ~~Often~~ May choose K s.t. increasing K does not reduce (*) very much.

Gaussian Mixtures and EM

- like k-means, but with "soft" cluster assignments each cluster is modelled by a Gaussian
- for simplicity: explain for two clusters

cluster 1: $X \sim \mathcal{N}(\mu_1, \sigma_1^2)$ w.p. $1-\pi$
cluster 2: $X \sim \mathcal{N}(\mu_2, \sigma_1^2)$ w.p π

probability density: $(1-\pi)\,\varphi_{\mu_1,\sigma_1^2}(x) + \pi\,\varphi_{\mu_2,\sigma_2}(x)$

To find parameters:
ML is difficult numerically
+ gives bad solution with $\hat{\sigma}_1 = 0$
$\hat{\mu}_1 = x_i$ for some i.

Solution: algorithm very similar to K-means

Hidden variables $\Delta_i = \begin{cases} 0 & \text{if } x_i \text{ in cluster 1} \\ 1 & \text{if } x_i \text{ in cluster 2} \end{cases}$

1. Initializes parameters $\underbrace{\hat{\pi}, \hat{\mu}_1, \hat{\sigma}_1, \hat{\mu}_2, \hat{\sigma}_2}_{\hat{\theta}}$ $(\hat{\sigma}_1 > 0.5, \hat{\sigma}_2 > 0.5)$

2. Soft cluster assignment:

$$\hat{\gamma}_i = \mathbb{E}[\Delta_i \mid \hat{\theta}, T] = \frac{\hat{\pi}\,\varphi_{\hat{\mu}_2,\hat{\sigma}_2}(x_i)}{\hat{\pi}\,\varphi_{\hat{\mu}_2,\hat{\sigma}_2}(x_i) + (1-\hat{\pi})\,\varphi_{\hat{\mu}_1,\hat{\sigma}_1}(x_i)}$$

3. Update parameter estimates for clusters:

$$\hat{\mu}_1 = \frac{\sum_{i=1}^{N}(1-\hat{\gamma}_i)x_i}{\sum_i (1-\hat{\gamma}_i)} \qquad \hat{\mu}_2 = \frac{\sum_{i=1}^{N} \hat{\gamma}_i x_i}{\sum_i \hat{\gamma}_i}$$

$$\hat{\sigma}_1^2 = \frac{\sum_{i=1}^{N}(1-\hat{\gamma}_i)(x_i-\hat{\mu}_1)^2}{\sum_i (1-\hat{\gamma}_i)} \qquad \hat{\sigma}_2^2 = \frac{\sum_{i=1}^{N} \hat{\gamma}_i (x_i-\hat{\mu}_2)^2}{\sum_i \hat{\gamma}_i}$$

$$\hat{\pi} = \frac{\sum_{i=1}^{N} \hat{\gamma}_i}{\sum_{i=1}^{N} \hat{\gamma}_i + \sum_{i=1}^{N}(1-\hat{\gamma}_i)} = \frac{\sum_{i=1}^{N} \hat{\gamma}_i}{N}$$

4. repeat from 2 until convergence

- converges to local minimum