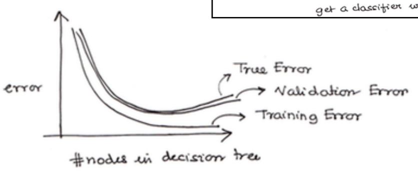
With Pruning:



As tree grows more complicated, training error decreases. At some point, true error stops improving and may even get worse. This will be reflected in the validation error, provided we have enough validation data.

Note:
1. Lower margin ⇒ more mistakes.
2. May need >1 pass over training data to get a classifier with no mistakes.

Properties of Entropy:

1. $H(X)$ does not depend on the exact values taken by X, only on the probabilities of distinct values.

2. $H(X) \geq 0$ (always). Why?

$$H(X) = -\sum_{i=1}^{K} p_i \log p_i, \text{ where } \log p_i \leq 0, \text{ as } p_i \leq 1.$$

3. If X takes k values, $H(X) \leq \log k$. This maximum is achieved only when each value occurs w.p. $1/k$.

Properties of Conditional Entropy:

1. Suppose $X = Z$. What is $H(X|Z)$?

$$H(X|Z) = \sum_z \Pr(Z=z)\, H(X|Z=z).$$

Now, given that $Z = z$, we know that $X = z$ w.p. 1.

So, $H(X|Z=z) = 0$.

Thus, $H(X|Z) = \sum_z \Pr(Z=z) \cdot 0 = 0$.

(This may hold even if $H(X)$ is very large!)

2. Suppose X, Z are independent. What is $H(X|Z)$?

Since X, Z are independent, $\Pr(X=x|Z=z)$ for any z and x, is equal to $\Pr(X=x)$.

i.e. $X|Z=z$ has exactly the same distribution as X.

So. $H(X|Z=z) = H(X)$

$$H(X|Z) = \sum_z \Pr(Z=z) H(X|Z=z) = H(X) \sum_z \Pr(Z=z) = H(X)$$

How to Avoid Overfitting in Decision Trees? By Pruning.

- By pruning the tree using a validation dataset.

1. Split the training data into training set S and validation set V.
2. Build ID3 tree T using training sets
3. Prune using V:

   Repeat:

   For each node v in T:

   Replace subtree rooted at v by single node that predicts majority label in v to get tree T'

   If error(T') on V $\leq$ error(T) on V, then $T = T'$

   Continue till there is no such node v.

Measure of Separability: Margin.

For a vector w, and training set S, margin of w wrt S is:

$$\gamma = \min_{(x,y)\in S} \frac{|\langle w, x\rangle|}{\|w\|}$$

Example: $S = ((1,-1),1),\ ((-1,-1),1),\ ((0.01,0),1),\ ((-1,0),-1)$

$w = (1, 0)$

What is the margin of w wrt S? $\|w\| = 1$.

* $\frac{|\langle w, x_1\rangle|}{\|w\|} = 1$

$\frac{|\langle w, x_2\rangle|}{\|w\|} = 1$

$\frac{|\langle w, x_3\rangle|}{\|w\|} = 0.01$

$\frac{|\langle w, x_4\rangle|}{\|w\|} = 1$.

So, margin = 0.01

Voted Perceptron:

Initially, $m = 1$, $w_1 = 0$, $c_m = 1$

For $t = 1, 2, 3, \ldots$

If $y_t \langle w_m, x_t\rangle \leq 0$ then:

$w_{m+1} = w_m + y_t x_t$

$m = m + 1$

$c_m = 1$

Else:

$c_m = c_m + 1$.

Output $(w_1, c_1), (w_2, c_2), \ldots, (w_m, c_m)$

c_m is a count of how long w_m surviv

How to classify test example x?

Output: $\text{sign}\left(\sum_{i=1}^{m} c_i \,\text{sign}(\langle w_i, x\rangle)\right)$

Perceptron Example:

Training data: $((4,0),1),\ ((1,1),-1),\ ((0,1),-1),\ ((-2,-2),1)$

Initially, $w_1 = 0$, $m = 1$, $c_m = 1$

Round 1: $y_1 \langle w_1, x_1\rangle = 0$.

So: $m = 2$, $c_m = 1$, $w_2 = w_1 + y_1 x_1 = (4, 0)$

Round 2: $y_2 \langle w_2, x_2\rangle \leq 0$.

So: $m = 3$, $c_m = 1$, $w_3 = w_2 + y_2 x_2 = (3, -1)$

Linear Classification by a Hyperplane not thru Origin

We can transform this problem to linear classification by a hyperplane through the origin.

Original Problem:

Training data: (x_i, y_i), $i = 1, \ldots, n$, $x_i \in \mathbb{R}^d$, $y_i \in \{-1, 1\}$

Separating hyperplane: $a^T x + b \geq 0$

$(a \in \mathbb{R}^d,\ b = \text{a scalar})$

Transform it to:

Training data: (z_i, y_i), $i = 1, \ldots, n$, $z_i = \begin{bmatrix} 1 \\ x_i \end{bmatrix} \in \mathbb{R}^{d+1}$, $y_i \in \{-1, 1\}$

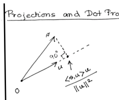
For all $i = 1, \ldots, n$,

$$y_i \left\langle z_i, \begin{bmatrix} b \\ a \end{bmatrix}\right\rangle = y_i \left\langle \begin{bmatrix} 1 \\ x_i \end{bmatrix}, \begin{bmatrix} b \\ a \end{bmatrix}\right\rangle = y_i (b + a^T x_i) \geq 0$$

(as (x_i, y_i) are linearly separable)

Thus, if (x_i, y_i)'s have a separating hyperplane, then then (z_i, y_i)'s have a separating hyperplane thru origin.

Projections and Dot Products:



Look at plane containing u, v.

Draw a perpendicular from v to u.

Projection of v along u is the vector $\frac{\langle v, u\rangle u}{\|u\|^2} = a$

This is also called "component" of v along u.

Component of v perpendicular to u is $v - \frac{\langle v, u\rangle u}{\|u\|^2} = b$.

How to Kernelize Perceptron:

1. Initially, $w_1 = 0$.
2. For $t = 1, 2, 3, \ldots$

   If $y_t \langle w_t, \Phi(x_t)\rangle \leq 0$ then

   $w_{t+1} = w_t + y_t \Phi(x_t)$

   Else

   $w_{t+1} = w_t$

Examples of Kernels:

[1] $K(x, z) = (\langle x, z\rangle)^2$.

What is the feature space?

* Suppose x, z are 2-d vectors, $x = [x_1, x_2]$, $z = [z_1, z_2]$

$K(x,z) = (x_1 z_1 + x_2 z_2)^2$

$= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2$

If $\Phi(x) = [x_1^2, x_2^2, \sqrt{2} x_1 x_2]$, then,

$K(x, z) = \langle \Phi(x), \Phi(z)\rangle$

* If x, z are d-dimensional vectors, $x = [x_1, \ldots, x_d]$, $z = [z_1, \ldots, z_d]$,

then, $K(x, z) = (x_1 z_1 + \cdots + x_d z_d)^2$

$= \sum_{i=1}^{d} x_i^2 z_i^2 + 2 \sum_{i \neq j} x_i z_i x_j z_j$

If $\Phi(x) = [\underbrace{x_1^2, \ldots, x_d^2}_{d \text{ terms}}, \underbrace{\sqrt{2} x_1 x_2, \sqrt{2} x_1 x_3, \ldots}_{\binom{d}{2} \text{ terms}}]$

then $K(x, z) = \langle \Phi(x), \Phi(z)\rangle$

[2] $K(x, z) = (\langle x, z\rangle + c)^2$ $c \geq 0$.

What is the corresponding feature map?

$K(x,z) = \langle x,z\rangle^2 + 2c\langle x,z\rangle + c^2$

$= \left(\sum_{i=1}^{d} x_i z_i\right)^2 + 2c \sum_{i=1}^{d} x_i z_i + c^2$

$= \sum_{i=1}^{d} x_i^2 z_i^2 + 2 \sum_{i \neq j} x_i x_j z_i z_j + 2c \sum_{i=1}^{d} x_i z_i + c^2$

If $\Phi(x) = [\underbrace{x_1^2, \ldots, x_d^2}_{d \text{ terms}}, \underbrace{\sqrt{2} x_1 x_2, \sqrt{2} x_1 x_3, \ldots}_{\binom{d}{2} \text{ terms}}]$ is feature map from example 1, and if

$\Psi(x) = [\Phi(x), \sqrt{2c}\, x, c]$, then

$K(x, z) = \langle \Psi(x), \Psi(z)\rangle$

Note: Length of $\Phi(x)$ is again $\Theta(d^2)$, but $K(x,z)$ takes $O(d)$ time to compute.

[3] $K(x, z) = (\langle x, z\rangle + c)^k$, $c \geq 0$, k = integer (Polynomial Kernel)

Corresponding feature map has $d^{O(k)}$ terms, but $K(x, z)$ can be computed in $O(d+k)$ time

($\Theta(d)$ to compute $\langle x, z\rangle$, $O(1)$ to add c, $O(k)$ to raise to k-th power)

[4] $K(x, z) = e^{-\|x-z\|^2/c^2}$ (called the Gaussian Kernel)

Corresponds to an infinite dimensional feature map.

* Time taken to write down $\Phi(x) = \Theta(d^2)$
* Space taken to store $\Phi(x) = \Theta(d^2)$
* Time taken to compute $K(x, z) = (\langle x, z\rangle)^2$ is $\Theta(d)$ [$\Theta(d)$ time to compute $\langle x, z\rangle$, $O(1)$ time to square]

String Kernels.

Let s and t be strings over an alphabet Σ, p = an integer > 0.

$K(s,t)$ = # of common substrings of length p in s and t.

eg. for $p = 1$, $K(s,t)$ = # common letters in s and t.

so, if s = "asdf", t = "gpsd", then $K(s,t) = 2$ for $p = 1$.

[6] Graph Kernels.

Let G_1, G_2 be two graphs, $p \geq 2$ an integer.

$\Phi(G)$ has a coordinate for every kind of subgraphs of size p. eg. for $p = 3$, possible subgraphs are:

$K(G_1, G_2) = \langle \Phi(G_1), \Phi(G_2)\rangle$

Kernel Properties:

Not all functions $K(x, z)$ are kernels.

Conditions for a function to be a kernel:

1. Symmetry: For all x, z, $K(x,z) = K(z,x)$
2. Positive Semi Definiteness: For a set of points $x^1, \ldots, x^m$, define kernel matrix as:

   $K_{m \times m}$, $K_{ij} = K(x^i, x^j)$

   For all $x^1, \ldots, x^m$, the kernel matrix is PSD.

These are necessary and sufficient conditions.

How to show a function $K(x, z)$ is a kernel?

1. Either find a feature map ϕ s.t. $K(x,z) = \langle \phi(x), \phi(z)\rangle$
2. Or show conditions (1) and (2) hold (usually harder)

The Perceptron Algorithm:

1. Initially, $w_1 = 0$.
2. For $t = 1, 2, 3, \ldots$

   If $y_t \langle w_t, x_t\rangle \leq 0$ then

   $w_{t+1} = w_t + y_t x_t$

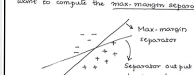
   Else

   $w_{t+1} = w_t$

However, perceptron will never converge to a single w if the data is not linearly separable, as we make more passes over training data.

Margins:

- Data linearly separable with a margin.
  * Perceptron stops once a separator is found, but we may want to compute the max-margin separator.



Fact: $\|x-y\|^2 = \|x\|^2 + \|y\|^2 - 2\langle x, y\rangle$

Proof: $\|x-y\|^2 = \langle x-y, x-y\rangle$ (From (1))

$= \langle x, x\rangle + \langle y, y\rangle - 2\langle x, y\rangle$ (From properties of dot products)

$= \|x\|^2 + \|y\|^2 - 2\langle x, y\rangle$

Training error of a classifier:

$\hat{err}(h) = \frac{1}{n}\sum_{i=1}^{n} 1(h(x_i) \neq y_i)$

[$1(\cdot)$ means an indicator variable which is 1 when the condition is true,]

True error of a classifier:

$err(h) = \Pr_{(x,y)\sim D}(h(x) \neq y)$

For a fixed classifier, with more data, training error should approach true error.

Example:

- Suppose a node v has 5 points with label 0, 2 with label 1.

  Then, the corresponding X would be a r.v.

  with: $\Pr(X=0) = \frac{5}{7}$, $\Pr(X=1) = \frac{2}{7}$.

So, $H(X) = -\frac{5}{7}\log\frac{5}{7} - \frac{2}{7}\log\frac{2}{7}$

Splitting Rule: We pick the feature and threshold which maximizes information gain.

$IG(Z) = H(X) - H(X|Z)$

Information Gain(Z) = $H(X) - H(X|Z)$

(essentially, how much entropy of X is reduced because we know Z).

Conditional Entropy:

Let X, Z be two r.v.s. The conditional entropy of X given Z is defined as:

$$H(X|Z) = \sum_z \Pr(Z=z) H(X|Z=z)$$

(Intuitively, average entropy of X given that we know Z).

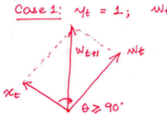
Entropy: Let X be a random variable that takes values $v_1, \ldots, v_K$ with probabilities $p_1, \ldots, p_K$. Then, entropy of X, denoted by $H(X)$ is defined as:

$$H(X) = -p_1 \log p_1 - p_2 \log p_2 - \cdots - p_K \log p_K$$

(Note: Use the convention that $0 \cdot \log 0 = 0$)

Geometric interpretation of: "$w_{t+1} = w_t + y_t x_t$"

Case 1: $y_t = 1$; $w_{t+1} = w_t + x_t$

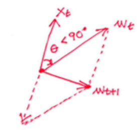


w_{t+1} moves "closer to" x_t

or

x_t moves towards the + side of the decision boundary (hyperplane)

Case 2: $y_t = -1$; $w_{t+1} = w_t - x_t$.



w_{t+1} moves "away from" x_t

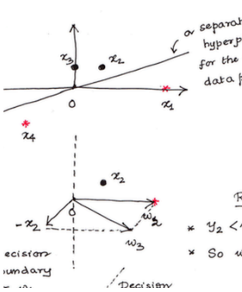
or

x_t moves towards the − side of the decision boundary hyperplane.

In both cases, we are moving towards the "correct solution".

Example:

Training Data: $((4,0),1),\ ((1,1),-1),\ ((0,1),-1),\ ((-2,-2),1)$



Round 1:

* $w_1 = 0$

* $y_t \langle w_t, x_t\rangle$ for $t = 1$ $= 0$ as well.

* $w_2 = w_1 + y_1 x_1 = (4, 0)$

Round 2:

* $y_2 \langle w_2, x_2\rangle < 0$

* So $w_3 = w_2 + y_2 x_2 = (4,0) - (1,1) = (3,-1)$

Boosting Algorithm:

Input: Training set $S = \{(x_1, y_1), .., (x_n, y_n)\}$, $y_i = \pm 1$

$D_1(i) = 1/n$ for all $i = 1, .., n$

For $t = 1, 2, 3, \ldots$

h_t = weak-learner wrt D_t. (so, $err_{D_t}(h_t) < 0.5$)

$\varepsilon_t = err_{D_t}(h_t)$

$\alpha_t = \frac{1}{2} \ln \frac{1-\varepsilon_t}{\varepsilon_t}$ (so, α_t is high when ε_t is low, and almost 0 when ε_t is close to 0.5)

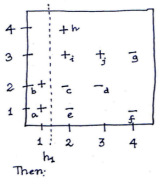
$D_{t+1}(i) = \frac{D_t(i)\, e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$ (D_{t+1} goes ↑ if i is misclassified by h_t; so higher D_t means harder example.)

where Z_t is a normalization constant to ensure that $\sum_i D_{t+1}(i) = 1$.

Final classifier: $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$ (weighted majority of $h_t(x)$'s)

Boosting Algorithm Example:

Training data: $((1,1),+)$ $((2,1),-)$ $((4,1),-)$ $((1,2),+)$ $((2,2),-)$ $((3,2),-)$ $((2,3),+)$ $((3,3),+)$ $((4,3),-)$ $((2,4),+)$



Initially: $D_1(i) = 0.1$ (for all i)

Weak Learners: • Set of vertical and horizontal thresholds.

① Suppose we pick $h_1(x) = +$ if $x_1 \le 1.5$ $= -$ otherwise

Name the points: a, b, .., j (for ease of understanding)

Then:

$err_{D_1}(h_1) = \varepsilon_1 = 0.3$ $\quad \alpha_1 = 0.42$

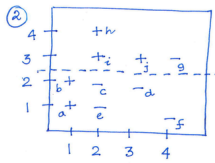
Weights of a, b, c, d, e, f, g: $D_2 = 0.07$

Weights of h, i, j: $D_2 = 0.17$

$Z_2 = 7 \cdot e^{-0.42} \cdot 0.1 + 3 \cdot 0.1 \cdot e^{0.42} = 0.92$

Note: Calculations rounded to 2 decimal places.

② In Round 2, suppose we pick



$h_2(x) = +$ if $x_2 > 2.5$ $= -$ otherwise.

$err_{D_2}(h_2) = \varepsilon_2 = 0.21$

$\alpha_2 = 0.66$

Weights of a, b: $D_3 := 0.07 \times e^{0.66}/Z_3 = 0.17$

Weights of c, d, e, f: $D_3 := 0.07 \times e^{-0.66}/Z_3 = 0.04$

Weights of h, i, j: $D_3 := 0.17 \times e^{-0.66}/Z_3 = 0.11$

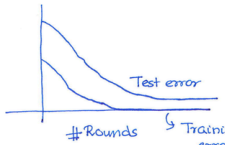
Weight of g: $D_3 := 0.07 \times e^{0.66}/Z_3 = 0.17$

$Z_3 = 0.81$

Boosting and Overfitting:

Overfitting can happen with boosting, but often does not.

Typical boosting run:



Reason is that the margin of classification often increases with boosting.

When do we have high variance?

High variance when there is a small amount of training data and a very complicated concept class.

When you have high variance, it is called overfitting

Variance decreases with larger training data, and increases with more complicated classifiers.

High bias ⇒ High training and test errors.

High variance ⇒ low training error, high test errors.

Bias-variance tradeoff: If we make the concept class more complicated, then, for the same training set size, bias decreases but variance increases. Thus there is a bias-variance tradeoff.

Bias is the true error of the best classifier in the concept class. eg. best linear separator, best decision tree, etc.

When do we have high bias?

High bias when the concept class cannot model the true data distribution well; this doesn't depend on the training data size.

When you have high bias, its called underfitting

Problem 1

	1 Cancer	0 No Cancer
1 Gene A	(1/2)	(1/10)
0 No Gene A	(1/5)	(1/5)
1 Gene B	(2/5)	(3/20)
0 No Gene B	(3/10)	(3/20)

1.1) conditional distribution for $X|Y = y$ for $y = 0, 1$

$P(X|Y=0,1) =$ $\quad Pr(Y=0) = \frac{1}{5} + \frac{1}{5} = \frac{2}{5}$, $Pr(Y=1) = \frac{1}{2} + \frac{1}{10} = \frac{3}{5}$

$X|Y=0$: $\quad Pr(X=0|Y=0) = (1/5) \div (2/5) = (1/2) = \boxed{1/2}$

$Pr(X=1|Y=0) = (1/5) \div (2/5) = \boxed{1/2}$

$X|Y=1$: $\quad Pr(X=0|Y=1) = (1/10) \div (3/5) = \boxed{1/6}$

$Pr(X=1|Y=1) = (2/3) \div (3/5) = \boxed{5/6}$

1.2) Calculate Conditional Entropies $H(X|Y)$ and $H(X|Z)$

$H(X|Y) = P(Y=0)\, H(X|Y=0) + P(Y=1)\, H(X|Y=1)$

$H(X|Y=0) = -\frac{1}{2}\log\left(\frac{1}{2}\right) - \frac{1}{2}\log\left(\frac{1}{2}\right) = .30102999957$

$H(X|Y=1) = -\frac{1}{6}\log\left(\frac{1}{6}\right) - \frac{5}{6}\log\left(\frac{5}{6}\right) = .19567624468$

$\therefore H(X|Y) = \frac{2}{5}(.30102999957) + \frac{3}{5}(.19567624468) = \boxed{.23781774464}$

Definitions:

1. Weak Learner: A simple rule of thumb that doesn't necessarily work very well.
2. Strong Learner: A good classifier (with high accuracy)

$$err_W(h) = \sum_{i=1}^{n} w_i\, 1(h(x_i) \neq y_i)$$

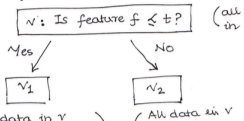
When to stop boosting? Use a validation dataset to find a stopping time. Stop when validation error does not improve.

ID3 Algorithm:

1. Initially, Whole training data is at root.
2. While there is an impure node:
   (a) Pick any impure node v
   (b) Pick a feature f and threshold t along which to "split" the data at v. Done according to "Splitting Rule" (to be described later)
   (c) Modify tree as:

Impure Node: Node with data points with multiple labels

Each node in the algorithm corresponds to a subset of the training data.



If any of v_1 or v_2 is pure (i.e. has data of only one label), then ~~predict~~ t make it a leaf that predicts this label.

‡ Averaged Perceptron uses the same algorithm as voted perceptron, but the classification rule is different.

Averaged Perceptron Classification Rule for test example x:

Output $\text{sign}\left(\langle \sum_{i=1}^{m} c_i w_i, x \rangle\right)$

When x and y are orthogonal,

$$\langle x, y \rangle = 0$$

$$\|x - y\|^2 = \|x\|^2 + \|y\|^2$$

Cauchy Schwartz Inequality:

For any vectors x and y,

$$\langle x, y \rangle \leq \|x\| \cdot \|y\|$$

A matrix $A_{d \times d}$ is PSD if for all $d \times 1$ vectors x,

$$x^T A x \geq 0$$

$$x^T A x = x^T z = \sum_{i=1}^{d} x_i z_i = \sum_{i=1}^{d} \sum_{j=1}^{d} A_{ij} x_i x_j$$

Kernels to Distances:

Given a kernel function $K(x,z)$, define a distance function:

$$D_K(x,z) = \sqrt{K(x,x) + K(z,z) - 2K(x,z)}$$

If $K(x,z) = \langle \Phi(x), \Phi(z) \rangle$ then:

$$D_K^2(x,z) = \langle \Phi(x), \Phi(x) \rangle + \langle \Phi(z), \Phi(z) \rangle - 2\langle \Phi(x), \Phi(z) \rangle$$

$$= \langle \Phi(x) - \Phi(z), \Phi(x) - \Phi(z) \rangle$$

$$= \|\Phi(x) - \Phi(z)\|^2$$