

Statistical Learning VII

1

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

2. Decision Trees

- a) General set-up
- b) Growing
- c) Pruning

Vapnik & Chervonenkis: foundational work on statistical learning starting in sixties and seventies, leading to SVMs in 90s.

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: $P(Y|X)$

now: decision boundary directly

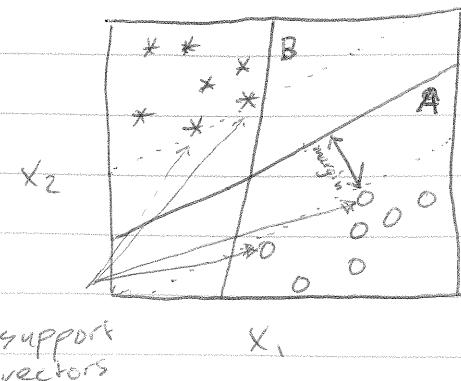
Assume: 2 classes, linearly separable

$$\begin{pmatrix} y_1 \\ x_1 \end{pmatrix}, \dots, \begin{pmatrix} y_N \\ x_N \end{pmatrix} \quad y \in \{-1, +1\}$$

Linear

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

(2)



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.

support vectors: points on 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 $\frac{|f(x_i)|}{\|\beta\|}$

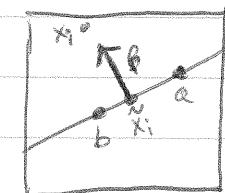
margin: $\frac{y_i \cdot f(x_i)}{\|\beta\|}$



is distance if x_i classified correctly

is negative distance if classified incorrectly

Optional: for any a, b on decision boundary:



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

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

$$x_i = \hat{x}_i + \alpha \cdot \frac{\beta}{\|\beta\|} \quad (\text{|\alpha| is distance of } x_i \text{ from decision boundary})$$

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

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

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

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

$$= f(x_i)$$

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

(3)

to maximize the margin M :

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

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

margin of (x_i)

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

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

$$\text{s.t. } \|\beta\| = \frac{1}{M}$$

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

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

$$\text{s.t. } \frac{y_i(x_i^T \beta + \beta_0)}{\|\beta\|} \geq 1 \quad \forall i$$

solution achieved by same parameters β, β_0

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

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

④

b) SVMs

What if classes are not linearly separable?

Greek letter "xi"

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

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

$$\text{subject to: } \xi_i \geq 0, \frac{y_i(x_i^\top \beta + \beta_0)}{\|\beta\|} \geq M(1 - \xi_i) \quad i=1, \dots, N$$

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

support vectors: all points inside the margin

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

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

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

equivalent

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

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

(*)

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

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"

(5)

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

s.t. $\xi_i \geq 0, \underbrace{\xi_i \geq 1 - y_i(\mathbf{x}_i^\top \beta + \beta_0)}_{\xi_i \geq L(y_i, \mathbf{x}_i^\top \beta + \beta_0)} \quad i=1, \dots, N$

$$\min_{\beta, \beta_0} \sum_{i=1}^N L(y_i, \mathbf{x}_i^\top \beta + \beta_0) + \frac{1}{2C} \|\beta\|^2$$

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

d) Dual Formulation

$$\max_{\alpha_1, \dots, \alpha_N} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{k=1}^N \alpha_i \alpha_k y_i y_k \langle \mathbf{x}_i, \mathbf{x}_k \rangle \quad \approx \mathbf{x}_i^\top \mathbf{x}_k$$

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

Then solution to (*) is:

$$\hat{\beta} = \sum_{i=1}^N \hat{\alpha}_i y_i \mathbf{x}_i \quad \begin{array}{l} \text{very different, but} \\ \text{reminiscent of nearest neighbor, because} \\ \text{also defined in terms of training data} \end{array}$$

$\hat{\alpha}_i = 0$ for \mathbf{x}_i outside margin

$0 < \hat{\alpha}_i < C$ for \mathbf{x}_i on margin

$\hat{\alpha}_i = C$ for \mathbf{x}_i inside margin

solve $\hat{\beta}_0$ from $\hat{\alpha}_i [y_i (\mathbf{x}_i^\top \hat{\beta} + \hat{\beta}_0) - 1] = 0$
for any i s.t. $0 < \hat{\alpha}_i < C$

⑥

c) 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 $K(x_i, x_k)$ measure of similarity, larger if x_i, x_k more similar.

Examples: $K(x, x') = (1 + \langle x, x' \rangle)^d$: d-th degree polynomial
 $\log. d = 2, x \in \mathbb{R}^2$:

$$\begin{aligned} K(x, x') &= (1 + x_1 x'_1 + x_2 x'_2)^2 \\ &= \langle h(x), h(x') \rangle \end{aligned}$$

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: $K(x, x') = \tanh(\alpha \langle x, x' \rangle + b)$

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

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

(7)

Classifying a new x :

$$\begin{aligned}\hat{f}(x) &= h(x)^T \beta + \hat{\beta}_0 = h(x)^T \sum_{i=1}^N \alpha_i y_i h(x_i) + \hat{\beta}_0 \\ &= \sum_{i=1}^N \alpha_i y_i \langle h(x), h(x_i) \rangle + \hat{\beta}_0 \\ &= \sum_{i=1}^N \alpha_i y_i K(x, x_i) + \hat{\beta}_0\end{aligned}$$

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

Fig. 12.3

f.) Derivation of Dual Formulation ← optional! □

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

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

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

(If β, β_0, ξ_i violate constraints, then α_i or ρ_i becomes 0 and A=0, so β, β_0, ξ_i only achieve minimum while satisfying constraints. And then α_i, ρ_i become 0, so the constraints drop away and we are minimizing the previous objective.)

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

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

plugging

(8)

$$\min_{\alpha, \beta_0, \gamma_i} \sup_{x_i, p_i \geq 0} A = \sup_{x_i, p_i \geq 0} \underbrace{\min_{\alpha, \beta_0, \gamma_i} A}_{\text{by convex optimization theory (Slater's condition)}}$$

↑
can solve this

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

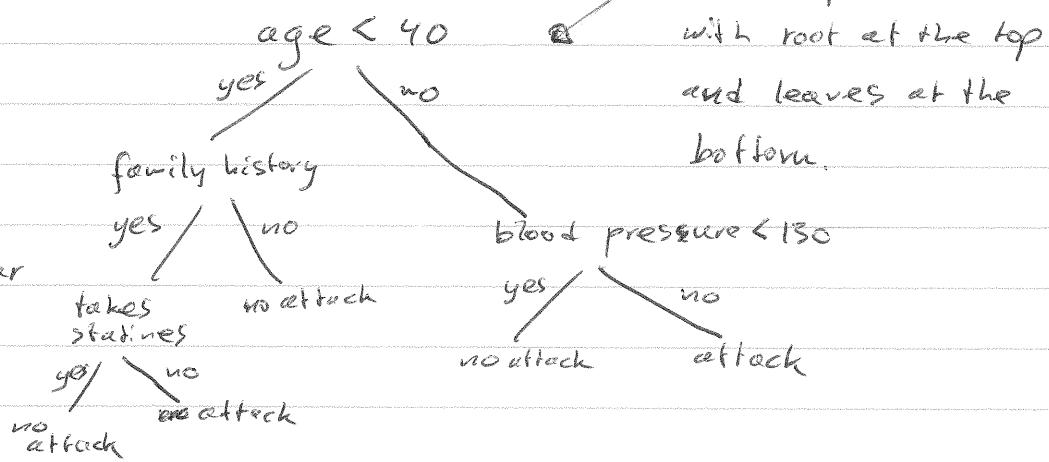
$$\nabla_{\alpha_i} A = 0 \Rightarrow p_i = C - \alpha_i$$

Plugging these in gives dual formulation.

2. Decision Trees

a) General Set-up

decision tree
for classification of
heart attack next year



Advantages:
+ easy to interpret
+ nice algorithms

Disadvantages:
 * not competitive with logistic regression/kernel method
 (will be fixed next lecture)
 * instability / learning algorithms have high variance
 ⇒ small changes in data lead to very different tree
 ⇒ undermines interpretation

⑨

Fig. 9-2

Estimates class or regression value independently per region with ERM^{Rm}

$$\hat{c}_m = \underset{c}{\operatorname{argmin}} \sum_{x_i \in R_m} L(y_i, c)$$

regression: $L(y_i, c) = (y_i - c)^2$ "squared error" $\Rightarrow c_m$ is avg y_i in R_m

classification: 0/1-loss: $\Rightarrow c_m$ is most common class in R_m

or multinomial model with log loss:

(for two classes for simplicity)

* c_m is now probability $P(Y=1 | x \in R_m)$

$$\hat{c}_m = \underset{c}{\operatorname{argmin}} \sum_{x_i \in R_m} -\log P(Y=y_i | x_i \in R_m)$$

$$= \underset{c}{\operatorname{argmin}} -n_1 \log c - n_0 \log(1-c) \quad (n_k = \#k \text{ in } R_m)$$

$$= \underset{c}{\operatorname{argmin}} -\hat{p}_{m,1} \log c - \hat{p}_{m,0} \log(1-c)$$

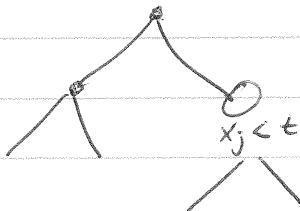
$$\text{where } \hat{p}_{m,k} = \frac{n_k}{n_0 + n_1}$$

$\hat{c}_m = \hat{p}_{m,1}$ is ML estimate for Bernoulli model

$-\hat{p}_{m,1} \log \hat{p}_{m,1} - \hat{p}_{m,0} \log \hat{p}_{m,0}$ is "entropy"

(10)

b) Growing



How to choose feature x_j + threshold t ?

Greedy approach: split region R_m into R_{m1} and R_{m2} that minimize

$$\text{Loss}(R_{m1}) + \text{Loss}(R_{m2})$$

$$\min_c \sum_{x_i \in R_{m1}} L(y_i, c)$$

Regression: always use squared error loss

Classification:

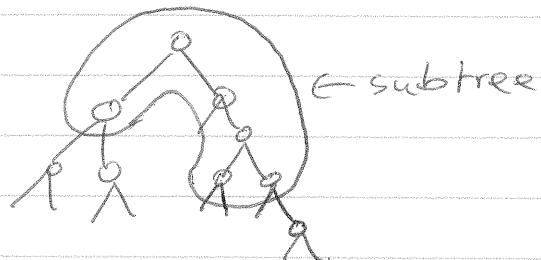
- use entropy or "Gini index" while growing
- 0/1-loss to estimate class in each region

Fig. 9.3: entropy and Gini more sensitive to changes around 0 and 1.

(11)

Pruning:

- * Bigger tree \rightarrow more regions \rightarrow better fit training data
- * Too large \rightarrow overfitting because too little data per region.
too small \rightarrow underfitting.
- * Difficult to know when to stop growing tree
- * Solution: grow very big tree, then prune to smaller tree.



choose subtree $T_\alpha \subseteq T$ that minimizes:

$$\sum_{i=1}^N L(y_i, T_\alpha(x_i)) + \alpha \cdot |T_\alpha|$$

$$T_\alpha(x_i) = \hat{c}_n$$

for m s.t. $x_i \in R_m$.

$|T_\alpha|$
nr of terminal nodes
in tree