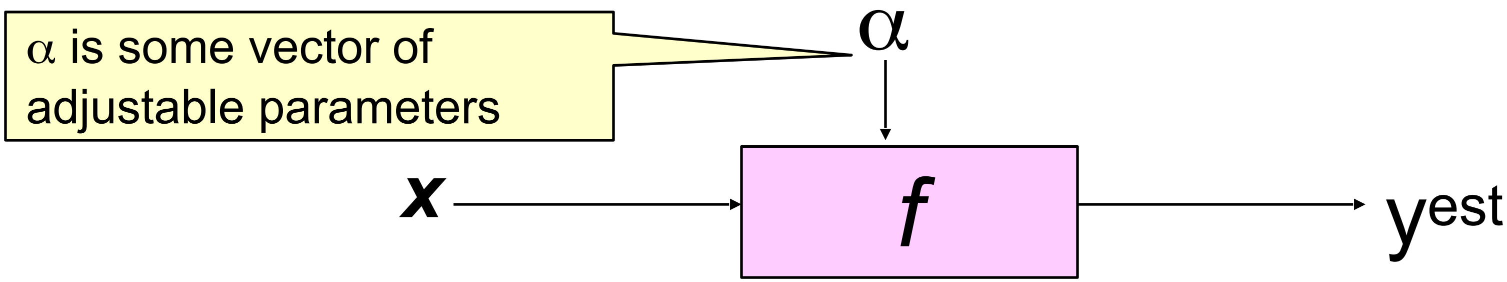
# VC-dimension for characterizing classifiers



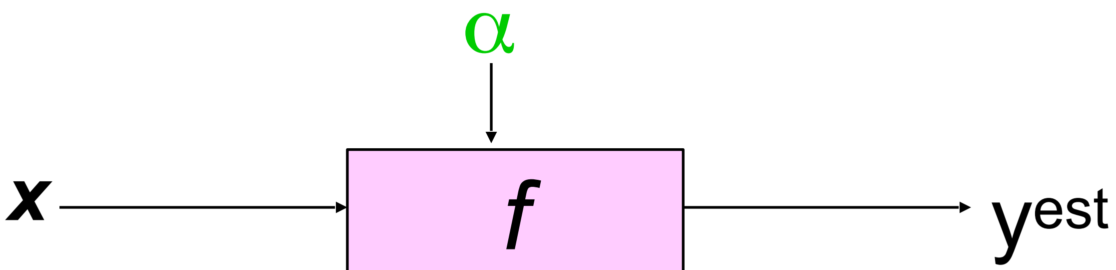
**Andrew W. Moore**
**Associate Professor**
**School of Computer Science**
**Carnegie Mellon University**
www.cs.cmu.edu/~awm
awm@cs.cmu.edu
412-268-7599

# A learning machine

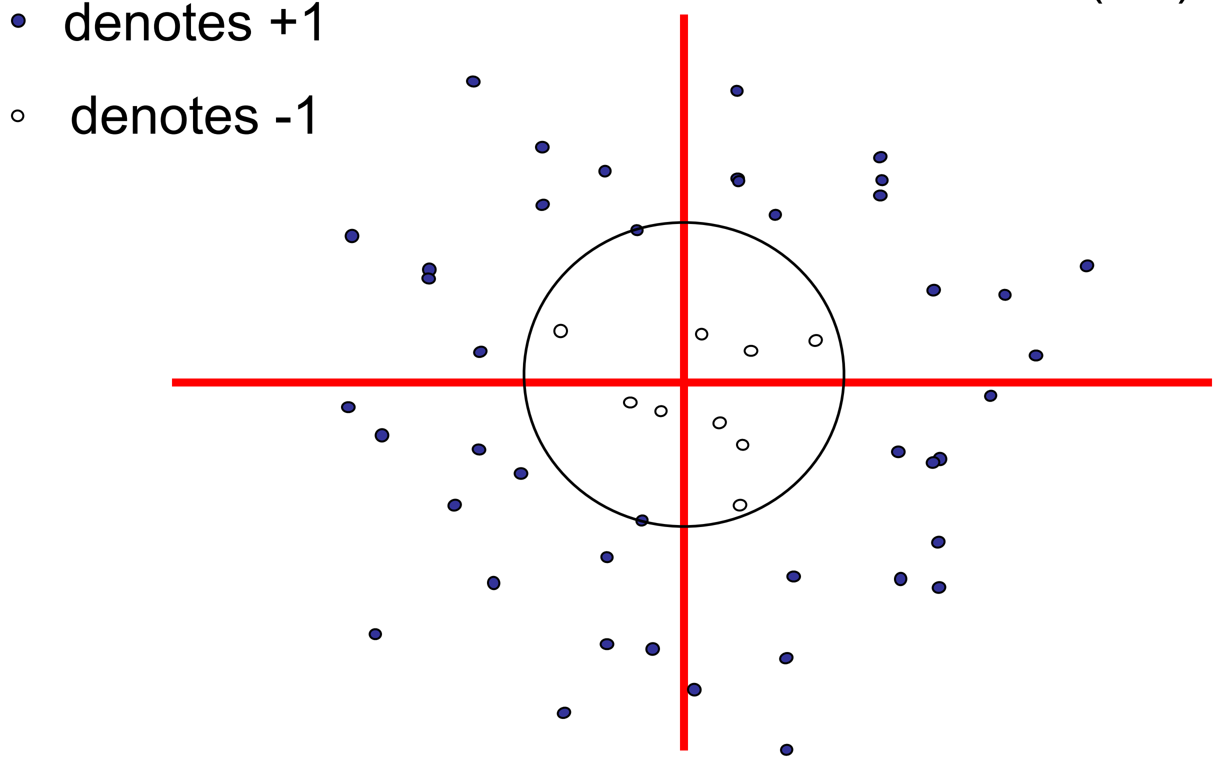
- A learning machine ***f*** takes an input ***x*** and transforms it, somehow using weights $\alpha$, into a predicted output $y^{est}$ = +/- 1



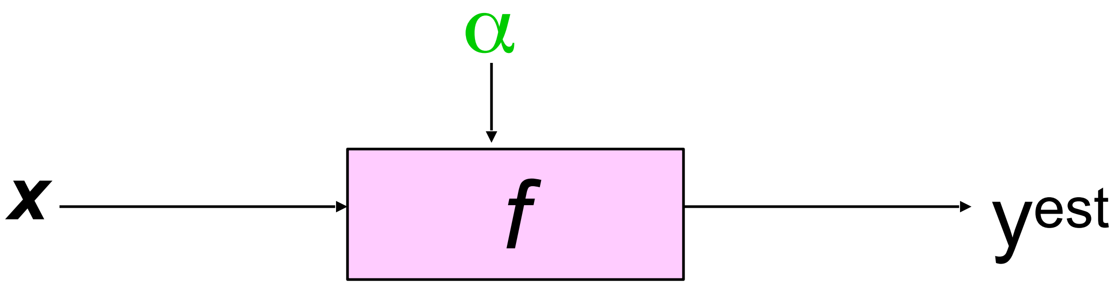
# Examples




$f(x,b) = \text{sign}(x.x - b)$

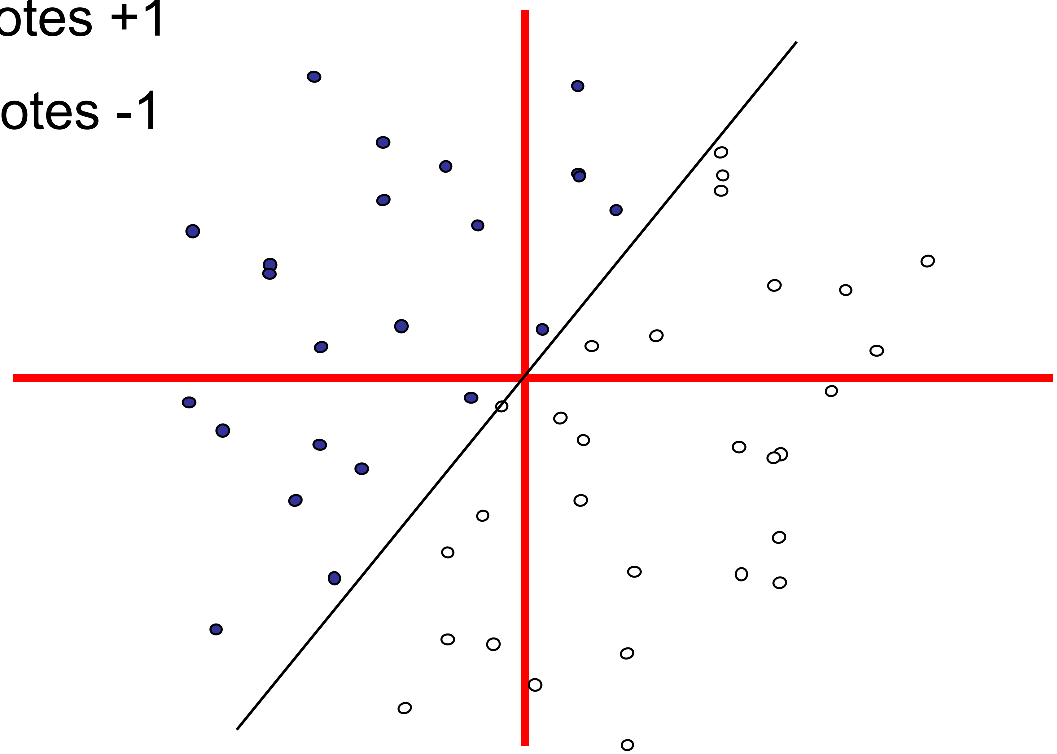


# Examples

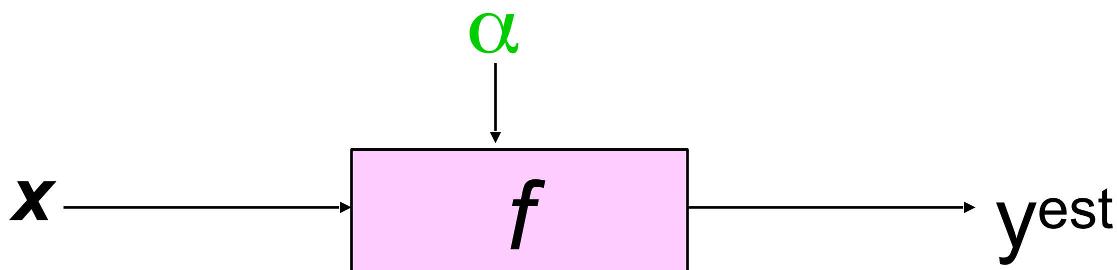



$f(x,w) = \mathrm{sign}(x.w)$

- denotes +1
- denotes -1

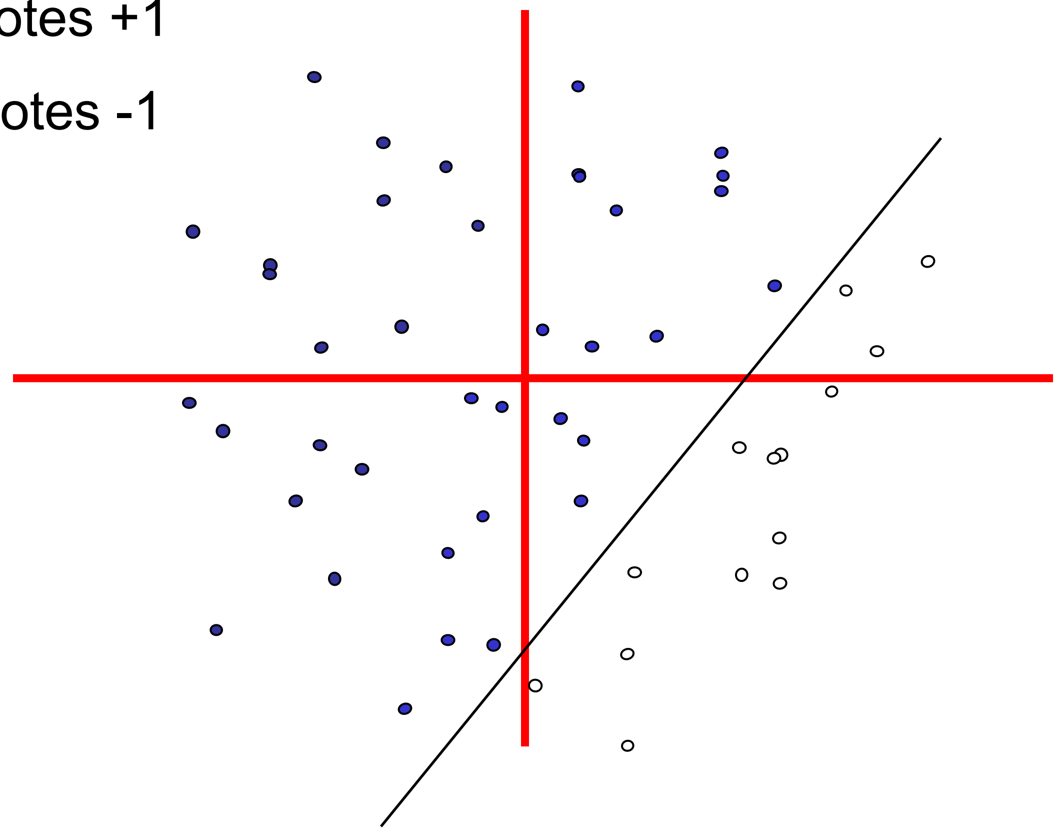


# Examples




$f(x,w,b) = \text{sign}(x.w+b)$

- denotes +1
- denotes -1



# How do we characterize "power"?

- Different machines have different amounts of "power".
- Tradeoff between:
  - More power: Can model more complex classifiers but might overfit.
  - Less power: Not going to overfit, but restricted in what it can model.
- How do we characterize the amount of power?

# Some definitions

- Given some machine **f**
- And under the assumption that all training points *($x_k$,$y_k$)* were drawn i.i.d from some distribution.
- And under the assumption that future test points will be drawn from the same distribution
- Define

$$R(\alpha) = \mathrm{TESTERR}(\alpha) = E\left[\frac{1}{2}\left|y - f(x,\alpha)\right|\right] = \begin{array}{c}\text{Probability of}\\ \text{Misclassification}\end{array}$$

Official terminology

Terminology we'll use

# Some definitions

- Given some machine **f**
- And under the assumption that all training points *$(x_k, y_k)$* were drawn i.i.d from some distribution.
- And under the assumption that future test points will be drawn from the same distribution
- Define

$$R(\alpha) = \text{TESTERR}(\alpha) = E\left[\frac{1}{2}\left|y - f(x,\alpha)\right|\right] = \begin{array}{c}\text{Probability of}\\ \text{Misclassification}\end{array}$$

Official terminology

Terminology we'll use

$$R^{emp}(\alpha) = \text{TRAINERR}(\alpha) = \frac{1}{R}\sum_{k=1}^{R}\frac{1}{2}\left|y_k - f(x_k,\alpha)\right| = \begin{array}{c}\text{Fraction Training}\\ \text{Set misclassified}\end{array}$$

R = #training set data points

# Vapnik-Chervonenkis dimension

$$\text{TESTERR}(\alpha) = E\left[\frac{1}{2}\left|y - f(x,\alpha)\right|\right] \qquad \text{TRAINERR}(\alpha) = \frac{1}{R}\sum_{k=1}^{R}\frac{1}{2}\left|y_k - f(x_k,\alpha)\right|$$

- Given some machine **f**, let *h* be its VC dimension.
- *h* is a measure of **f**'s power (*h* does not depend on the choice of training set)
- Vapnik showed that with probability 1-$\eta$

$$\text{TESTERR}(\alpha) \le \text{TRAINERR}(\alpha) + \sqrt{\frac{h(\log(2R/h)+1) - \log(\eta/4)}{R}}$$

This gives us a way to estimate the error on future data based only on the training error and the VC-dimension of ***f***

# What VC-dimension is used for

$$\text{TESTERR}(\alpha) = E\left[\frac{1}{2}\left|y - f(x,\alpha)\right|\right] \qquad \text{TRAINERR}(\alpha) = \frac{1}{R}\sum_{k=1}^{R}\frac{1}{2}\left|y_k - f(x_k,\alpha)\right|$$

- Given some machine **f**, let *h* be its VC d... ne... io...
- *h* is a measure of **f**'s power.
- Vapnik showed th... wit...

But given machine **f**, how do we define and compute h?

TEST... ...1) $-\log(\eta/4)$ ... $R$

...ay...o estimate the error on ...utu... da... based only on the training error and the VC-dimension of f

# Shattering

- Machine f can *shatter* a set of points $x_1, x_2 .. x_r$ if and only if…

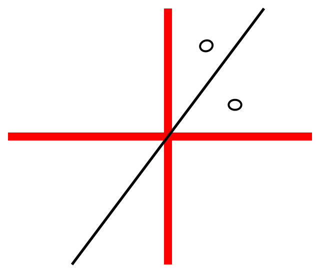
  For every possible training set of the form $(x_1,y_1)$ , $(x_2,y_2)$ ,… $(x_r ,y_r)$

  …There exists some value of $\alpha$ that gets zero training error.

There are $2^r$ such training sets to consider, each with a different combination of +1's and –1's for the y's

# Shattering

- Machine f can *shatter* a set of points $x_1, x_2 .. X_r$ if and only if…
  For every possible training set of the form $(x_1,y_1)$ , $(x_2,y_2)$ ,… $(x_r ,y_r)$
  …There exists some value of $\alpha$ that gets zero training error.
- Question: Can the following f shatter the following points?



f(x,w) = sign(x.w)

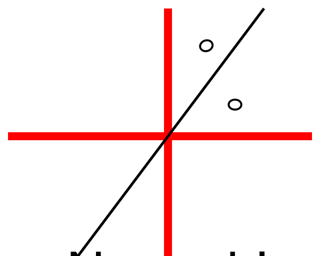
# Shattering

- Machine f can *shatter* a set of points $x_1$, $x_2$ .. $X_r$ if and only if…

  For every possible training set of the form $(x_1,y_1)$ , $(x_2,y_2)$ ,… $(x_r,y_r)$

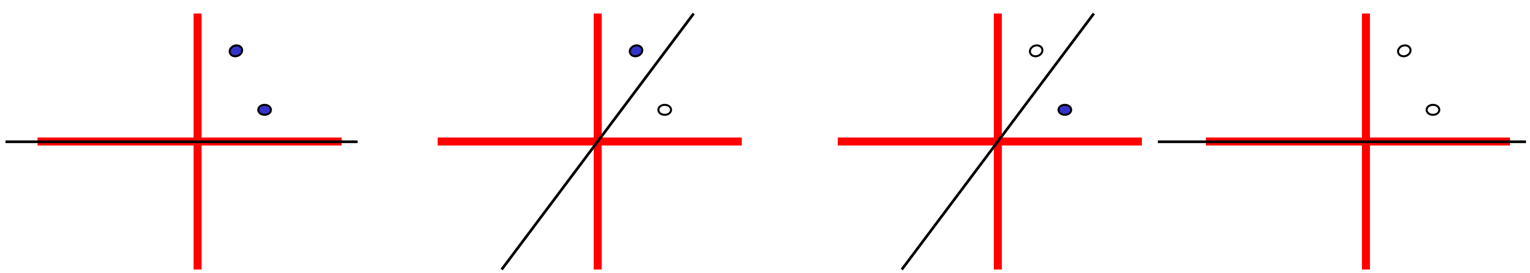
  …There exists some value of $\alpha$ that gets zero training error.

- Question: Can the following f shatter the following points?



$f(x,w) = sign(x.w)$

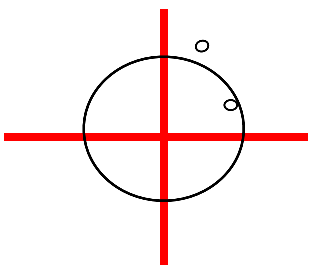
- Answer: No problem. There are four training sets to consider



w=(0,1) w=(-2,3) w=(2,-3) w=(0,-1)

# Shattering

- Machine f can *shatter* a set of points $x_1$, $x_2$ .. $X_r$ if and only if…
  For every possible training set of the form $(x_1,y_1)$ , $(x_2,y_2)$ ,… $(x_r ,y_r)$
  …There exists some value of $\alpha$ that gets zero training error.
- Question: Can the following f shatter the following points?



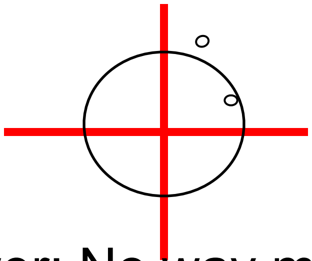
f(x,b) = sign(x.x-b)

# Shattering

- Machine f can *shatter* a set of points $x_1, x_2 .. X_r$ if and only if…

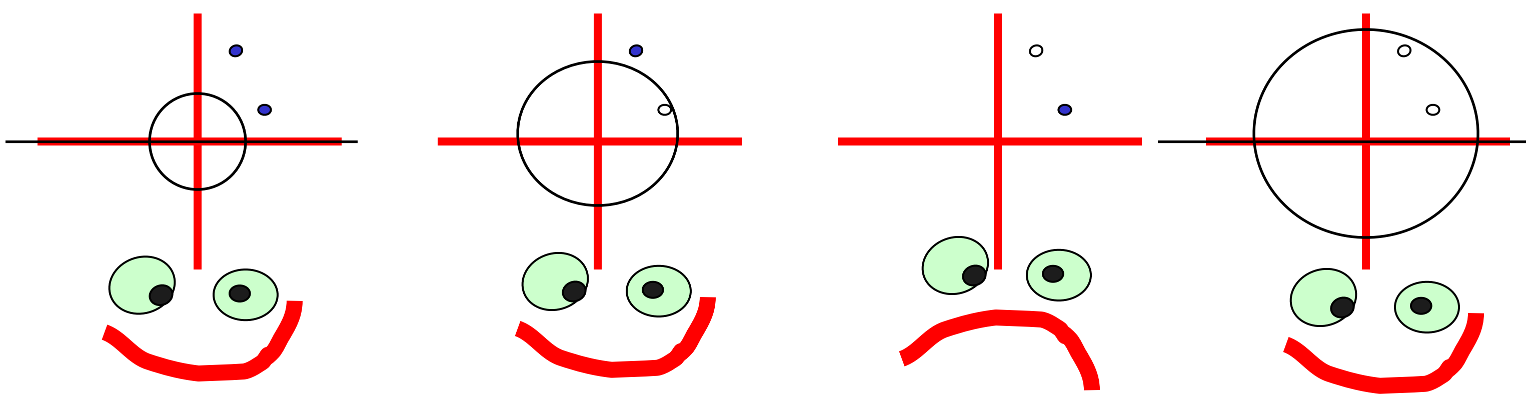
  For every possible training set of the form $(x_1,y_1)$ , $(x_2,y_2)$ ,… $(x_r ,y_r)$

  …There exists some value of $\alpha$ that gets zero training error.
- Question: Can the following f shatter the following points?



$f(x,b) = \text{sign}(x.x-b)$

- Answer: No way my friend.



# Definition of VC dimension

Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: What's VC dimension of f(x,b) = sign(x.x-b)

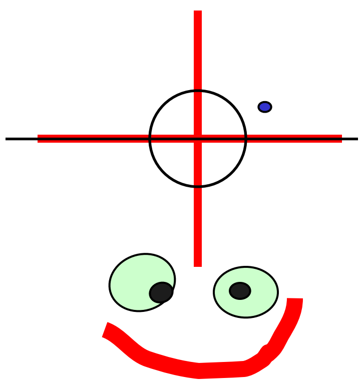
# VC dim of trivial circle

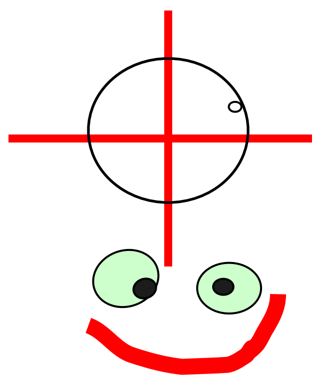
Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: What's VC dimension of f(x,b) = sign(x.x-b)

Answer = 1: we can't even shatter two points! (but it's clear we can shatter 1)





# Reformulated circle

Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: For 2-d inputs, what's VC dimension of f(x,q,b) = sign(qx.x-b)

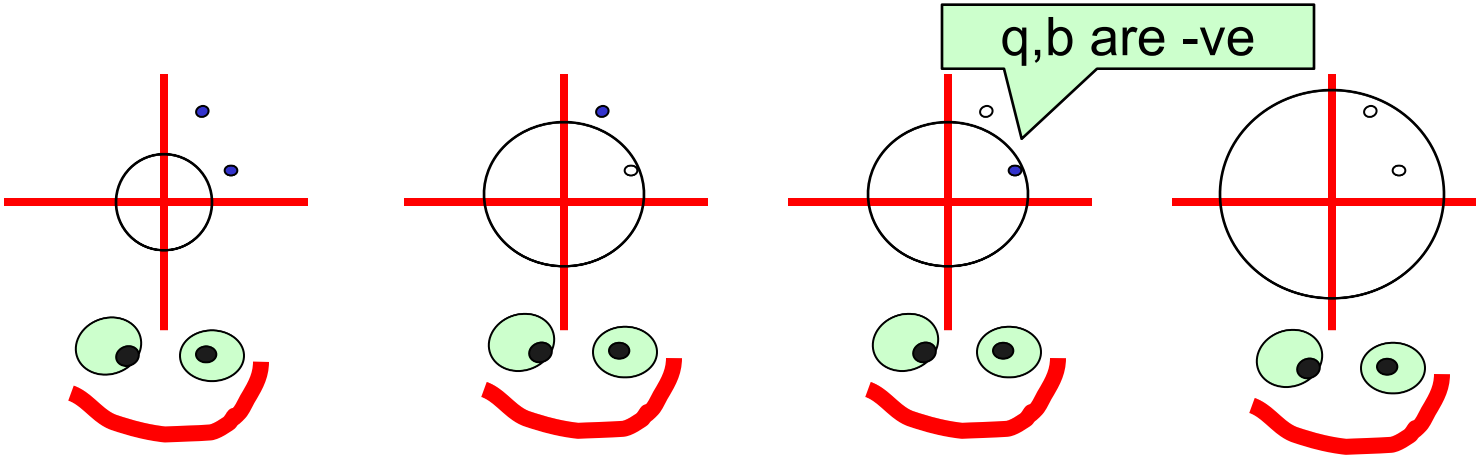
# Reformulated circle

Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: What's VC dimension of f(x,q,b) = sign(qx.x-b)

- Answer = 2

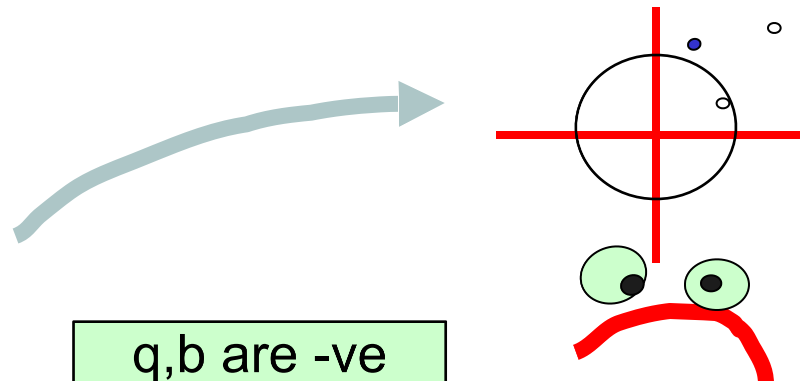


# Reformulated circle

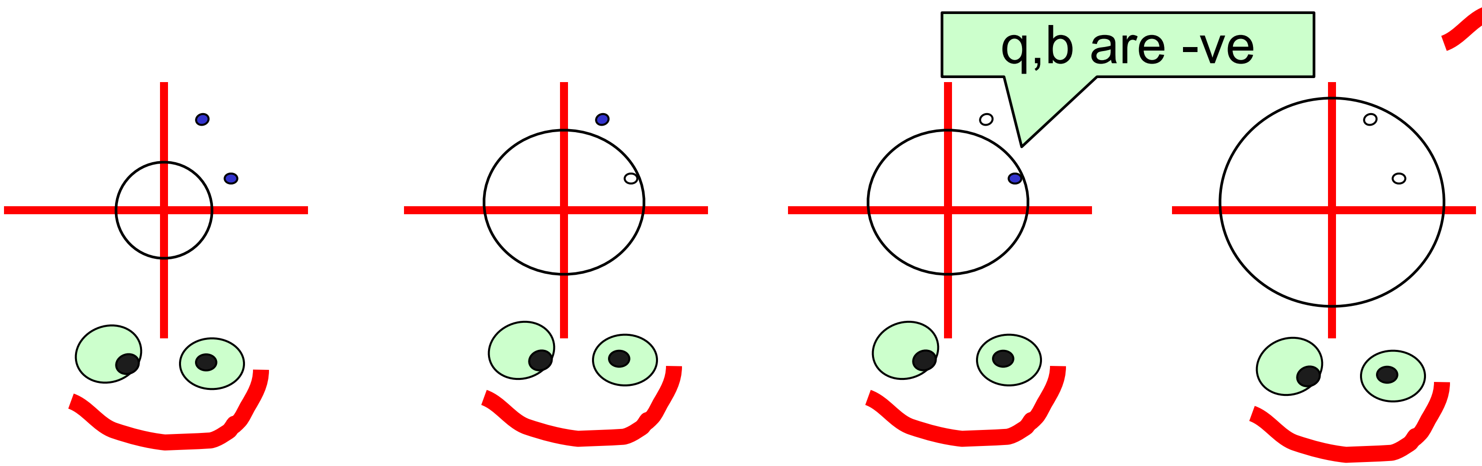
Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: What's VC dimension of f(x,q,b) = sign(qx.x-b)



- Answer = 2 (clearly can't do 3)



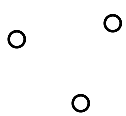
# VC dim of separating line

Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: For 2-d inputs, what's VC-dim of f(x,**w**,b) = sign(**w**.x+b)?

Well, can f shatter these three points?



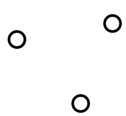
# VC dim of line machine

Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: For 2-d inputs, what's VC-dim of $f(\mathbf{x},\mathbf{w},b) = \mathrm{sign}(\mathbf{w}.\mathbf{x}+b)$?

Well, can f shatter these three points?



Yes, of course.

All -ve or all +ve is trivial

One +ve can be picked off by a line

One -ve can be picked off too.

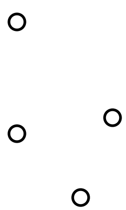
# VC dim of line machine

Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: For 2-d inputs, what's VC-dim of f(x,**w**,b) = sign(**w**.x+b)?

Well, can we find four points that ***f*** can shatter?



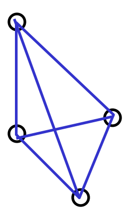
# VC dim of line machine

Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: For 2-d inputs, what's VC-dim of f(x,**w**,b) = sign(**w**.x+b)?

Well, can we find four points that ***f*** can shatter?



Can always draw six lines between pairs of four points.

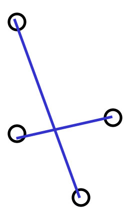
# VC dim of line machine

Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: For 2-d inputs, what's VC-dim of f(x,**w**,b) = sign(**w**.x+b)?

Well, can we find four points that ***f*** can shatter?



Can always draw six lines between pairs of four points.

Two of those lines will cross.

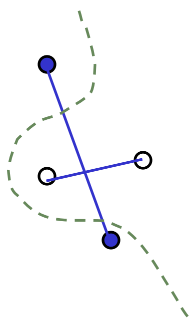
# VC dim of line machine

Given machine ***f***, the VC-dimension *h* is

The maximum number of points that can be arranged so that ***f*** shatter them.

Example: For 2-d inputs, what's VC-dim of f(x,**w**,b) = sign(**w**.x+b)?

Well, can we find four points that ***f*** can shatter?



Can always draw six lines between pairs of four points.

Two of those lines will cross.

If we put points linked by the crossing lines in the same class they can't be linearly separated

So a line can shatter 3 points but not 4

So VC-dim of Line Machine is 3

# VC dim of linear classifiers in m-dimensions

If input space is m-dimensional and if **f** is *sign*(***w***.***x***-*b*), what is the VC-dimension?

Proof that *h* >= *m*: Show that *m* points can be shattered

*Can you guess how?*

# VC dim of linear classifiers in m-dimensions

If input space is m-dimensional and if **f** is *sign*(**w**.**x**-*b*), what is the VC-dimension?

Proof that $h >= m$: Show that $m$ points can be shattered

Define m input points thus:

$\mathbf{x}_1 = (1,0,0,\ldots,0)$
$\mathbf{x}_2 = (0,1,0,\ldots,0)$
:
$\mathbf{x}_m = (0,0,0,\ldots,1)$ So $x_k[j] = 1$ if $k=j$ and $0$ otherwise

Let $y_1, y_2, \ldots y_m$ , be any one of the $2^m$ combinations of class labels.

Guess how we can define $w_1, w_2, \ldots w_m$ and $b$ to ensure sign($\mathbf{w}$. $\mathbf{x}_k$ + b) = $y_k$ for all $k$ ? Note:

$$\text{sign}(\mathbf{w}.\mathbf{x}_k + b) = \text{sign}\left( b + \sum_{j=1}^{m} w_j . x_k[j] \right)$$

# VC dim of linear classifiers in m-dimensions

If input space is m-dimensional and if **f** is *sign*($\mathbf{w}.\mathbf{x}-b$), what is the VC-dimension?

Proof that $h >= m$: Show that $m$ points can be shattered

Define m input points thus:

$\mathbf{x}_1 = (1,0,0,\ldots,0)$
$\mathbf{x}_2 = (0,1,0,\ldots,0)$
:
$\mathbf{x}_m = (0,0,0,\ldots,1)$ So $x_k[j] = 1$ if $k=j$ and $0$ otherwise

Let $y_1, y_2, \ldots y_m$, be any one of the $2^m$ combinations of class labels.

Guess how we can define $w_1, w_2, \ldots w_m$ and $b$ to ensure $\text{sign}(\mathbf{w}.\ \mathbf{x}_k + b) = y_k$ for all $k$ ? Note:

$$\text{sign}(\mathbf{w}.\mathbf{x}_k + b) = \text{sign}\left( b + \sum_{j=1}^{m} w_j \, . \, x_k[j] \right)$$

Answer: $b=0$ and $w_k = y_k$ for all $k$.

# VC dim of linear classifiers in m-dimensions

If input space is m-dimensional and if **f** is *sign*(***w***.***x***-*b*), what is the VC-dimension?

- Now we know that h >= m
- In fact, h=m+1
- Proof that h >= m+1 is easy
- Proof that h < m+2 is moderate

# What does VC-dim measure?

- Is it the number of parameters?
  
  Related but not really the same.
- I can create a machine with one numeric parameter that really encodes 7 parameters (How?)
- And I can create a machine with 7 parameters which has a VC-dim of 1 (How?)
- *Andrew's private opinion: it often is the number of parameters that counts.*

# Structural Risk Minimization

- Let $\phi(f)$ = the set of functions representable by f.
- Suppose $\varphi(f_1) \subseteq \varphi(f_2) \subseteq \boxed{?} \varphi(f_n)$
- Then $h(f_1) \leq h(f_2) \leq \boxed{?} h(f_n)$ *(Hey, can you formally prove this?)*
- We're trying to decide which machine to use.
- We train each machine and make a table…

$$\text{TESTERR}(\alpha) \leq \text{TRAINERR}(\alpha) + \sqrt{\frac{h(\log(2R/h)+1) - \log(\eta/4)}{R}}$$

| $i$ | $f_i$ | TRAINER R | VC-Conf | Probable upper bound on TESTERR | Choice |
|---|---|---|---|---|---|
| 1 | $f_1$ | | | | |
| 2 | $f_2$ | | | | |
| 3 | $f_3$ | | | | Ö |
| 4 | $f_4$ | | | | |
| 5 | $f_5$ | | | | |
| 6 | $f_6$ | | | | |

# Using VC-dimensionality

That's what VC-dimensionality is about

People have worked hard to find VC-dimension for..

- Decision Trees
- Perceptrons
- Neural Nets
- Decision Lists
- Support Vector Machines
- And many many more

All with the goals of

1. Understanding which learning machines are more or less powerful under which circumstances
2. Using Structural Risk Minimization for to choose the best learning machine

# Alternatives to VC-dim-based model selection

- What could we do instead of the scheme below?

$$\text{TESTERR}(\alpha) \leq \text{TRAINERR}(\alpha) + \sqrt{\frac{h(\log(2R/h)+1)-\log(\eta/4)}{R}}$$

| $i$ | $f_i$ | TRAINERR | VC-Conf | Probable upper bound on TESTERR | Choice |
|---|---|---|---|---|---|
| 1 | $f_1$ | | | | |
| 2 | $f_2$ | | | | |
| 3 | $f_3$ | | | | Ö |
| 4 | $f_4$ | | | | |
| 5 | $f_5$ | | | | |
| 6 | $f_6$ | | | | |

# Alternatives to VC-dim-based model selection

- What could we do instead of the scheme below?
    1. Cross-validation

| $i$ | $f_i$ | TRAINERR | 10-FOLD-CV-ERR | Choice |
|---|---|---|---|---|
| 1 | $f_1$ | | | |
| 2 | $f_2$ | | | |
| 3 | $f_3$ | | | Ö |
| 4 | $f_4$ | | | |
| 5 | $f_5$ | | | |
| 6 | $f_6$ | | | |

# Alternatives to VC-dim-based model selection

- What could we do instead of the scheme below?
  1. Cross-validation
  2. AIC (Akaike Information Criterion)

As the amount of data goes to infinity, AIC promises* to select the model that'll have the best likelihood for future data

*Subject to about a million caveats

$$\text{AICSCORE} = LL(Data \mid \text{MLE params}) - (\#\,\text{parameters})$$

| $i$ | $f_i$ | LOGLIKE(TRAINERR) | #parameters | AIC | Choice |
|---|---|---|---|---|---|
| 1 | $f_1$ | | | | |
| 2 | $f_2$ | | | | |
| 3 | $f_3$ | | | | |
| 4 | $f_4$ | | | | Ö |
| 5 | $f_5$ | | | | |
| 6 | $f_6$ | | | | |

# Alternatives to VC-dim-based model selection

- What could we do instead of the scheme below?
  1. Cross-validation
  2. AIC (Akaike Information Criterion)
  3. BIC (Bayesian Information Criterion)

As the amount of data goes to infinity, BIC promises* to select the model that the data was generated from. More conservative than AIC.

*Another million caveats

$$\text{BICSCORE} = LL(Data \mid \text{MLE params}) - \frac{\#\text{ params}}{2} \log R$$

| $i$ | $f_i$ | LOGLIKE(TRAINERR) | #parameters | BIC | Choice |
|---|---|---|---|---|---|
| 1 | $f_1$ | | | | |
| 2 | $f_2$ | | | | |
| 3 | $f_3$ | | | | Ö |
| 4 | $f_4$ | | | | |
| 5 | $f_5$ | | | | |
| 6 | $f_6$ | | | | |

# Which model selection method is best?

1. (CV) Cross-validation
2. AIC (Akaike Information Criterion)
3. BIC (Bayesian Information Criterion)
4. (SRMVC) Structural Risk Minimize with VC-dimension

- AIC, BIC and SRMVC have the advantage that you only need the training error.
- CV error might have more variance
- SRMVC is wildly conservative
- Asymptotically AIC and Leave-one-out CV should be the same
- Asymptotically BIC and a carefully chosen k-fold should be the same
- BIC is what you want if you want the best structure instead of the best predictor (e.g. for clustering or Bayes Net structure finding)
- Many alternatives to the above including proper Bayesian approaches.
- It’s an emotional issue.

# Extra Comments

- Beware: that second “VC-confidence” term is usually very very conservative (at least hundreds of times larger than the empirical overfitting effect).
- An excellent tutorial on VC-dimension and Support Vector Machines (which we’ll be studying soon):

    C.J.C. Burges. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2(2):955-974, 1998. http://citeseer.nj.nec.com/burges98tutorial.html

# What you should know

- The definition of a learning machine: $\boldsymbol{f}(\boldsymbol{x},\alpha)$
- The definition of Shattering
- Be able to work through simple examples of shattering
- The definition of VC-dimension
- Be able to work through simple examples of VC-dimension
- Structural Risk Minimization for model selection
- Awareness of other model selection methods