6.034 Lecture

11/2

Support Vector Machines — Vapnik's miracle

- ☐ Decision Boundaries
- ☐ The widest Street
- ☐ The math ⇝ kernel functions
- ☐ The story
- ✱ How Discovery works
- ✱ Derived features

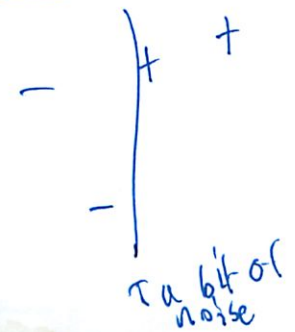
Music: Beatles Sgt. Peppers

This topic takes us back to NN, DB, but also focusing on problem trying to solve, don't become fixated on it

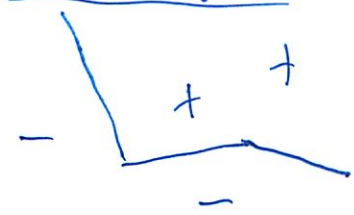
Vapnik's introduced this in 1993
└ Support Vector Machines

lots of people use it now

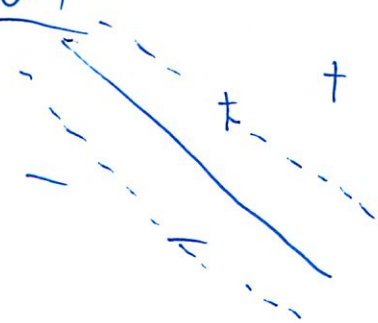
Classification Trees



Nearest Neighbors



SVM



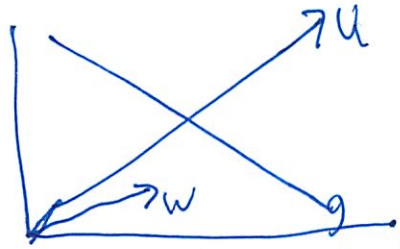
(2)

Look for widest street

Points from beyond street don't matter

The ones on the street are called <u>support vectors</u>

(1) <u>Decision Rule</u>:



w is normal to a line

Project u line onto w

$$\|\bar{w}\|\,\|\bar{u}\|\cos\theta = \bar{w}\cdot\bar{u}$$

↑ magnitude of

$\bar{w}\cdot\bar{u} > c$ then u should be (+)

$\bar{w}\cdot u + b > 0$ then (+) $\quad b = -c$

(2) <u>Constraints</u>

$$\bar{w}\cdot\bar{x}_+ + b \geq 1$$

$$\bar{w}\cdot\bar{x}_- + b \leq 1$$

Not quite convinent form for us

New variable $\quad y = \begin{cases} 1 & (+) \quad = y_+ \\ -1 & (-) \quad = y_- \end{cases}$

③

Multiply constraints by Y

$$Y_+(\bar{w} \cdot \bar{x}_+ + b) \geq 1$$

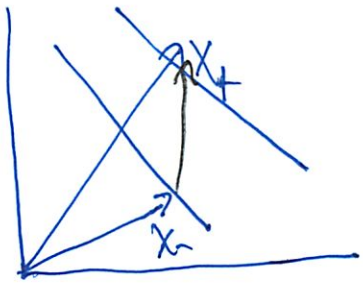
$$Y_-(\bar{w} \cdot \bar{x}_- + b) \geq 1$$

Can take all this and rewrite

$$\boxed{Y_i(\bar{w} \cdot \bar{x}_i + b) - 1 \geq 0}$$ ← important

└ if outside street will get 1 or -1
if just inside street - will get < 1 ?

Figure out how wide street is
└ and gutters



Can calculate $x_+ - x_-$ (black line)

Take dot product of w/ unit vector

$$(\bar{x}_+ - \bar{x}_-) \cdot \frac{\bar{w}}{\|\bar{w}\|} = \text{width}$$

(4)

$$\left.\begin{aligned} \bar{w} \cdot \bar{x}_+ &= 1 - b \\ \bar{w} \cdot x_- &= -1 - b \end{aligned}\right\} \quad \frac{1}{\|\bar{w}\|}\underbrace{\left([1-b]-[-1-b]\right)}_{2}$$

So width of street

$$(\bar{x}_+ - \bar{x}_-)\frac{\bar{w}}{\|\bar{w}\|} = \frac{2}{\|\bar{w}\|}$$

So want to maximize $\frac{2}{\|\bar{w}\|}$

Can invert $\frac{\|\bar{w}\|}{2}$

And minimize $\|\bar{w}\|$

Can convert to $\frac{1}{2}\|\bar{w}\|^2$

- width will be maximum when minimizing this
- but <u>within</u> constraints

Can set up Lagrangean

$$L = \frac{1}{2}\|\bar{w}\|^2 - \sum_i \alpha_i \left[y_i (\bar{w} \cdot \bar{x}_i + b) - 1 \right]$$

Differentiate

$$\frac{\partial L}{\partial \bar{w}} = \bar{w} - \sum_i \alpha_i y_i \bar{x}_i = 0$$

differentiating vector

(5)

$$\boxed{\bar{w} = \sum_i \alpha_i y_i \bar{x}_i}$$

↑ w is linear sum of ⊕/⊖ samples

α will be 0 except for lines on gutter

↳ our <u>support vectors</u>

Now differentiate w/ c to b

$$\frac{\partial L}{\partial b} = -\sum \alpha_i y_i = 0 = \sum \alpha_i y_i = 0$$

↑ (undetermind symbol)

Can plug $\bar{w}$ back into Lagrangian

$$L = \frac{1}{2}\left(\sum_i \alpha_i y_i x_i\right)\left(\sum_j \alpha_j y_j x_j\right) - \sum_i \alpha_i y_i x_i \left(\sum_j \alpha_j y_j x_j\right) - b\sum \alpha_i y_i + \sum \alpha_i$$

We know some parts $= 0$

Can rewrite

$$= \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i x_j - \sum_i \sum_j \alpha_i \alpha_j y_i y_j \bar{x}_i \bar{x}_j + \sum_i \alpha_i$$

(6)

Can cross some terms out
See that it is Lagrangian

Can rewrite decision rule

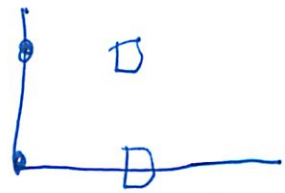
$$\sum \alpha_i \underbrace{y \bar{x}_i \vec{u}}_{\text{depends only on samples dotted w/ unknowns}} + b \geq 0 \quad +$$

Numerical analysts can find this w/ quadratic optimization problem → means no local maximum

If doesn't work - it will try a lot and eventually give up

Why does everyone use?

- Dependency on dot product



Not linearly seperable
But could be seperable in 3rd dimension

(7)

$\bar{z} = \phi(\bar{x})$

$\phi[\bar{x}_i] \cdot \phi(\bar{x}_j) = k(\bar{x}_i, \bar{x}_j)$

$\phi[\bar{x}_i] \cdot \phi[\bar{u}] = k(\bar{x}_i, \bar{u})$

Don't need ϕ - just need k

- the kernal function - charactizes internally some transformation we might not know

One popular k

$$k = e^{\frac{\|x_i - x_j\|^2}{\sigma}}$$

? constant - distance each sample has influence

Here hard problems may be seperable - in our new plane

In original plane - may be circle

(8)

This was actually in 60s

But iron curtain prevented spread

Trounces neural net guys

Had to find new kernal to get it to work

6.034 Meg4
Recitation

11/4

Hardest topic in class

Support Vector Machines - seperate ⊕ ⊖ w/ as wide a margin as possible

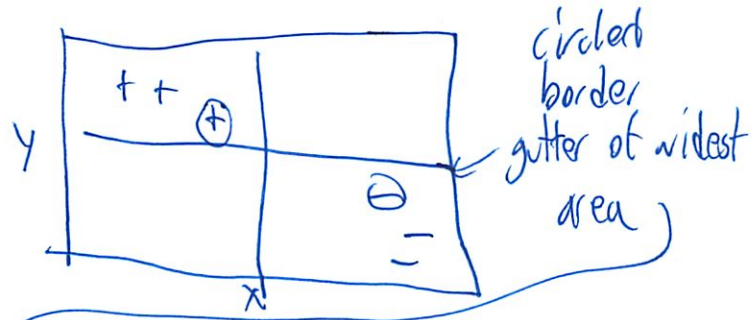
figure out sol w/ widest road in between

Computer optimizes over the αs

- complicated

Silver Star

* Avoid the system $\frac{1}{\|w\|} = d$
* k is the new dot product if Φ is applied to all vectors
* α of non support $= 0$



these are called the support vectors

- look like points
- but every point is endpoint of vector from origin to point

②

Represented by α score

What computer optimizes over

α = how much a particular point is constraining the width of the road

high = very constrained

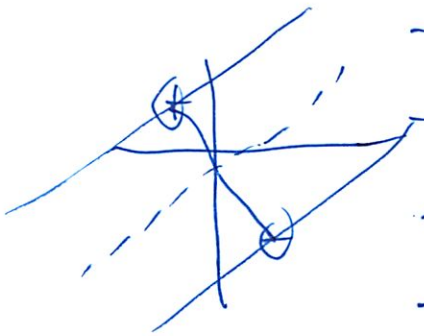
low = not constrained

0 = does not relate to road

can remove w/o changing road

every other pt except support vectors

Can draw line b/w support vectors



- for own use
- can write lines' eq'n by reading tick marks
- then draw dotted line perp. in middle
- then draw parallel lines through ⊕ ⊖

For whats called kernal

└ here just using linear kernal

Dotted line is what divides <u>new points</u>

Should be no points already in road

(3)

A few eq'n to solve for

$W_1 x + W_2 y + b = 0$ for ~~support~~ dotted line
↳ eq for line

x, y here are any pt on dotted line

Quiz often asks solve for w, b

$W_1 x + W_2 y + b = 1$ for support ⊕ solid line x, y here are ⊕ support vector

$W_1 x + W_2 y + b = -1$ for " ⊖ " "

Can plug in x, y from each + solve system of eq

α is what pushes road together

$\sum \alpha_{\oplus} = \sum \alpha_{\ominus}$ remember only support vectors matter

$\bar{W} = \sum \alpha_+ \bar{x}_+ - \sum \alpha_- \bar{x}_-$

↑ can solve for weights like this but hard

Usually no more than 2 points

(4)

Shortcut

d is half of perpendicular bisector

$$d = \frac{1}{\|w\|}$$

midpoint

$$w_1 x_1 + w_2 y_1 + b = 0$$

$$w_1 x_2 + w_2 y_2 + b = 1$$

$\oplus$ support vector

$\bar{w} \cdot \bar{x}_1$ ← vectors (dot product of)

$$\bar{w} \cdot \underbrace{(\bar{x}_1 - \bar{x}_2)}_{D} = 1$$

$$(\bar{x}_1 - \bar{x}_2) = \frac{1}{\|\bar{w}\|}$$

$$d = \frac{1}{\|w\|}$$

So can easily get other ans now

Dotted line $\Rightarrow y = x - 1$

Infinite # of w, b - lots are multiples of others

- you need to know the correct one
- not nessarily the smallest

5

Quiz asked for w_1 w_2 b

Claiming its

$$ky = kx - k$$

← multiplying by k

$y = x - 1$ ← our line

In their way

$$-kx + ky + k = 0$$ ← for dotted line

Can we calculate D

- maybe not
- since its not an integer
- get prob $2D$ instead

distance formula

$$2D = \sqrt{32}$$

$\sqrt{4^2 + 4^2}$

across up

$$D = 2\sqrt{2} = \frac{1}{\|w\|}$$

So $\|w\| = \frac{\sqrt{2}}{4}$

~~Now need k~~

$$\|w\| = \sqrt{2k^2}$$

k is $\sqrt{w_1^2 + w_2^2}$

(6)

So $\bar{w} = \begin{bmatrix} -1/4 \\ 1/4 \end{bmatrix}$ $b = 1/4$

But it could be all $\ominus$

$\frac{1}{4}$ - go up so y component should be $\oplus$

Now can use - to solve α

$$\bar{w} = \sum \alpha_+ \bar{x}_+ - \sum \alpha_- x_1$$

$$\begin{bmatrix} -\frac{1}{4} \\ \frac{1}{4} \end{bmatrix} = \alpha \begin{bmatrix} -1 \\ 2 \end{bmatrix} - \alpha \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$

$$= \begin{bmatrix} -4 \\ 4 \end{bmatrix} \alpha$$

So $\alpha = \frac{1}{16}$

Opposite of what computer actually does

Q: Conceptually what is w

w is weight vector

dot w/ vectors on boundry line, support vector

α is actually what is important

just weights

(7)

$\boxed{\sqrt{2k^2}}$

Since $\underset{w_1}{-kx} + \underset{w_2}{ky} - k = 0$

Close to

but $w_1 = -k$

$w_2 = k$

$b = k$

So therefore take magnitude

$\sqrt{(-k)^2 + (k^2)}$

$\sqrt{2k^2}$ ← Value only for this problem

- same procedure later

W should point to +

+ on left should be positive

If you can multiply by -1 - then you messed up

Always want line to point ⊕ if left

Want to classify ⊕ as ⊕

(8) <u>Kernal</u>

have Φ

in happy world linear kernals
just dot producting
$w \cdot x$

What it wanted circle or other figure

Where Φ comes in

Can apply Φ to each vector to transform dimension

Φ can be hard to solve for

Secret w/ k - k is new dot product in
world of Φ

So when doing SVM - only need to do

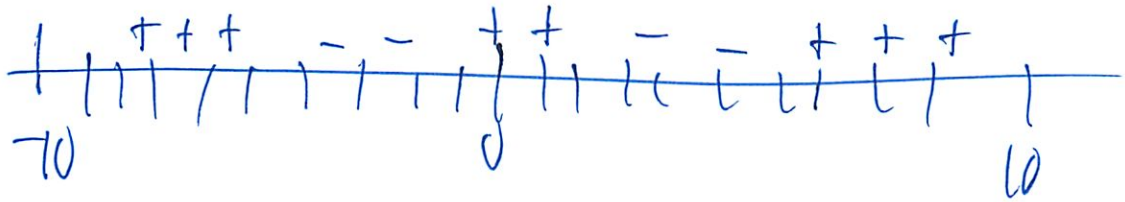
$$k(\vec{u}, \vec{v}) = \Phi(\vec{u}) \cdot \Phi(\vec{v})$$

But don't need to use Φ fn

(9)

Lots of qv are k if have Φ
Φ if have k

i Can draw on # line
- too long?

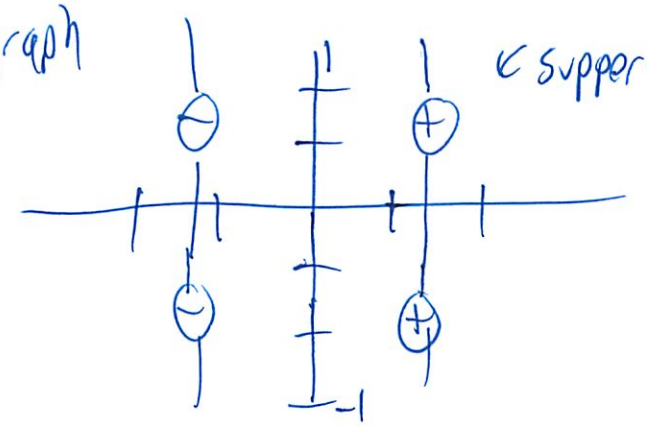


$$k(u,v) = \cos\left(\frac{\pi}{4}u\right)\cos\left(\frac{\pi}{4}v\right) + \sin\left(\frac{\pi}{4}u\right)\sin\left(\frac{\pi}{4}v\right)$$

Looks kinda like a dot product

$$\Phi(u) = \begin{bmatrix} \cos\left(\frac{\pi}{4}u\right) \\ \sin\left(\frac{\pi}{4}u\right) \end{bmatrix}$$

Can graph



← support vectors

(10)

Can do 06, 08 on own

- pictures at end

etc

6.034 Lecture

11/7

Bulldozers vs Scalpels

One shot learning from examples

- Trains + Arches
- Arch Heuristics
- Felicity Conditions
- How to package your idea

Have the students given their grades by their parents

* Talk to yourself
* Winstons'

You can learn by figuring out or asking others

This is the most important lecture at MIT

Most methods need a ton of data to learn

But what would a martian do with only a few examples

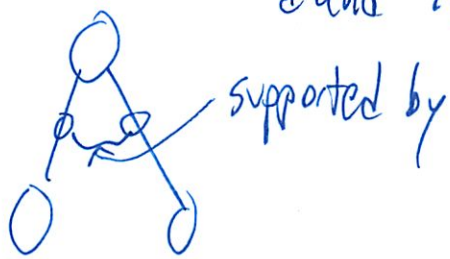
What is an arch?

1. Model 1 "Arch" 
   - what is it about this?
   - that all are white?
   - that all are square?

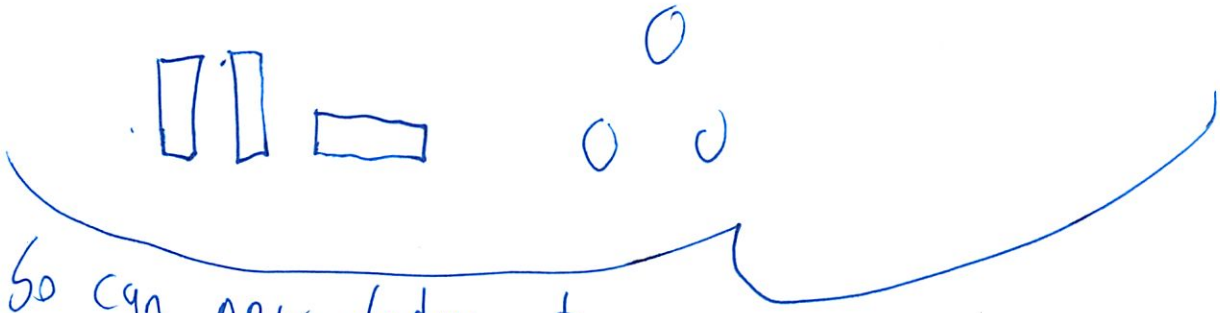
(2)

Represent w/ nodes

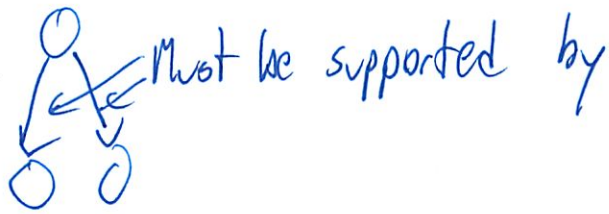
and relationships



2. Near Miss

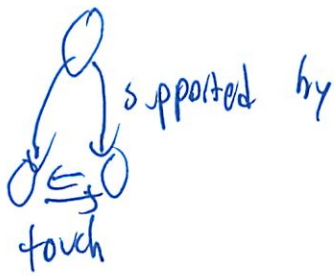


So can now deduce to



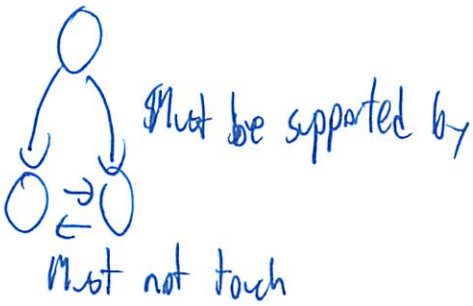
- with only 2 examples

3. Near Miss

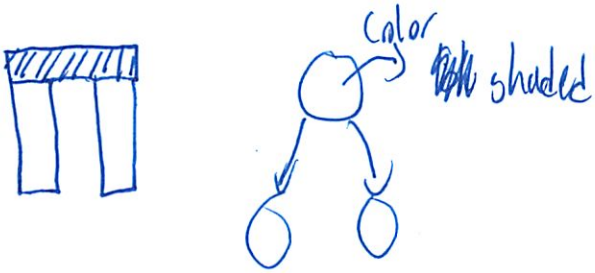


(3)

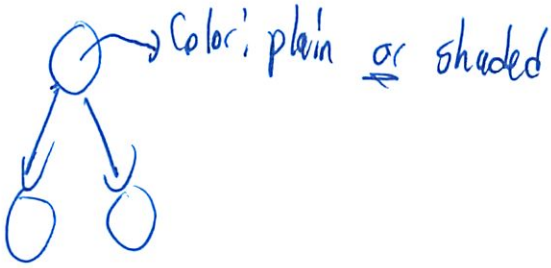
So now deduce



4. Arch

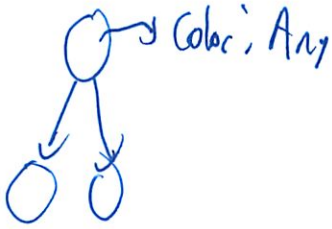


So now deduce



5. Arch

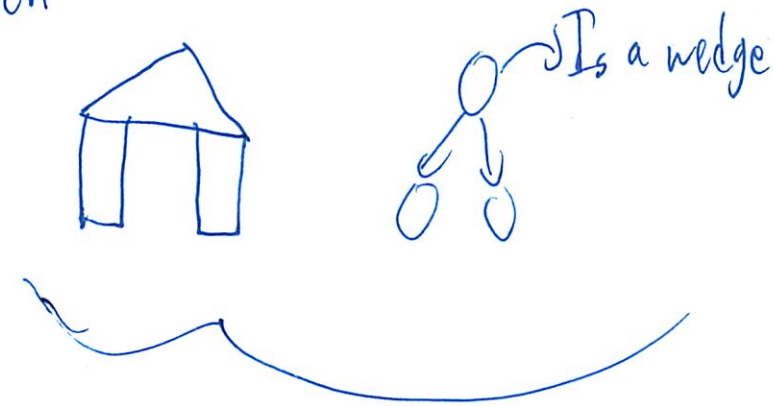
Deduce



only possibilites
- clear
- shaded
- crosshatched

(4)

5. Arch



Deduce

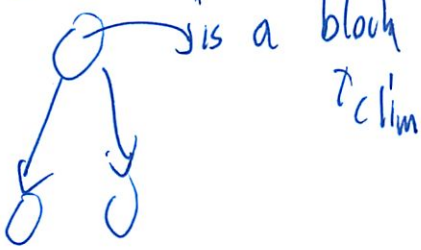
We know:

Brick → Block → Toy → etc...

Wedge → Block

↑ we decided to have a relation

Deduce



↑ climb 1 step

Generalization or Specialization process:

Both

depends on which steps

my defs

Specialization - ~~get~~ get more specific
like saying must be supported by

Generalization - when have enough info can make generalization
like saying any color

(5)

Ahhh

If saw a near miss → specialization
" arch / "ok" → generalization

Names for steps

1. Require Link
2. Forbid Link
3. Extend Set
4. Drop Link
   - must leave color * to know we figured that out
5. Climb tree

↑ refers to each example

Train Car recognition example

- do what we did today
- plus Sussman / Yip example from before

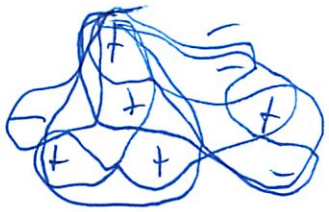
(+) ← seed

\- + \- + + \- \- + \- generalize/specialize to match only the + , not the −

so step, by step refine your results

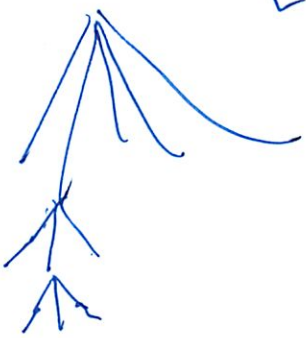
(6)

Have lots of possible choices →

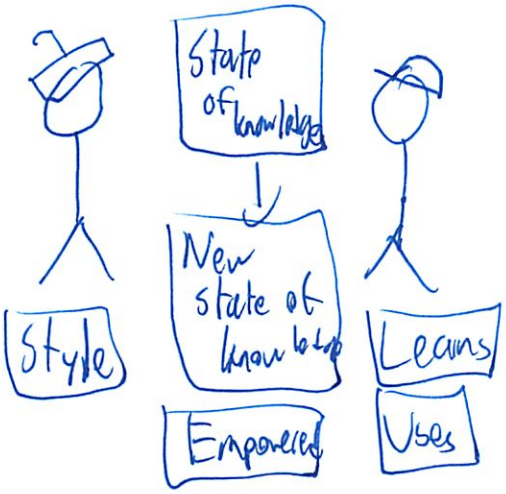


└ Turn into search

└ beam search



Relationship b/w Teacher + Student



Conditions for state transition

prof ← trust student

← style

how they learn

how they can use

(7)

Have the students given their grades by / to their parents ↓ incorrect

- we know sentence is wrong
- but how do we put it to use

Go one way if student is human
other computer

If student thinks its near miss → student must engage with the material

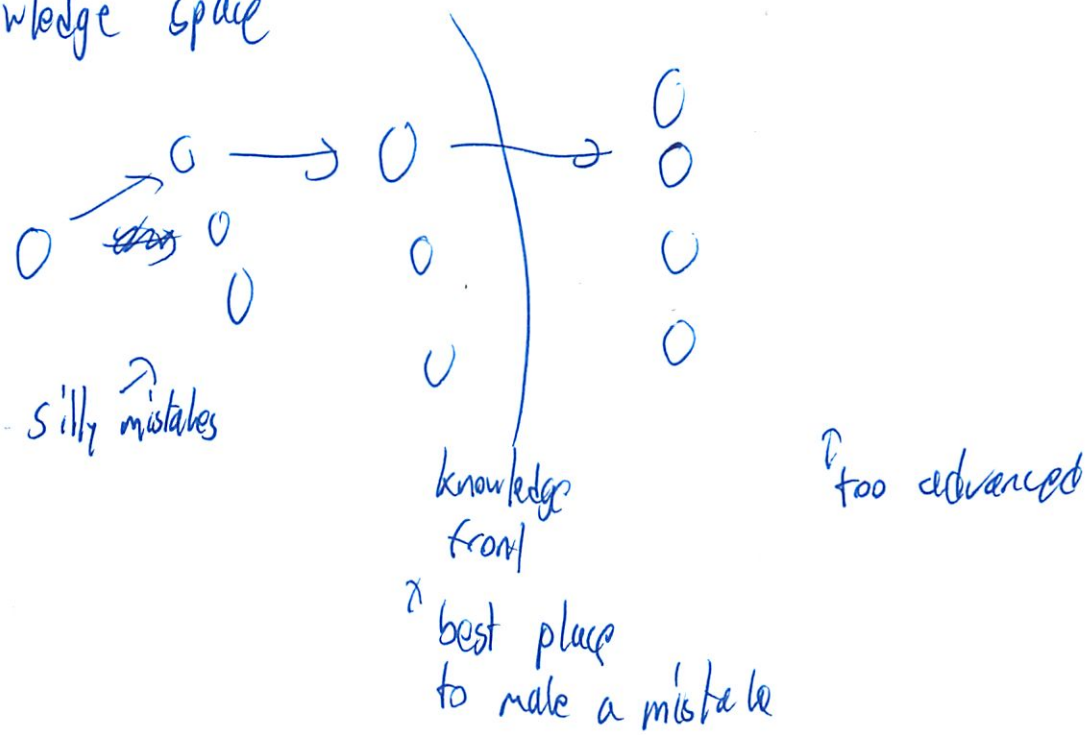
Self explination pre-8.01 quizzes
↳ predicted 8.01 grades
- best people said a lot about problem
  - How am I doing
  - I'm stuck
  - F = ma
- People who do well talk to themselves a lot
- Talk more - it provides index to what you already learned
  ↳ helps to work in groups
  - talk to yourself

8)

- taking notes
  - even if you never look at

TAs best teachers since best model of student
- You start w/ you
  - and then layer differences on top

Must ~~[illegible]~~ discover where student is in knowledge space

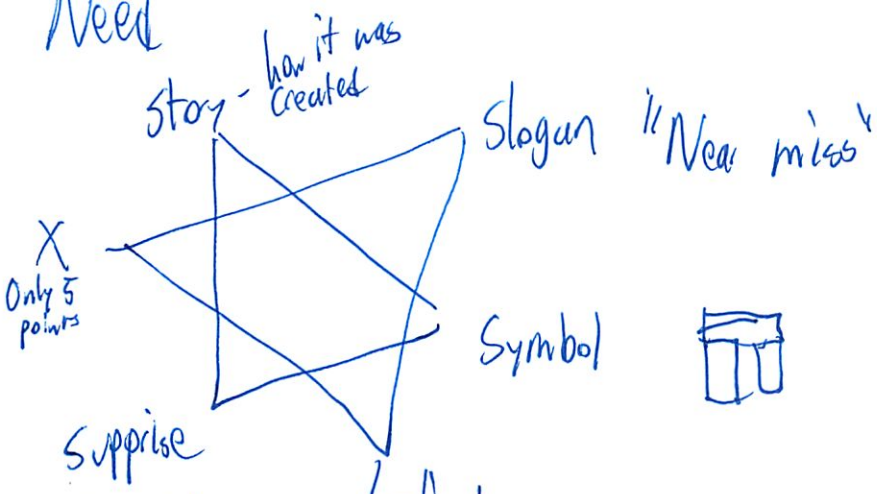


<u>Packaging</u>

This stuff became famous in AI immediately
- Prof Winston didn't know how to deal with

(9)

Need



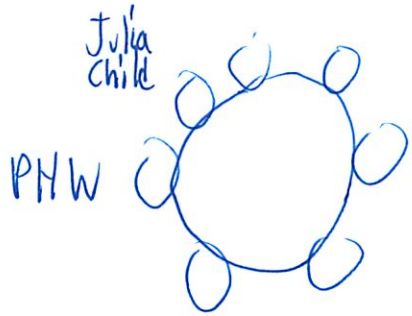
L something to learn
On 1 example
"One shot"

Sallent
Something that sticks out
1 or 2 small ideas that stick out
L can't have too many good ideas

Salient idea:
One shot via near miss

Is it legitimate to aspire to be famous

Winston at Sorrée - save Venice artwork



"Is it fun to be famous"
L you get used to it

But its not fun to be ignored

6.034 Lecture

11/9

Boosting

- ☐ Weak to Strong
- ☐ Adaptive Boosting Miracle
- ☐ Thank God Holes

* Many Weighted Smarter Than 1

Classic

- Classification trees
- Nearest neighbors

Bio

- Genetic algorithms
- Neural nets

Theory

- SVM
- Boosting

Specialization leads to work efficiencies

Military : career defined by worst thing
Academic : best

(2)

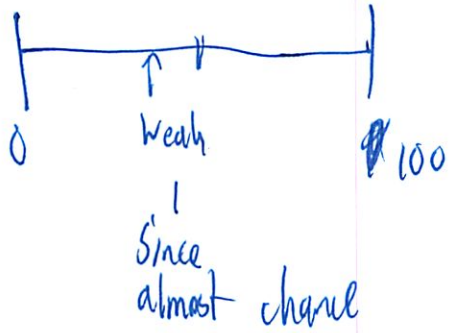
Shapiros' Boosting

- took a while to figure out
- diving +/-
- guarenteed to find a classifier

We've done a bunch of methods

Boosting is even more elegant

<u>Classifyer</u> divides samples into +/-



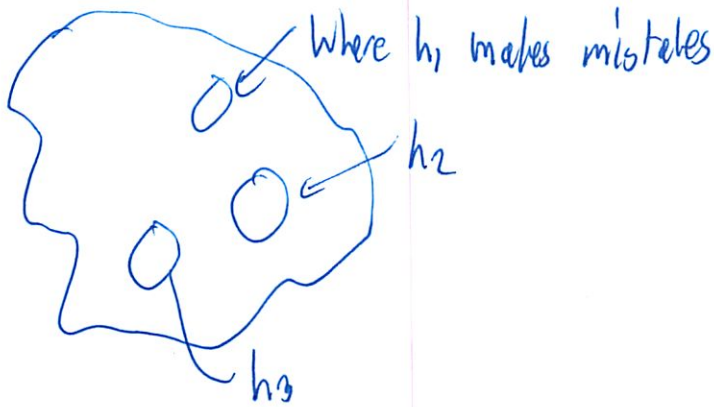
Can you combine a bunch of weak classifiers into a strong classifier?

(3)

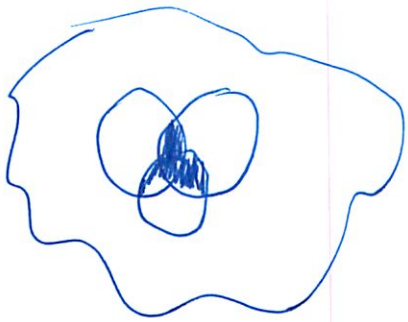
$$H(x) = \text{sign}\left(h_1(x) + h_2(x) + h_3(x) \right)$$

+1 if in class
−1 if not in class

So would this work?



No place where 2 classifiers are wrong
So right ones always outvote wrong one
But that won't always happen

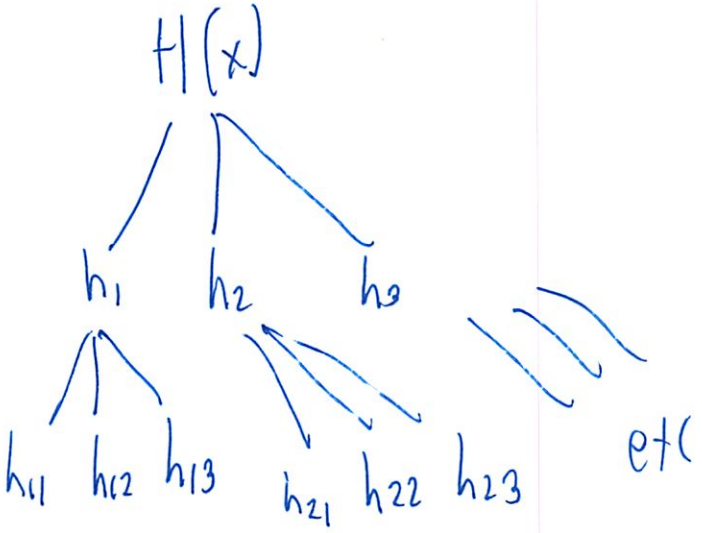


But it is usually true

(4)

Need good algorithm to pick h_1 h_2 h_3

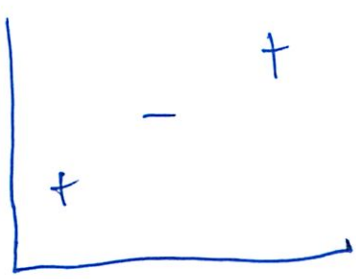
1. Pick h_1 w/ minimum ϵ ← so circle is small
2. Pick h_2 w/ emphasis on h_1 error ← take care of errors made earlier
3. Pick h_3 w/ emphasis on $h_1 \neq h_2$
   ↑ differing opinion
   need help to break ties



Can get better + better by adding classifyer

But isn't elegant - can't guarentee anything

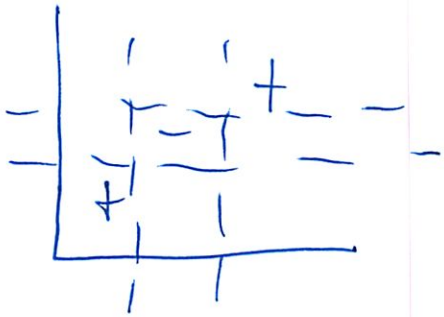
<u>Distribution</u>



(5)

Can try to use a classification tree

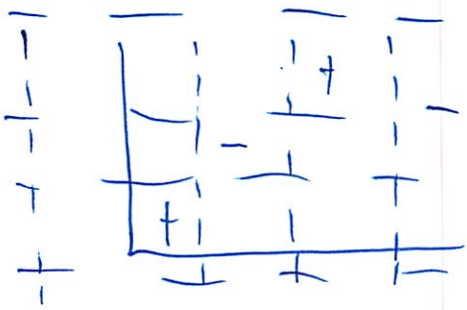
We will try to use a classification tree "stump"



Have 12 choices where we can test

~~[illegible]~~.

12 since also everything +
everything −



Try to find one to ↓ error

But what is an error?

Associte each point with a weight

initalize each weight as $w_i^0 = \frac{1}{N}$

So error is

$$\epsilon = \sum_{\text{nwrong}} w_i$$

that are

So at initalized - its just $\sum w$

But as progress - weights ~~seem~~ change to emphize samples where classifiers are making a mistake

Want classifier to ↓ weighted error

But how to adjust weights?

~~Step~~ Goal: Find $H(x) = \text{sign}(\alpha^0 h^0(x) + \alpha^1 h^1(x) + \ldots)$

enough to perfectly classify samples

1. Initalize $w_i = \frac{1}{N}$
2. Find h^t that minimizes $\sum w_i$ wrong
   classifier
3. Find the outcome α^t (mult'iplier) x time, which step on
4. Find w_i^{t+1} (weights on next round)
5. Go to 2.
   until we find something that works

7

Several points of magic

from previous exercise
+1 if classification +
-1 " " -

(A) $$W_i^{t+1} = \frac{w_i^t}{Z^t} e^{-\alpha^t h^t(x_i) y(x_i)}$$

? want weights to add to 1
normalizing constant to make new weights Σ to 1

(B) Exponential bound on ϵ if Z is minimized
- can go up
- but it will ultimatly go to 0
- proven

(C) $$\alpha^t = \frac{1}{2} \ln\left(\frac{1-\epsilon^t}{\epsilon^t}\right)$$

Exam will be use data boost algorthm
But we won't be able to solve it yet

What all this means

Can plug formula for α in
If sample is misclassified

$$W_i^{t+1} = \frac{W_i^t}{Z^t} e^{\frac{1}{2}\ln\left(\frac{1-\epsilon^t}{\epsilon^t}\right)}$$

8

$$= \frac{W_i^t}{Z^t}\sqrt{\frac{1-\epsilon^t}{\epsilon^t}}$$

Classified Correct

$$W_i^{t+1} = \frac{W_i^t}{Z^t}\sqrt{\frac{\epsilon^t}{1-\epsilon^t}}$$ ← flipped

↑ need to normalize

But still would flunk quiz

What must Z be?

Is a normalizer

Must add up to Z

So when divide by Z, it all sums to 1

$$Z^t = \sqrt{\frac{1-\epsilon^t}{\epsilon^t}} \underbrace{\sum W_i^t}_{\text{misclassified}} + \sqrt{\frac{\epsilon^t}{1-\epsilon^t}} \underbrace{\sum W_i^t}_{\text{classified}}$$

↑ definition of ϵ^t ↑ $1-\epsilon^t$

$$Z^t = \sqrt{\frac{1-\epsilon^t}{\epsilon^t}}\,\epsilon^t + \sqrt{\frac{\epsilon^t}{1-\epsilon^t}}\,(1-\epsilon^t)$$

$$= 2\sqrt{\epsilon}\sqrt{1-\epsilon}$$

(9)

Now plug in

Correct $W_i^{t+1} = \frac{W_i^t}{2} \frac{1}{1-\epsilon^t}$

two

incorrect/ missclassified $W_i^{t+1} = \frac{W_i^t}{2} \frac{1}{\epsilon^t}$

Thats what can use on exam

Is actually quite simple on exam

Need Thank God Holes

- like rockclimber - little thing to hold on
- need some for quiz

missclassified $\sum_i W_i^{t+1} = \frac{1}{2\cancel{\epsilon^t}} \underbrace{\sum_{\text{missclassified}} W_i^t}_{\cancel{\epsilon^t}}$ reweight

$= \frac{1}{2}$

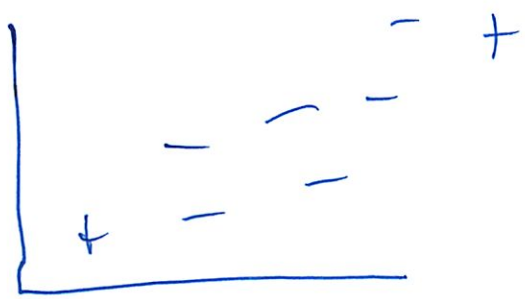
make sure $\Sigma = 1/2$

(10)

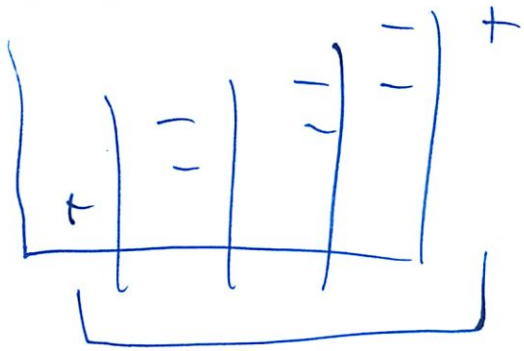
ok $\sum_i w_i^{t+1} = \frac{1}{2}$

Makes it easy to simulate by hand on easy cases

Thank God Hole 2



a bunch of stubs



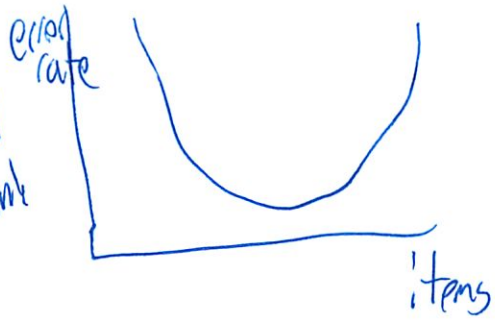
all useless
- so don't do
- saving effor

Overfitting

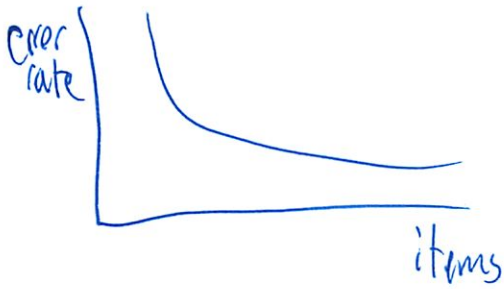
everything so far can overfit
Plage of machine learning
this seems <u>not</u> to overfit

(11)

You'd think



but actually



No one knows exactly why

If test w/ 7 dimensions

- Choose a sample to be noise
- will you overfit to it

What is volume of space around outlyer?

Very very small

Can you boost people?

People can do classification trees

Yes - did chart of team records vs pundits

- boosted better than pure

(12)

Put the 2 together boosted - 12% better than anything else

6.034 Recitation

1/10

2 handouts

Quiz Wed

k. Nearest Neighbors
ID trees / Classification trees
Neural Nets

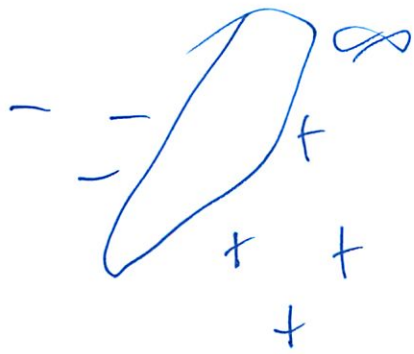
Not

SVMs
Boosting

SVM +/− classifier

tries to fit at many pts as possible
so can seperate regions as much
as possible w/o compromising split

Maximize $\frac{2}{\|w\|}$

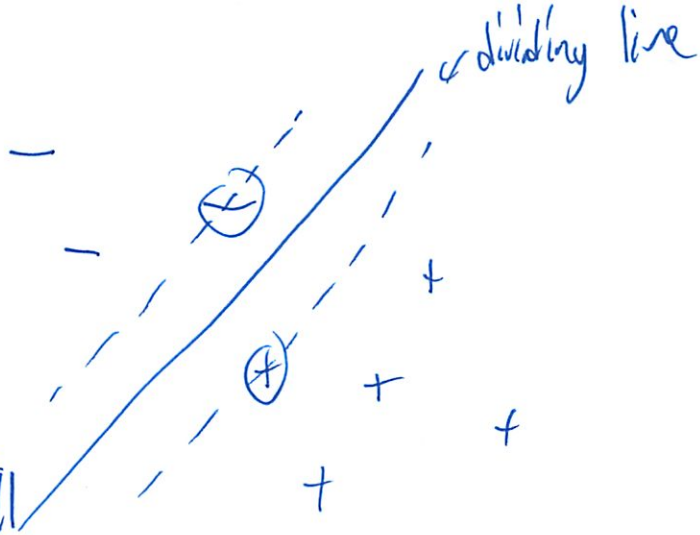


∞ ways to draw this line

②

Want to maximize distance b/w

- gutter + street



maximize $\frac{2}{||w||}$

use

$$||w|| = \sqrt{\sum w_i^2}$$

minimize this

Twant

$$s.t. \quad y_i(w_i x_i + b) \geq 1$$

dirty math

└ Lagrangian eq

min/max subject to this constraint

$$\frac{\partial L}{\partial b} =$$

hard on computer

└ quadratic programing problem

$$\frac{\partial L}{\partial w} =$$

③

$W_1 X_1 + W_2 X_2 + \cdots + W_n X_n$

Handout 2 Pg 1 – Quiz Problem

2 points that fix gutter point

Vectors (from origin) – data points in space

that is biggest margin can be given this data

here we do by visual inspection

a) What is distance? $2\sqrt{2}$ = margin

b) Wants equation of boundry lines?

$y = x - 4$ ⊕ line

$y = x - 2$ middle

$y = x$ ⊖ line

since $y \leq x - 2$ depends which side +, −

$x - y - 2 \geq 0$

(4)

Can you just read off line?

$$W_1 x - W_2 y - b \geq 0$$

$$1x - 1y - 2 > 0$$

No! This is not formula that maximizes margin

$$\sqrt{\sum w_i} = \sqrt{2} \qquad \frac{2}{\sqrt{2}}$$

"margin when $x - y - 2$"

This is not ~~smallest~~ largest margin which we said in last problem is $2\sqrt{2}$

Need to solve for W_1 W_2

$$Cx - Cy - 2C \geq 0$$

also sol, for any C

just another multiple of

$\downarrow$

$W_1 = C \quad W_2 = -C$

$$\frac{2}{\sqrt{\sum w_i^2}} = \frac{2}{\sqrt{C^2 + C^2}} = \frac{2}{\sqrt{2C^2}} = 2\sqrt{2}$$

(5)

Square both sides

$$\frac{4}{2c^2} = 8$$

Solve for c

$c > 0$ since otherwise switch $\oplus / \ominus$

$$c = \frac{1}{2}$$

So

$$W_1 = c = \frac{1}{2}$$

$$W_2 = -c = -\frac{1}{2}$$

$$b = -1$$

2. What is α of each data point

$$\boxed{\begin{array}{l} (1)\ \sum \alpha_i y_i = 0 \\ (2)\ \sum \alpha_i \vec{y_i x_i} = \vec{w} \end{array}}$$

So the non support ones are 0 - using (1)

The other ones must be found

- using (2)

got from $\frac{\partial L}{\partial w_i}$

(6)

$\alpha_1 = 0$

$\alpha_2 = 0$

$$\alpha_3 \underbrace{(-1)}_{\substack{\text{Y: output} \\ \text{of classifier} \\ (-3,-3) \to -1 \\ \ominus}} \underbrace{\begin{bmatrix} 3 \\ 3 \end{bmatrix}}_{\substack{\text{data point} \\ \#3}} + \alpha_4 \underbrace{(1)}_{\oplus} \underbrace{\begin{bmatrix} 5 \\ 1 \end{bmatrix}}_{\substack{\text{data pt} \\ \#4}} = \underbrace{\begin{bmatrix} 1/2 \\ -1/2 \end{bmatrix}}_{w}$$

2 eq 2 unknowns

$-3\alpha_3 + 5\alpha_4 = \frac{1}{2}$ since $2\alpha_5$

$-3\alpha_3 + 1\alpha_4 = -\frac{1}{2}$

Subtract 2nd eq from 1st

etc

⋮ Solve

$4\alpha_4 = 1$

$\alpha_3 = \frac{1}{4}$

$\alpha_4 = \frac{1}{4}$

Can try

$\sum \alpha_3 y_3 + \alpha_4 y_4 = 0$

(7) continued

(8)

Also penalty rule to have less points on there
└ Winston does not really cover

~~Euclidean space~~

Kernal

- math function used to compute
- Eucleiden $u \cdot v$
- New space $\phi(u) \cdot \phi(v)$
  ↑ kernal fn: dot product in new space
- for when single linear cut ~~space~~ no longer works
  - like w/ perceptron
- 1 straight cut

(8)

So do these kernals work

a) $u \cdot v$ straight line – so no

← can't have 1 cut divide

b) $(u \cdot v + 1)^2$ quadratic – so curved – parabolic

Yes:

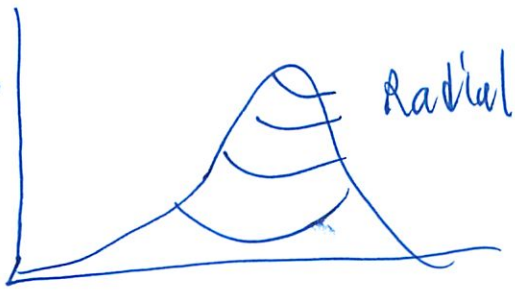
Counter lines can't cross

as few pts as possible on gutter

do need at least 2pts on gutter line

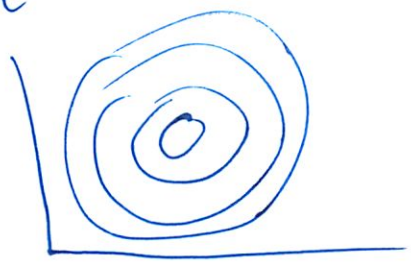
c) gutters won't be even/nice this time

$e^{-\frac{|u-v|^2}{2}}$



(9)

So looks like



So will that work

(Yes)

Next pg

(D) over fits

- twists too close to data

want to classify - not a line through each pt

(E) does good job

- divides points well w/o overfitting

~~Sometimes~~ Always easier to train quadratic for SVM than ~~xxxxx~~ neural nets

Massachvsetts Institvte of Technology
Department of Electrical Engineering and Computer Science
6.034 Recitation, Thursday, November 10

SVMs & Boosting — Prof. Bob Berwick, 32D-728

Part I. SVMs: the Basic Ideas

Another method for classifying unseen data that avoids over-fitting. We train a system on a set of known (x, y) points, with known output + or –, then give it an unknown sample (as with neural nets, or k-Nearest Neighbors, or ID-trees…)

We are trying to find the decision boundary that maximizes the "margin" m or the "width of the street" separating the positive from the negative training data points (the "positive gutter" plus the "negative gutter"). Referring to Winston's notes, the equation for the width of the margin (aka the "street" or "road") is:

$$m = \frac{2}{\|\vec{w}\|} \text{ where } \|\vec{w}\| = \sqrt{\sum_i w_i^2} \text{ (i.e., the length of } \vec{w}\text{)}$$

Thus to **maximize** the street width, we must **minimize** $\|\vec{w}\|$, subject to the constraint that the decision boundary classifies the training points correctly as either positive or negative, i.e.:

$$y_i(\vec{w} \bullet \vec{x}_i + b) \geq 1$$

The resulting Lagrange multiplier equation L that will solve this optimization problem is the following:

$$L = \frac{1}{2}\|\vec{w}\|^2 - \sum_i \alpha_i (y_i(\vec{w} \bullet \vec{x}_i + b) - 1)$$

Solving the optimization problem above, we take partials with respect to b and then w (as shown below) to solve for w, b, and the values of alpha, α_i, parameters that determine a **unique** maximal margin (road/street) boundary line solution. On this maximum margin "street" the positive and negative data points that are on the road, with **nonzero alpha values**, are called **support vectors.** The decision boundary lies in the middle of the road. The definition of the road depends **only on the support vectors** so changing (adding or deleting) non-support vector points will not change the solution. (Note that in higher dimensions, we want the widest region separated by two planes, hyperplanes, etc.) That is, support vectors have **weights** α associated with them = amount of influence on the surrounding region = Langrangian multipliers found from the optimization problem. If a training data point receives a weight α_i of zero, this means that the data point **does not affect** the location of the decision boundary or the 'street'.

- The closer the support vectors of the opposite classes, the narrower the street, and the **greater** the weights (the alphas) for the corresponding support vectors. The support vectors have to supply more 'pressure' to push the margin tighter.
- The farther apart the support vectors of the opposite classes, the wider the street, and the **smaller** the alphas for the corresponding support vectors. The wider street needs less pressure on the supports to hold it in place.

The kernel trick.

Note that the equation L will in fact only require us to be able to compute the "dot product" of w and x. This amounts to being able to compute the "distance" between w and x. In ordinary Euclidean space, with an ordinary metric, this just *is* the dot product as written. However, we can always map the system to a different dimension (either higher or lower, but typically higher because this "separates" the data better), via a transform that is called ϕ. We will give examples below. The beautiful part is, we **do not need to compute this mapping explicitly**. We need only know what the corresponding **dot product** (= distance measure) is in the new space, which is called the **kernel function, *K*.** (So in a usual, normal, untransformed

Euclidean space, we have a linear Kernel function.) That is, for any two vectors u, v, we have that $\boldsymbol{K(u, v) = \phi(u) \cdot \phi(v)}$.

Solving for the Lagrange multiplier alphas in general requires numerical optimization methods beyond the scope of this course. In practice, one uses quadratic programming methods. On quizzes, we generally just ask you to solve for the values using algebra and/or geometry.

1. Useful equations for solving SVM questions, followed by an explicit example.

A. Equations derived from optimizing the Lagrangian:

1. Partial of the Lagrangian *L* wrt to *b*, i.e., $\frac{\partial L}{\partial b} = 0$

$\sum_i \alpha_i y_i = 0$ Note that $y_i \in \{-1,+1\}$ and $\alpha_i = 0$ for non-suport vectors

Important: The SUM of all the alphas (support vector weights) with their signs should add to 0 !!!!

2. Partial of the Lagrangian *L* wrt *w*, i.e., $\frac{\partial L}{\partial w} = 0$

There are 2 cases.

Case 1. When using a linear kernel: positive terms of the sum involve support vectors. Support vectors are training data points with alphas > 0

$\sum_i \alpha_i y_i \vec{x}_i = \vec{w}$

Case 2. The general case, where we have kernels that define the 'distance' between pts differently.

$\sum_i \alpha_i y_i \phi(\vec{x}_i) = \vec{w}$

B. Equations derived from the +/– boundaries and the constraints:

3. The decision boundary (used to classify a new data point as either + or –):

Case 1. When using a linear kernel, $K(\vec{x}_i, \vec{x}) = \vec{x}_i \bullet \vec{x}$

$h(\vec{x}) = \sum_i [(\alpha_i y_i \cdot \vec{x}_i) \cdot \vec{x}] + b \geq 0$ or $h(\vec{x}) = \vec{w} \cdot \vec{x} + b \geq 0$

Case 2. When using a general kernel function. We compute the kernel function $K(\vec{x}_i, \vec{x}) = \vec{x}_i \bullet \vec{x}$ **against each of the support vectors** x_i**. Support vectors are training data points with** $\alpha_i > 0$**.**

$h(\vec{x}) = \sum_i \alpha_i y_i K(\vec{x}_i, \vec{x}) + b \geq 0$

4. The positive gutter:

Case 1. When using a linear kernel:

$h(\vec{x}) = \sum_i [(\alpha_i y_i \cdot \vec{x}_i) \cdot \vec{x}] + b = 1$ or $h(\vec{x}) = \vec{w} \cdot \vec{x} + b = 1$

Case 2. When using a general kernel:

$h(\vec{x}) = \sum_i \alpha_i y_i K(\vec{x}_i, \vec{x}) + b = 1$

5. The negative gutter:

$h(\vec{x}) = \sum_i [(\alpha_i y_i \cdot \vec{x}_i) \cdot \vec{x}] + b = -1$ or $h(\vec{x}) = \vec{w} \cdot \vec{x} + b = -1$; $h(\vec{x}) = \sum_i \alpha_i y_i K(\vec{x}_i, \vec{x}) + b = -1$

6. The width of the margin (or "street" or "road"):

$m = \frac{2}{\|\vec{w}\|}$ where $\|\vec{w}\| = \sqrt{\sum_i w_i^2}$ (i.e., the length of $\vec{w}$)

if just the two support vector case:

$m = \frac{2}{\|\vec{w}\|} \cdot (\vec{x}_+ - \vec{x}_-)$

(The last equation is useful in the 1-D or 2-D case where the width of the margin can be visually determined.)

As noted above, one common SVM kernel is just the **linear** one, where the kernel $K(u, v)$ is just the ordinary dot product of u and v. This is often used in document classification, with binary-valued feature vectors, and we output a +1 if the document is one class, and –1 if in another. Before looking at other kernels, let's first try an SVM problem where we have to arrive at the SVM solution visually (from the 2010 exam). Let's take a look, on your other handout,.
Let's first determine by eye where the boundary line goes. (Surprise – this has already been done for you.) What about the width of the margin m, i.e., the width of the 'street'? What is that width?

OK, the next part of the problem will ask us to find the actual equation of the SVM decision boundary. We can do this by inspection in such cases, in two steps.

Step 1. Find a (preliminary) decision boundary line, by visual inspection.
1. The decision boundary is on the line: $y= x–2$.
2. We have a positive support vector at the point (5, 1), with the associated line equation, $y= x–4$
3. We have a negative support vector at the point (3, 3) with the associated line equation, $y=x$

Step 2. We manipulate the boundary line equation to get it into the canonical form, $h(x)=w_1x + w_2y+b \geq 0$
1. $y \leq x–2$ (Why? Because positive points + are **below** the line.)
2. $x–y –2 \geq 0$

From this it *seems* like we can "read off" the 'solution' directly of: $w_1= +1$; $w_2= –1$; $b= –2$.
But is this correct? If we plug in these values for w into our equation for the road/street width m we get:

$$m = \frac{2}{\|\vec{w}\|} \text{ where } \|\vec{w}\| = \sqrt{\sum_i w_i^2} = \frac{2}{\sqrt{\sum_i w_i^2}} = \frac{2}{\sqrt{1+1}} = \frac{2}{\sqrt{2}}$$

Oops! This is **not** what the result for what the width is supposed to be – it is too small (by half). The street width, m, from visual inspection is $2\sqrt{2}$. The margin must be wider!

Step 3. So, one must realize that **any multiple c of the boundary equation is still the same decision boundary**. So, the general equation form below gives us the decision boundary, where we must now solve for c by using the other constraints of the problem (namely, the known street width):

$$\boxed{cx - cy - 2c \geq 0}$$

If we plug these values into the computation for the street/road width m, we get the following equation, because we can set it equal to the **known** street width $2\sqrt{2}$.:

$$\frac{2}{\sqrt{(c)^2 + (-c)^2}} = \frac{2}{\sqrt{2c^2}} = 2\sqrt{2}$$

$$\frac{4}{2c^2} = 8$$

$$\frac{2}{c^2} = 8$$

$$c^2 = \frac{1}{4}$$

$$c = \frac{1}{2}$$

Now we can read off the proper values for the w vector from the general equation:

$$cx - cy - 2c \geq 0$$

$$\frac{1}{2}x - \frac{1}{2}y - 1 \geq 0$$

$$\text{so: } \vec{w} = \begin{bmatrix} +1/2 \\ -1/2 \end{bmatrix} \text{ and } b = -1$$

The next page of the problem asks us to compute the alpha (α) values for **each** of the training points. Recall that we can do this by using Equation (2), Case 1 above (since we are using a linear kernel):

$$\sum_i \alpha_i y_i \vec{x}_i = \vec{w}$$

But before plunging ahead, **what is it we already know about the alpha values associated with NON-SUPPORT VECTORS? Answer: their alphas are 0. That is, for non-support vectors (all the points EXCEPT for the two we used) we already know that their alpha values are 0.**

As for the precisely two other data points, to find their associated α_3 and α_4 values we merely have to plug in values for (3, 3), which has an associated output of –1 (this becomes the 'y' value in the equation above – don't get confused by the problem's notation), and for (5, 1), which has an associated output value of +1:

$$\sum_i \alpha_i y_i \vec{x}_i = \vec{w} = \text{ so } \alpha_3(-1)\begin{bmatrix} 3 \\ 3 \end{bmatrix} + \alpha_4(+1)\begin{bmatrix} 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/2 \\ -1/2 \end{bmatrix}$$

$$-3\alpha_3 + 5\alpha_4 = 1/2$$

$$-3\alpha_3 + \alpha_4 = -1/2 \ \therefore$$

$$4\alpha_4 = 1,\ \alpha_4 = 1/4;\quad -3\alpha_3 = -3/4;\quad \alpha_3 = 1/4$$

Now, to cement our understanding of what the support vectors *mean*, the question goes on to ask what the alpha values would be for some new points. First, what about a new data point (0, 6) classified as negative? The answer is that the alpha would be 0, because it **cannot** be a support vector – it lies behind the negative gutter line and so has no impact on the decision boundary. So what about a new data point (0, 0) also classified as negative?

More generally, we can consider what happens when we move one of the support vectors, as the last question does. What happens to its corresponding alpha value? (If it is still a support vector!). The *general* rule is that alpha will change *inversely* with the road width m. Since moving the support vector from (3, 3) to (4, 2) **decreases *m*,** it will **increase the alpha associated with the support vector. You can think of this intuitively as the vector having to 'push' harder on the gutter line to force it in.**

3. Other spaces, other kernels

OK, now the other big win with SVMs has to do with the ease with which you can transform from one space to another, where the data may be more easily separable. The remaining questions all ask you about that, for **different** kinds of kernels (= different ways to compute inner products, or 'distance', aka, 'similarity' of two points). That is, we need to define $K(u,v) = \varphi(u)\cdot\varphi(v)$, the dot product in the transformed space. (You should try these out in Winston's demo program to see how the decision boundaries change.)

The basic kernels we consider are these:

1. Single linear kernel. These are just straight lines in the plane (or in higher dimensions). You should remember what perceptrons can and cannot 'separate' via cuts, and this tells you what linear kernels can do. (But see below under linear combination of kernels!!!)

$K(\vec{u},\vec{v}) = (\vec{u}\cdot\vec{v}) + b$, e.g., $K(\vec{u},\vec{v}) = (\vec{u}\cdot\vec{v})$ (ordinary dot product)

2. Polynomial kernel.

$K(\vec{u},\vec{v}) = (\vec{u}\cdot\vec{v} + b)^n$, $n > 1$

eg., Quadratic kernel: $K(\vec{u},\vec{v}) = (\vec{u}\cdot\vec{v} + b)^2$

In 2-D the resulting decision boundary can look parabolic, linear, or hyperbolic depending on which terms in the expansion dominate.

3. Radial basis function (RBF) or Gaussian kernel.

$$K(\vec{u},\vec{v}) = -\exp\left(-\frac{\|\vec{u}-\vec{v}\|^2}{2\sigma^2}\right)$$

In 2-D, the decision boundaries for RBFs resemble contour circles around clusters of positive and negative points Support vectors are generally positive or negative points that are closest to the opposing cluster. The contour space that results is drawn from the sum of support vector Gaussians. Try the demo to see.

When the variance or spread of the Gaussian curve σ^2 ('sigma-squared') is large, you get 'wider' or 'flatter' Gaussians. When it is small, you get sharper Gaussians. Hence, when using a small sigma-squared, the contour density will appear closer, or tighter, around the support vector points. In 2-D, as a point gets closer to a support vector, it will approach *exp*(0)=1, and as it gets farther away, it approaches *exp*(–infinity)= 0.

Note that you can **combine** several radial basis function kernels to get a perfect fit around **any** set of data points, but this will usually amount to a typical case of over-fitting – there are 2 free parameters for every RBF kernel function.

4. Sigmoidal (tanh) kernel. This allows for a combination of linear decision boundaries, like neural nets.

$K(\vec{u},\vec{v}) = \tanh(k\vec{u}\cdot\vec{v} + b)$

$$K(\vec{u},\vec{v}) = \frac{e^{k\vec{u}\cdot\vec{v}+b} + 1}{e^{k\vec{u}\cdot\vec{v}+b} - 1}$$

The properties of this kernel function: it is similar to the sigmoid function; it ranges from –1 to +1; it approaches +1 when $x >> 0$; and it approaches –1 when $x << 0$. The resulting decision boundaries are logical combinations of linear boundaries, not that different from second-layer neurons in neural nets.

5. Linear combinations of kernels (scaling or general linear combination).

Kernel functions are closed under addition and scaling by a positive factor.

Let's practice on one more sample problem. There are also 2 appendices at the end of this handout for folks that are hard-core, one working out the math, the second, solving the XOR problem using SVMs, by using a polynomial kernel function.

Part II. Boosting and the Adaboost algorithm

0. The idea behind **boosting** is to find a weighted combination of s "weak" classifiers (classifiers that underfit the data and still make mistakes, though as we will see they make mistakes on less than ½ the data), $h_1, h_2 ..., h_s$, into a **single strong** classifer, $H(x)$. This will be in the form:

$$H(\vec{x}) = sign(\alpha_1 h_1(\vec{x}) + \alpha_2 h_2(\vec{x}) + \cdots + \alpha_s h_s(\vec{x})$$

$$H(\vec{x}) = sign\left(\sum_{i=1}^{s} a_i h_i(\vec{x})\right)$$

$$\text{where: } H(\vec{x}) \in \{-1,+1\}, h_i(\vec{x}) \in \{-1,+1\}$$

Recall that the *sign* function simply returns +1 if weighted sum is positive, and –1 if the weighted sum is negative (i.e., it classifies the data point as + or –).

Each training data point is **weighted.** These weights are denoted w_i for i=1, ..., n. **Weights** are *like* probabilities, from the interval (0, 1], with their **sum equal to 1. BUT** weights are **never 0**. This implies that **all data points have some vote** on what the classification shuld be, at all times. (You might contrast that with SVMs.)

The general idea will be to pick a single 'best' classifier h (one that has the lowest error rate when acting all alone), as an initial 'stump' to use. Then, we will **boost** the weights of the data points that this classifier **mis-classifies (makes mistakes on),** so as to focus on the next classifier h that does best on the re-weighted data points. This will have the effect of trying to fix up the errors that the first classifier made. Then, using this next classifier, we repeat to see if we can now do better than in the first round, and so on. In computational practice, we use the same sort of entropy-lowering function we used with ID/classifier trees: the one to pick is the one that lowers entropy the most. But usually we will give you a set of classifiers that is easier to 'see', or will specify the order.

In Boosting we always pick these initial 'stump' classifiers so that the error rate is strictly < ½. Note that if a stump gives an error rate greater than ½, this can always be 'flipped' by reversing the + and – classification outputs. (If the stump said –, we make it +, and vice-versa.) Classifiers with error exactly equal to ½ are useless because they are no better than flipping a fair coin.

1. Here are the definitions we will use.

Errors:

The error rate of a classifier s, E^s, is simply the sum of all the *weights* of the training points classifier h_s gets **wrong**.

$(1-E^s)$ is 1 minus this sum, the sum of all the *weights* of the training points classifier h_s gets **correct.**

By assumption, we have that:

$E^s < ½$ and $(1-E^s) > ½$, so $E^s < (1-E^s)$, which implies that $(1-E^s)/E^s > 1$

Weights:

α_s is **defined** to be ½ $\ln[(1-E^s)/E^s)]$, so from the definition of weights, the quantity inside the ln term is > 1, so all alphas must be positive numbers.

Let's write out the Adaboost algorithm and then run through a few iterations of an example problem.

2. Adaboost algorithm

Input: training data, $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$

1. **Initialize data point weights.**

   Set $w_i^1 = \frac{1}{n} \ \forall i \in (1, \ldots, n)$

2. **Iterate over all 'stumps':** for s=1, ..., T

   a. **Train base** learner using distribution w^s on training data.

   Get a base (stump) classifier $h_s(x)$ that achieves the lowest error rate E^s .

   (In examples, these are picked from pre-defined stumps.)

   b. **Compute the stump weight:** $\alpha_s = \frac{1}{2} \ln \frac{(1 - E^s)}{E^s}$

   c. **Update weights** (3 ways to do this; we pick Winston's method)

   For points that the classifier **gets correct,** $w_i^{s+1} = \left[\frac{1}{2} \cdot \frac{1}{1 - E^s} \right] \cdot w_i^s$

   (Note from above that $1 - E^s > ½$, so the fraction $1/(1 - E^s)$ must be < 2, so the total factor scaling the old weight must be < 1, i.e., the **weight of correctly classified points must go DOWN in the next round)**

   For points that the classifier **gets incorrect,** $w_i^{s+1} = \left[\frac{1}{2} \cdot \frac{1}{E^s} \right] \cdot w_i^s$

   (Note from above that $E^s < ½$, so the fraction $1/E^s$) must be > 2, so the total factor scaling the old weight must be > 1, i.e., the **weight of incorrectly classified points must go UP in the next round)**

3. **Termination:**

   If $s > T$ or if $H(x)$ has error 0 on training data or < some error threshold, exit;

   If there are no more stumps h where the weighted error is < ½, exit (i.e., all stumps now have error exactly equal to ½)

4. **Output final classifier:**

$H(\vec{x}) = sign\left(\sum_{i=1}^{s} a_i h_i(\vec{x}) \right)$ **[this is just the weighted sum of the original stump classifiers]**

Note that test stump classifiers that are **never** used are ones that make more errors than some pre-existing test stump. In other words, if the set of mistakes stump X makes is a **superset** of errors stump Y makes, then Error(X) > Error(Y) is **always** true, no matter weight distributions we use. Therefore, we will **always** pick Y over X because it makes fewer errors. So X will **never** be used!

3. Let's try a boosting problem from an exam (on the other handout).

4. Food for thought questions.

1. How does the weight α^s given to classifier h_s relate to the performance of h_s as a function of the error E^s?
2. How does the error of the classifier E^s affect the new weights on the samples? (How does it raise or lower them?)
3. How does AdaBoost end up treating outliers?
4. Why is not the case that new classifiers "clash" with the old classifiers on the training data?
5. Draw a picture of the training error, theoretical bound on the true error, and the typical test error curve.
6. Do we expect the error of new weak classifiers to increase or decrease with the number of rounds of estimation and re-weighting? Why or why not?

Answers to these questions:

1. How does the weight α^s given to classifier h_s relate to the performance of h_s as a function of the error E^s?

Answer: The lower the error the better the classifier h is on the (weighted) training data, and the larger the weight α^t we give to the classifier output when classifying new examples.

2. How does the error of the classifier E^s affect the new weights on the samples? (How does it raise or lower them?)

Answer: The lower the error, the better the classifier h classifies the (weighted) training examples, hence the larger the increase on the weight of the samples that it classifies incorrectly and similarly the larger the decrease on those that it classifies correctly. More generally, the smaller the error, the more significant the change in the weights on the samples. Note that this dependence can be seen indirectly in the AdaBoost algorithm from the weight of the corresponding classifier α_t. The lower the error E^t, the larger α_t, the better h_t is on the (weighted) training data.

3. How does AdaBoost end up treating outliers?

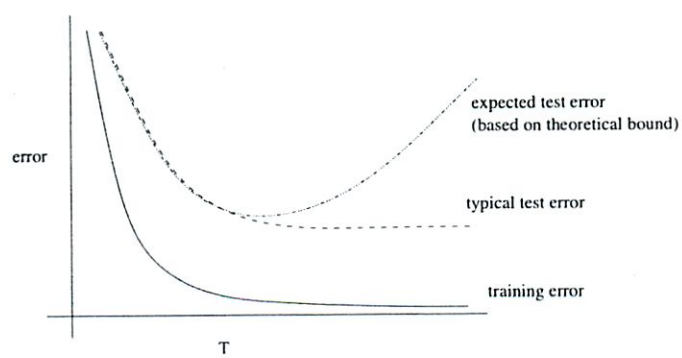
Answer: AdaBoost can help us identify outliers since those examples are the hardest to classify and therefore their weight is likely to keep increasing as we add more weak classifiers. At the same time, the theoretical bound on the training error implies that as we increase the number of base/weak classifiers, the final classifier produced by AdaBoost will classify all the training examples. This means that the outliers will eventually be "correctly" classified from the standpoint of the training data. Yet, as expected, this might lead to overfitting.

4. Why is not the case that new classifiers "clash" with the old classifiers on the training data?

Answer: The intuition is that, by varying the weight on the examples, the new weak classifiers are trained to perform well on different sets of examples than those for which the older weak classifiers were trained on. A similar intuition is that at the time of classifying new examples, those classifiers that are not trained to perform well in such examples will cancel each other out and only those that are well trained for such examples will prevail, so to speak, thus leading to a weighted majority for the correct label.

5. Draw a picture of the training error, theoretical bound on the true error, and the typical test error curve.

Answer:



6. Do we expect the error of new weak classifiers to increase or decrease with the number of rounds of estimation and re-weighting? Why?

Answer: We expect the error of the weak classifiers to increase in general since they have to perform well in those examples for which the weak classifiers found earlier did not perform well. In general, those examples will have a lot of weight yet they will also be the hardest to classify correctly.

Appendix I. Gory details for SVMs [to be read at your non-leisure]

Equations

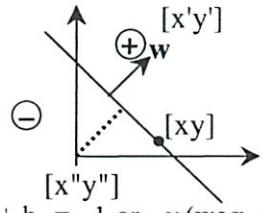
a. **Training**: maximize width of street by maximizing the equation of support vector weights (α_i) and the dot products of the support vectors: $L_{DUAL}=\Sigma\alpha_i - 1/2\ \Sigma_i\Sigma_j\alpha_i\alpha_j y_i y_j(\mathbf{x_i}\bullet\mathbf{x_j})$

b. **Classification:** assigns unknown point **u** to either one of 2 classes: + or – class:

$\mathbf{w}\bullet\mathbf{u} + b > 0$ then +

$\mathbf{w}\bullet\mathbf{u} + b < 0$ then –

o.w., ??? so assume –



b= –d

$\mathbf{w}\bullet[xy] = d$

$\mathbf{w}\bullet[x'y'] > d$

$\mathbf{w}\bullet[x''y''] < d$

c. Support vector constraints: $\mathbf{w}\bullet\mathbf{x}_+ + b = 1$; $\mathbf{w}\bullet\mathbf{x}_- + b = -1$ or $y_i(\mathbf{w}\bullet\mathbf{x_i} + b) = 1$ where y_i= +1 or –1

d. $\mathbf{w} = \Sigma\alpha_i y_i \mathbf{x_i}$ is a function of weights on the support vectors, which are 0 for non-support vectors (Thus non-support vectors contribute nothing to where the decision boundary goes.)

e. $\Sigma\alpha_i y_i = 0$ (The constraints that must be obeyed to correctly classify the training points.)

Now let's do the math to see where this all comes from

0. The two gutter lines are: $\overline{w}\cdot\mathbf{x}_1 + b = +1;\ \overline{w}\cdot\mathbf{x}_2 + b = -1$

1. From Winston's picture, the street width is found as:

$$\overline{w}\cdot(\mathbf{x}_1 - \mathbf{x}_2) = 2$$

$$\frac{\overline{w}}{\|\overline{w}\|}\cdot(\mathbf{x}_1 - \mathbf{x}_2) = \frac{2}{\|\overline{w}\|}$$

2. To maximize this, we minimize $\|\overline{w}\|$, which is the same as minimizing ½ $\|\overline{w}\|^2$ (Why this? We will be taking the derivative in just a bit, and the derivative of this is just $\overline{w}$.)

3. So we want to:

A. *minimize* ½ $\|\overline{w}\|^2$ s.t. $y_i(\overline{w}\cdot\overline{x}_i + b) - 1 = 0$

4. To do the minimization, we use a Lagrangian formulation, to add a penalty function in order to ensure that the constraints on correctly classifying the training points are obeyed:

B. *minimize L* where $L = ½\|\overline{w}\|^2 - \sum_{i=1}^{r}\alpha_i[y_i(\overline{w}\cdot\overline{x}_i + b) - 1]$

(See last page for cheat sheet on how Lagrange multipliers work.)

5. Take partial derivatives with respect to the 2 variables, $\overline{w}$ and b:

C.

$$\frac{\partial L}{\partial \overline{w}} = 0 \quad \frac{\partial L}{\partial b} = 0$$

So this means we have:

C.1 $\dfrac{\partial ½\|\overline{w}\|^2}{\partial \overline{w}} = \overline{w}$

C.2 $\dfrac{\partial L}{\partial \overline{w}} = \overline{w} - \sum_{i=1}^{r}\alpha_i y_i \overline{x}_i \Rightarrow \overline{w} = \sum_{i=1}^{r}\alpha_i y_i \overline{x}_i$ **we substitute for $\overline{w}$ in (B) above to get C.4**

C.3 $\dfrac{\partial L}{\partial b} = \sum_{i=1}^{r} \alpha_i y_i = 0$ **= basic constraint on support vectors** α_i

C.4 Substituting for $\|\bar{w}\| = \sum_{i=1}^{r} \alpha_i y_i \bar{x}_i$:

$$L = \tfrac{1}{2}\|\bar{w}\|^2 - \sum_{i=1}^{r} \alpha_i [y_i(\bar{w}_i \cdot \bar{x}_i + b) - 1]$$

$$= \tfrac{1}{2}(\sum_{i=1}^{r} \alpha_i y_i \bar{x}_i) \cdot (\sum_{j=1}^{r} \alpha_j y_j \bar{x}_j) - (\sum_{i=1}^{r} \alpha_i y_i \bar{x}_i)(\sum_{j=1}^{r} \alpha_j y_j \bar{x}_j) - b\cancelto{0 \text{ from constraint } \mathbf{C.3}}{\sum_{i=1}^{r} \alpha_i y_i} + \sum_{i=1}^{r} \alpha_i$$

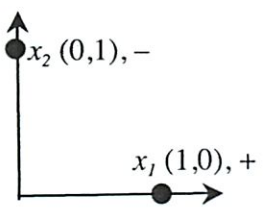
$$= \sum_{i=1}^{r} \alpha_i + \tfrac{1}{2}\sum_{i=1}^{r}\sum_{j=1}^{r} \alpha_i \alpha_j y_i y_j \bar{x}_i \cdot \bar{x}_j - \sum_{i=1}^{r}\sum_{j=1}^{r} \alpha_i \alpha_j y_i y_j \bar{x}_i \cdot \bar{x}_j$$

$$= \sum_{i=1}^{r} \alpha_i - \tfrac{1}{2}\sum_{i=1}^{r}\sum_{j=1}^{r} \alpha_i \alpha_j y_i y_j \bar{x}_i \cdot \bar{x}_j$$

C.5 *maximize* $L_{Dual} = \sum_{i=1}^{r} \alpha_i - \tfrac{1}{2}\sum_{i=1}^{r}\sum_{j=1}^{r} \alpha_i \alpha_j y_i y_j \bar{x}_i \cdot \bar{x}_j$

(IMPORTANT: Note <u>dot product</u> of training data, the last 2 terms! This is the "similarity" measure.)

<u>Worked Math example, linear SVM, 2 support vector "points" (one positive example, one negative example)</u>



$$y_i(\bar{w} \cdot \bar{x}_i + b) = 1; \quad y_i(\bar{w} \cdot \bar{x}_i + b) - 1 = 0$$

$$\bar{w} = \sum_{i=1}^{2} \alpha_i y_i \bar{x}_i$$

$$\sum_{i=1}^{2} \alpha_i y_i = 0$$

Find: $\alpha_1, \alpha_2, \bar{w}, b$:

$$L = \tfrac{1}{2}\|\bar{w}\|^2 - \sum_{i=1}^{r} \alpha_i [y_i(\bar{w} \cdot \bar{x}_i + b) - 1]$$

$$= \tfrac{1}{2}\|\bar{w}\|^2 - \alpha_1(+1 \cdot (\bar{w} \cdot \bar{x}_1 + b) - 1) - \alpha_2(-1 \cdot (\bar{w} \cdot \bar{x}_2 + b) - 1)$$

$$\frac{\partial L}{\partial \bar{w}} = \bar{w} = \alpha_1 \bar{x}_1 - \alpha_2 \bar{x}_2$$

$$\frac{\partial L}{\partial b} = -\alpha_1 + \alpha_2 = 0; \quad \therefore \ \alpha_1 = \alpha_2$$

Substituting for $\bar{w} = \alpha_1 \bar{x}_1 - \alpha_2 \bar{x}_2$ from above in the equation for L, we have:

$$L = \tfrac{1}{2}[\alpha_1 \bar{x}_1 - \alpha_2 \bar{x}_2]^2 - \alpha_1[+1 \cdot ((\alpha_1 \bar{x}_1 - \alpha_2 \bar{x}_2) \cdot \bar{x}_1 + b) - 1] - \alpha_2[-1 \cdot ((\alpha_1 \bar{x}_1 - \alpha_2 \bar{x}_2) \cdot \bar{x}_2 + b) - 1]$$

$$= \tfrac{1}{2}[\alpha_1^2 \bar{x}_1 \cdot \bar{x}_1 - \alpha_1 \alpha_2 \bar{x}_1 \cdot \bar{x}_2 + \alpha_2^2 \bar{x}_2 \cdot \bar{x}_2] - \alpha_1[\alpha_1 \bar{x}_1 \cdot \bar{x}_1 - \alpha_2 \bar{x}_2 \cdot \bar{x}_1 + b - 1] - \alpha_2[-\alpha_1 \bar{x}_1 \cdot \bar{x}_2 + \alpha_2 \bar{x}_2 \cdot \bar{x}_2 - b - 1$$

Since

$\overline{x}_1 \cdot \overline{x}_1 = 1;\ \overline{x}_2 \cdot \overline{x}_2 = 1;\ \overline{x}_1 \cdot \overline{x}_2 = 0$ (here is where we use the dot product!), we can simplify this to:

$$L = \tfrac{1}{2}[\alpha_1^2 + \alpha_2^2] - \alpha_1^2 - \alpha_1 b + \alpha_1 - \alpha_2^2 + \alpha_2 b + \alpha_2$$

Since $\alpha_1 = \alpha_2$

$$L = \tfrac{1}{2}[2\alpha_1^2] - 2\alpha_1^2 + 2\alpha_1 = -\alpha_1^2 + 2\alpha_1$$

Since we want to maximize (minimize) L,

$\frac{\partial L}{\alpha_1} = 0 = -2\alpha_1 + 2 \Rightarrow \alpha_1 = 1;$ since $\alpha_1 = \alpha_2,\ \ \alpha_2 = 1$ (Note that $\sum_{i=1}^{2} \alpha_i y_i = 1(+1) + 1(-1) = 0$ as required)

Finally, to find b:

$$y_i(\overline{w} \cdot \overline{x}_i + b) = 1$$

$$\overline{w} = \alpha_1 \overline{x}_1 - \alpha_2 \overline{x}_2 = \overline{x}_1 - \overline{x}_2$$

Substituting for $\overline{w}$,

$$1 \cdot (\overline{x}_1 - \overline{x}_2) \cdot \overline{x}_1 + b = 1$$

$\overline{x}_1 \cdot \overline{x}_1 - \overline{x}_2 \cdot \overline{x}_1 + b = 1$, but $\overline{x}_1 \cdot \overline{x}_1 = 1$ and $\overline{x}_2 \cdot \overline{x}_1 = 0$, so

$$b = 0$$

Alternatively, note that decision boundary goes through the origin, so b=0. Final decision boundary line is therefore just the line $x_1 = x_2$, i.e., line through origin at 45 degrees, with slope 1.

Lagrange Multipliers Cheat Sheet

1. A method for finding the maximum or minimum or a function, subject to constraints.
2. Key idea: define a new function, L, in terms of the original function, $f(x,y)$, the constraint equation $g(x,y)$, and a new variable, the "Langrange multiplier" λ, which is a penalty for when the constraint equation is violated (not equal to zero), i.e., we want to maximize $f(x,y)$ s.t. $g(x,y)=0$.

$$L = f(x,y) + \lambda g(x,y)$$

3. To do this we take partial derivatives, with respect to x, y, and λ:

$$\frac{\partial L}{\partial x} = \frac{\partial f(x,y)}{\partial x} + \lambda \frac{\partial g(x,y)}{\partial x} = 0$$

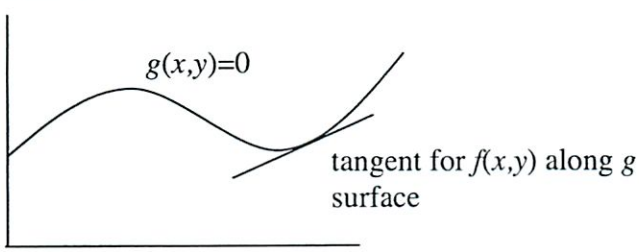
$$\frac{\partial L}{\partial y} = \frac{\partial f(x,y)}{\partial y} + \lambda \frac{\partial g(x,y)}{\partial y} = 0$$

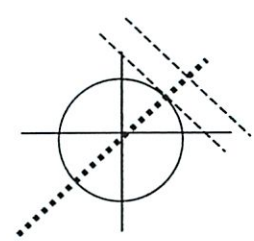
$$\frac{\partial L}{\partial \lambda} = g(x,y) = 0$$

4. Graphical intuition. As we travel along the constraint surface $g(x,y)=0$, clearly $g(x,y)$ does not change, so the partials of g with respect to both x and y must be 0, i.e., $\frac{\partial g(x,y)}{\partial x} = 0$ $\frac{\partial g(x,y)}{\partial y} = 0$

When we get to the maximum of $f(x,y)$, $\frac{\partial f(x,y)}{\partial x} = 0$ and $\frac{\partial f(x,y)}{\partial y} = 0$ because slope at max $=0$.

Picture:





Example: $f = x+y$; $g = x^2+y^2$ (constraint is a circle)

$L = (x+y) + \lambda(x^2+y^2-1)$

$$\frac{\partial L}{\partial x} = 1 + 2\lambda x = 0$$

$$\frac{\partial L}{\partial y} = 1 + 2\lambda y = 0$$

$$\lambda = \frac{-1}{2x}$$

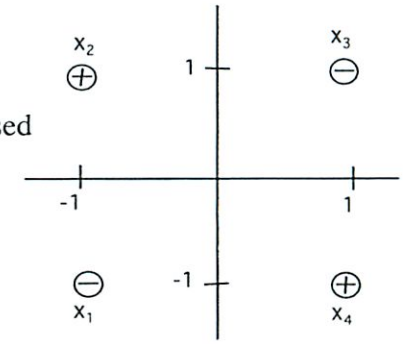
$$1 + 2\left(\frac{-1}{2x}\right)y = 0 \quad \Rightarrow y = x$$

From constraint as circle:

$$x^2 + y^2 = 2x^2 = 1 \quad \therefore \; x = \sqrt{1/2} = y$$

Appendix II. SVMs, another worked example, for a polynomial kernel function, XOR problem

The input data are shown in the graph in the original space.



For this problem, the polynomial kernel transform $K = (1 + v_1.v_2)^2$ is used

1. Note that $K(x_i\, x_j)=K(x_j\, x_i)$

$K(x_1,x_1) = (1 + [-1\ -1].[-1\ -1])^2$
$= (1 + \sqrt{2}\,\sqrt{2}\cos\theta)^2$ [or $(1 + (-1)(-1) + (-1)(-1))^2$]
$= (1 + 2)^2 = 9$

$K(x_1,x_2) = (1 + [-1\ -1].[-1\ +1])^2$
$= (1 + \sqrt{2}\,\sqrt{2}\cos\theta)^2$ [or alternatively: $(1+(-1)(-1)+(-1)(+1))^2$
$= (1 + 0)^2 = 1$

$K(x_1,x_3) = (1 + [-1\ -1].[+1\ +1])^2 = 1$; $K(x_1,x_4) = (1 + [-1\ -1].[+1\ -1])^2 = 1$
$K(x_2,x_2) = (1 + [-1\ +1].[-1\ +1])^2 = 9$; $K(x_2,x_3) = (1 + [-1\ +1].[+1\ +1])^2 = 1$
$K(x_2,x_4) = (1 + [-1\ -1].[+1\ -1])^2 = 1$
$K(x_3,x_3) = (1 + [-1\ -1].[-1\ -1])^2 = 9$; $K(x_3,x_4) = (1 + [-1\ -1].[+1\ -1])^2 = 1$;
$K(x_4,x_4) = (1 + [+1\ -1].[+1\ -1])^2 = 9$

2. Enter L_{Dual} as an algebraic expression of a_1, a_2, a_3, a_4.

Sum the alphas (a_1 to a_4), then sum over **all** dot products multiplied by their a's (alphas) and y_i multipliers, either +1 or –1 whether the data sample point is positive or negative (given in the data table).

$$L_{\text{Dual}} = \sum_i a_i - \tfrac{1}{2}\sum_i\sum_j a_i a_j y_i y_j x_i . x_j$$

$$= (a_1 + a_2 + a_3 + a_4) - \tfrac{1}{2}((a_1a_1(-1)(-1)K(x_1\, x_1) + a_1a_2(-1)(1)K(x_1\, x_2) + a_1a_3(-1)(-1)K(x_1\, x_3) + a_1a_4(-1)(1)K(x_1\, x_4) + a_2a_1(1)(-1)K(x_2\, x_1) + a_2a_2(1)(1)K(x_2\, x_2) + a_2a_3(1)(-1)K(x_2\, x_3) + a_2a_4(1)(1)K(x_2\, x_4)$$
$$+ a_3a_1(-1)(-1)K(x_3\, x_1) + a_3a_2(-1)(1)K(x_3\, x_2) + a_3a_3(-1)(-1)K(x_3\, x_3) + a_3a_4(-1)(1)K(x_3\, x_4)$$
$$+ a_4a_1(1)(-1)K(x_4\, x_1) + a_4a_2(1)(1)K(x_4\, x_2) + a_4a_3(1)(-1)K(x_4\, x_3) + a_4a_4(1)(1)K(x_4\, x_4))$$

$$=(a_1+a_2+a_3+a_4) - \tfrac{1}{2}(9a_1^2 - a_1a_2 + a_1a_3 - a_1a_4 - a_2a_1 + 9a_2^2 - a_2a_3 + a_2a_4 + a_3a_1 - a_3a_2 + 9a_3^2 - a_3a_4 - a_4a_1 + a_4a_2 - a_4a_3 + 9a_4^2)$$
$$=(a_1+a_2+a_3+a_4) - \tfrac{1}{2}(9a_1^2 - 2a_1a_2 + 2a_1a_3 - 2a_1a_4 + 9a_2^2 - 2a_2a_3 + 2a_2a_4 + 9a_3^2 - 2a_3a_4 + 9a_4^2)$$

3. You can optimize the above equation by setting to zero its partial derivatives with respect to each a_i.

$$dL/da_1 = 1 - 9a_1 + a_2 + a_3 - a_4$$
$$dL/da_2 = 1 + a_1 - 9a_2 + a_3 - a_4$$
$$dL/da_3 = 1 + a_1 - a_2 - 9a_3 + a_4$$
$$dL/da_4 = 1 + a_1 - a_2 + a_3 - 9a_4$$

4. Solving these simultaneous equations, yields $a_i = 1/8$. (As a check, $\Sigma a_i y_i = 0$, as it should.)

5. For a polynomial kernel, the degree of the polynomial and the original number of features determine the ultimate number of features in the transformed space (basically all combinations of input features up to the specified degree, with some terms scaled by the magnitude of the feature vector)

for $x_i = [a\ b]$ (a and b are just used here to designate first and second feature values, respectively)
$\text{Phi}(x_i) = [a^2\ \ b^2\ \ ab\sqrt{2}\ \ a\sqrt{2}\ \ b\sqrt{2}\ \ 1]$

Substituting values for the four data points:
x_1 [−1 −1]: $\text{Phi}(x_1) = [1\ \ 1\ \ \sqrt{2}\ \ -\sqrt{2}\ \ -\sqrt{2}\ \ 1]$
x_2 [−1 +1]: $\text{Phi}(x_2) = [1\ \ 1\ \ -\sqrt{2}\ \ -\sqrt{2}\ \ \sqrt{2}\ \ 1]$
x_3 [+1 −1]: $\text{Phi}(x_3) = [1\ \ 1\ \ -\sqrt{2}\ \ \sqrt{2}\ \ -\sqrt{2}\ \ 1]$
x_4 [+1 +1]: $\text{Phi}(x_4) = [1\ \ 1\ \ \sqrt{2}\ \ \sqrt{2}\ \ \sqrt{2}\ \ 1]$

6. The classification (aka weight) vector $\mathbf{w} = \Sigma a_i y_i x_i$:
x_1: $1/8(-1)[1\ \ 1\ \ \sqrt{2}\ \ -\sqrt{2}\ \ -\sqrt{2}\ \ 1]$
x_2: $1/8(1)[1\ \ 1\ \ -\sqrt{2}\ \ -\sqrt{2}\ \ \sqrt{2}\ \ 1]$
x_3: $1/8(1)[1\ \ 1\ \ -\sqrt{2}\ \ \sqrt{2}\ \ -\sqrt{2}\ \ 1]$
x_4: $1/8(-1)\ [1\ \ 1\ \ \sqrt{2}\ \ \sqrt{2}\ \ \sqrt{2}\ \ 1]$

Adding these vectors as per the above equation, $\mathbf{w} = [0\ \ 0\ \ -\frac{\sqrt{2}}{2}\ \ 0\ \ 0\ \ 0]$

7. To calculate b, substitute w and one of the data samples into any constraint equation:

$\mathbf{w}.x_1 + b = -1$

$\mathbf{w}\ .\ x_1$: $[0\ \ 0\ \ -\frac{\sqrt{2}}{2}\ \ 0\ \ 0\ \ 0]\ .\ [1\ \ 1\ \ \sqrt{2}\ \ -\sqrt{2}\ \ -\sqrt{2}\ \ 1] = 0 + 0 + -1 + 0 + 0 + 0 = -1$

so $-1 + b = -1$, so $b = 0$. Alternatively, you could notice that b passes through the origin, so it is 0.

8. To find an algebraic expression for the classifier function, $\mathbf{w}\ .\ \boldsymbol{x} + b$, substitute in Phi($x$) for $\boldsymbol{x}$:
($\mathbf{x} = [x_1\ x_2]$ instead of $[a\ b]$ as described above)

$$\mathbf{w}\ .\ \text{Phi}(\boldsymbol{x}) + b = (0{*}x_1 + 0{*}x_2 + -\frac{\sqrt{2}}{2}\sqrt{2}\,x_1x_2 + 0{*}\sqrt{2}\,x_1 + 0{*}\sqrt{2}\,x_2 + 0{*}1) + 0 = -x_1x_2$$

9. For each test point, substitute the vector into the above equation and see if we get the correct output:
$f([-1, -1]) = -(-1)(-1) = 1$ (correct, a positive sample point)
$f([-1, 1]) = -(-1)(1) = -1$ (correct, a negative sample point)
etc.
$f([2, 2]) = -4$

10. A fifth data point at [2 2] with a y_i value of -1 (a negative example) would have an a_i (i.e. a_5) value of 0 because it would **not** be a support vector. You can see this two ways: Its classifier value, calculated above, is -4, and we know support vectors have values of 1 or -1; or you could plot it on the graph and notice that it does not change the decision boundary.

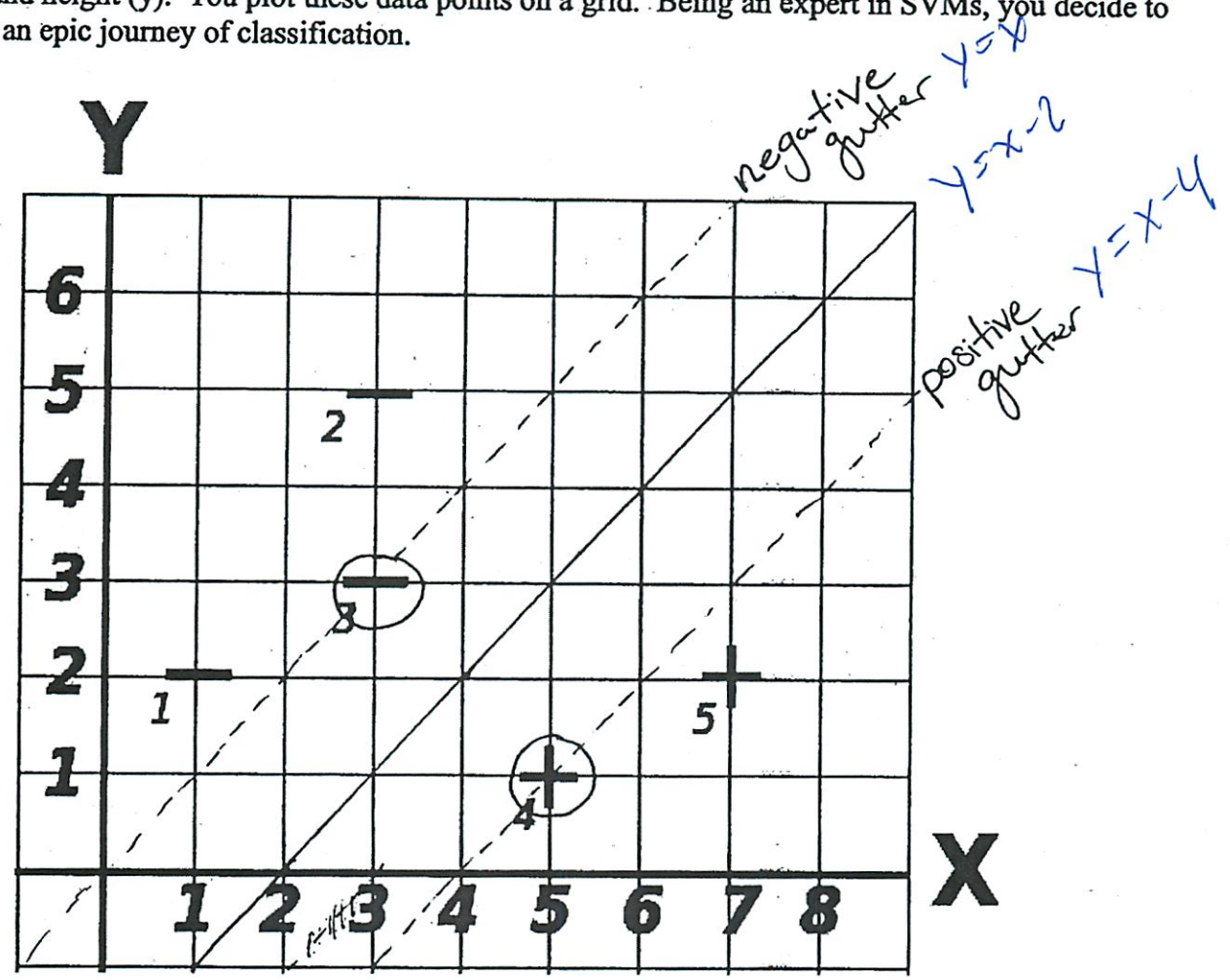
6.034 Recitation, Thursday, Nov. 10, Handout 2

Problem 1: SVMs (50 points)

After reading too much Lord of the Rings, you wake up to find yourself in Middle Earth. You decide the most relevant thing to do is to classify the different races around you.

Part A: Distinguishing Dwarves from Humans (38 points)

You meet 2 dwarves (+) and 3 humans (-) and realize that they have distinguishing features: beard length (x) and height (y). You plot these data points on a grid. Being an expert in SVMs, you decide to start off on an epic journey of classification.



A1 (7 points)

Draw the **decision boundary** on the graph above and **clearly label positive and negative gutters, and circle all support vectors.**

What is the width of the road/margin? _____

A2 (12 points)

Compute $\vec{w}$ and b in the decision boundary $h(\vec{u}) = \vec{w} \cdot \vec{u} + b \geq 0$ for the SVM solution to part A1.

Show your work here.

$\vec{w}$ =

b =

A2 (10 points)

Calculate the weights (alphas) of each data point.
Show your work here.

$\alpha_1 =$

$\alpha_2 =$

$\alpha_3 =$

$\alpha_4 =$

$\alpha_5 =$

A3 (9 points)

What will be the alpha of a new negative point 6 placed at (0, 6)?

What will be the alpha of a new negative point 6 placed at (0,0)?

Supposed we moved point 3 to (4, 2), how will the **magnitude of the alpha 3 change?**

Circle one:

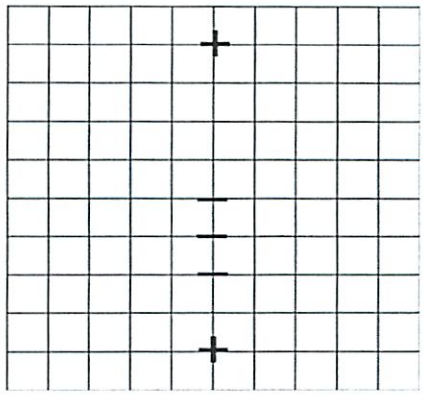
Larger Smaller Same

Part B: Distinguishing Kernels (12 points)

Back in his lab, Gandolf has been hacking on some kernels in preparation for greater classification adventures. For each of the following, indicate **YES** or **NO** whether the kernel can be used to ***perfectly*** classify the test points, and if **YES** ***sketch*** **the decision boundaries and gutters (the street) such a classifier might produce** and ***circle*** **which data points are support vectors**. Note that because of symmetry, more than one answer may be possible for one or more cases.

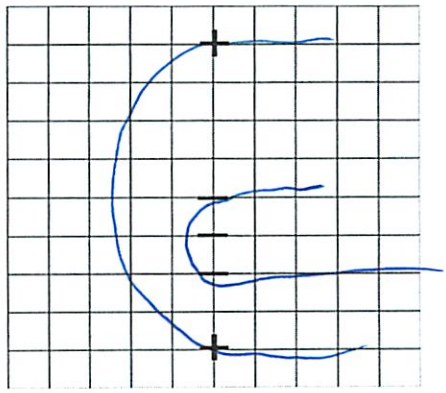
$$K(\vec{u}, \vec{v}) = \vec{u} \cdot \vec{v}$$

YES **NO**



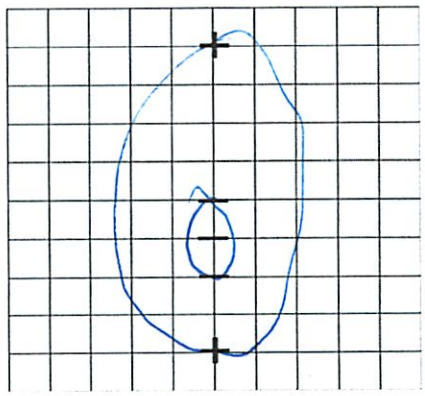
$$K(\vec{u}, \vec{v}) = (\vec{u} \cdot \vec{v} + 1)^2$$

YES **NO**



$$K(\vec{u}, \vec{v}) = e^{-(|\vec{u} - \vec{v}|)^2/2}$$

YES **NO**



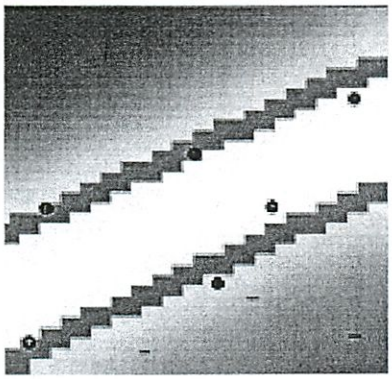
Final Exam 2002 Problem 6: Support Vector Machines (14 Points)

Part A: (2 Points)

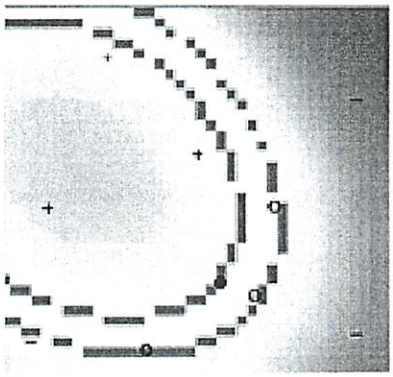
The following diagrams represent graphs of support vector machines trained to separate pluses (+) from minuses (-) for the same data set. The origin is at the lower left corner in all diagrams. Which represents the best classifier for the training data? ***See the separate color sheet for a clearer view of these diagrams.***

Indicate your choice here: ☐

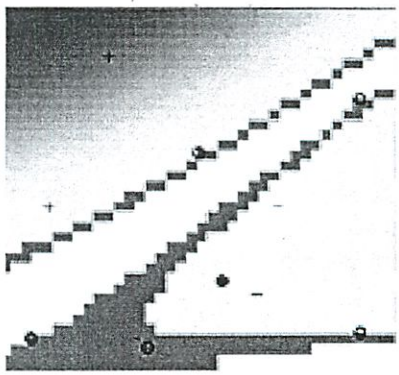
A.



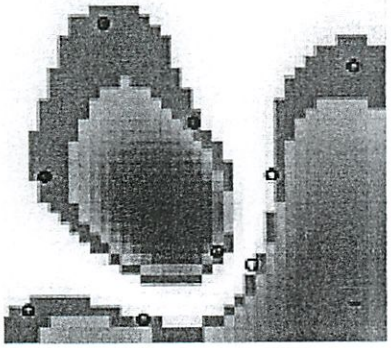
B.



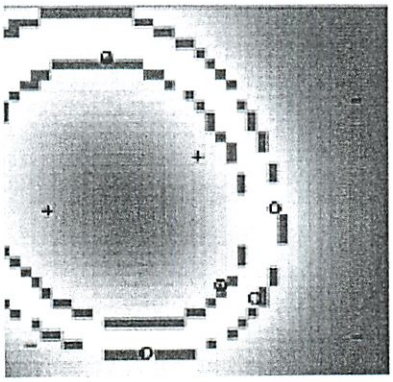
C.



D.



E.



Part B: (5 Points)

Match the diagrams in Part 1 with the following kernels:

Radial basis function, sigma .08 ☐

Radial basis function, sigma .5 ☐

Radial basis function, sigma 2.0 ☐

Linear ☐

Second order polynomial ☐

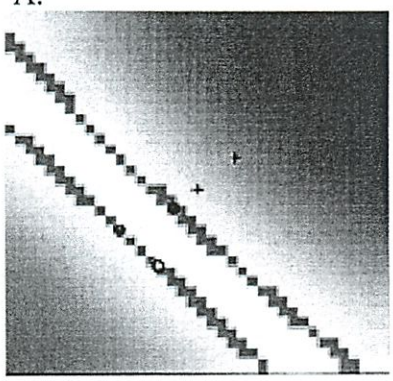
Part C: (3 Points)

Order the following diagrams from ***smallest*** support vector weights to ***largest*** support vector weights, assuming all diagrams are produced by the same mechanism using a linear kernel (that is, there is no transformation from the dot-product space).

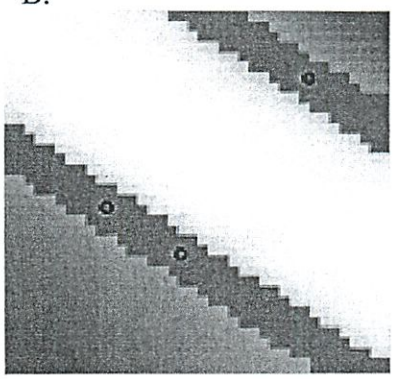
The origin is at the lower left corner in all diagrams. Support vector weights are also referred to as α_i values or LaGrangian multipliers. ***See the separate color sheet for a clearer view of these diagrams.***

Smallest	*Medium*	*Largest*

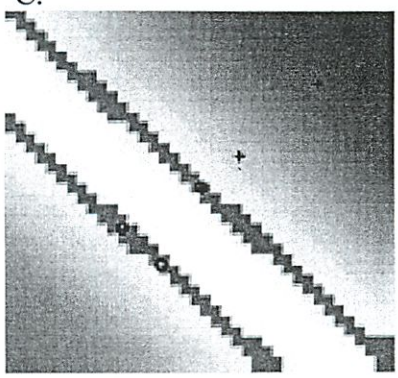
A.



B.



C.



Part D (4 Points)

Suppose a support vector machine for separating pluses from minuses finds a plus support vector at the point $\mathbf{x}_1 = (1, 0)$, a minus support vector at $\mathbf{x}_2 = (0, 1)$.

You are to determine values for the classification vector $\boldsymbol{w}$ and the threshold value b. Your expression for $\boldsymbol{w}$ may contain $\mathbf{x}_1$ and $\mathbf{x}_2$ because those are vectors with known components, but you are not to include any α_i or y_i. Hint: Think about the values produced by the decision rule for the support vectors, $\mathbf{x}_1$ and $\mathbf{x}_2$.

$\boldsymbol{w}$	
b	

Problem 2: Boosting (50 points)

After wearing Sauron's ring for several months, Frodo is rapidly losing his sanity. He fears that the ring will interfere with his better judgement and betray him to an enemy. To ensure that he doesn't put his trust into enemy hands, he flees Middle Earth in search of a way to classify his enemies from his friends. In his travels he had heard rumors of the magic of Artificial Intelligence and has decided to hire you to build him a classifier, which will correctly differentiate between his friends and his enemies. Below is all of the information Frodo remembers about the people back in Middle Earth.

ID	Name	Friend	Species	Has Magic	Part of the Fellowship	Has/Had a ring of power	Length of hair (feet)
1	Gandalf	**Yes**	Wizard	Yes	Yes	No	2
2	Sarumon	**No**	Wizard	Yes	No	No	2.5
3	Sauron	**No**	Wizard	Yes	No	Yes	0
4	Legolas	**Yes**	Elf	Yes	Yes	No	2
5	Tree-Beard	**Yes**	Ent	No	No	No	0
6	Sam	**Yes**	Hobbit	No	Yes	No	0.25
7	Elrond	**Yes**	Elf	Yes	No	Yes	2
8	Gollum	**No**	Hobbit	No	No	Yes	1
9	Aragorn	**Yes**	Man	No	Yes	No	0.75
10	Witch-king of Angmar	**No**	Man	Yes	No	Yes	2.5

Part A: Picking Classifiers (10 points)

A1 (6 points)

The data has a high dimensionality and so rather than trying to learn an SVM in a high dimension space you think it would be a smart approach to come up with a series of 1 dimensional stubs that can be used to construct a boosting classifier. Fill in the classifier table below. Each of the different classifiers are given a unique ID and a test returns +1 (friend) if true and -1 (enemy) if false.

Classifier	Test	Misclassified
A	Species is a Wizard	2, 3, 4, 5, 6, 7, 9
B	Species is an Elf	1, 5, 6, 9
C	Species is **not** a Man	2, 3, 8, 9
D	Does **not** have magic	1, 4, 7, 8
E	Is **not** part of the Fellowship	1, 2, 3, 4, 6, 8, 9, 10
F	Has **never** owned a ring of power	2, 7
G	Hair <= 1ft	1, 3, 4, 7, 8
H	Hair <= 2 ft	3, 8
I	Friend	2, 3, 8, 10
J	Enemy	1, 4, 5, 6, 7, 9

A2 (4 points)

Looking at the results of your current classifiers, you quickly see two more good weak classifiers (make fewer than 4 errors). What are they?

Classifier	Test	Misclassified
K		1, 8, 10
L		

Part B: Build a Strong Classifier (30 points)

B1 (25 points)

You realize that many of your tests are redundant and decide to move forward using only these four classifiers: {**B, D, F, I**}. Run the Boosting algorithm on the dataset with these four classifiers. Fill in the weights, classifiers, errors and alphas for three rounds of boosting. In case of ties, favor classifiers that come first alphabetically. Note: initial weights are set to be EQUAL and so 1/10 (they must add up to 1)

	Round 1		**Round 2**		**Round 3**	
w1	1/10	h_1 =F (why?)	F correct:	h_2 =		h_3 =
w2	1/10	Err =2/10	F incorrect:	Err =		Err =
w3	1/10	α =		α =		α =
w4	1/10					
w5	1/10					
w6	1/10					
w7	1/10					
w8	1/10					
w9	1/10					
w10	1/10					
Err(B)	/10					
Err(D)	/10					
Err(F)	2/10 WHY?					
Err(I)	/10					

So we pick F as our first 'stump' - why?

B2 (5 points)

What is the resulting classifier that you obtain after three rounds of Boosting?

H(x) = Sign[(1/2 ln) * F(x) + (1/2ln) * + (1/2ln)*

Part C: Boost by Inspection (10 points)

As you become frustrated that you must have picked the wrong subset of classifiers to work with, one of the 6.034 TA's, Martin, happens to walk by and sees your answer to part A1. He reminds you why the boosting algorithm works and then tells you that there is no reason to actually run boosting on this dataset. A boosted classifier of the form:

$$\mathbf{H(x) = Sign[h_1(x) + h_2(x) + h_3(x)]}$$

can be found which solves the problem. What three classifiers $\{h_1, h_2, h_3\}$ is Martin referring to, and why is the resulting H(x) guaranteed to classify all of the points correctly?

6.034 Lecture

11/14

The Symbolic Species

Frames + Representations

- [ ] Semantic Nets
- [ ] Classifications
- [ ] Transitions
- [ ] Trajectories
- [ ] Stories
- [ ] How to Write

* Don't use _____
* Don't use _____ + _____
* Don't chill a _____ a _____

Learning to play a bagpipe

Humans can figure out band's name

Why?

We can put concepts together

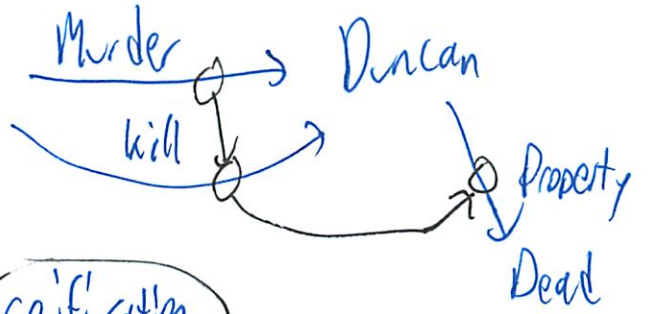
Right representations are very important

Later: How to write

②

Scemantic Nets

Must be supported by



reification

Can ~~rectify~~ the conections

└ take something abstract + make it real

↓

- make arrows nodes

Can also change the frame we looking at for knowledge

- isolate

Murder

Agent	Macbeth
Object	Duncan

slots

(3)

Can also sequence

$O \to O \to O \to O \to O \to O$

Gorillas don't start conversation

Can't sequence backward

Classification, Transition, Trajectory

1. Classification

if you know someone's class you know a lot about someone / something

professor

wall st trader

tool	General
↑	
hammer	Basic
↑	
Ball Peen hammer	Specific

How much knowledge at each level?

(4)

Musical instrument

↑

Piano

↑

Bosendopher

2. Transition

Someone wanted a book that answers question

How things work

What we learned about in engineering is bad to describe humans

- State transitions lead to other transitions

Need vocab of change

~~Speed~~

	Increasing	Decreasing	Changing	appearing	disappearing
	↑	↓	△	A	D
The Notsi	↑	↓	△	A	D

If car crashing into wall

↓

(5)

	T_1	T_2	T_3
Distance b/w Car and Wall	↓	D	A
Speed of Car	A	D	A
Condition of Car	A	A	A

time is not fixed or equal

Slowing down | hitting wall | sitting there crashed

Could use same f. describe Photon hitting camera
Just change the words above

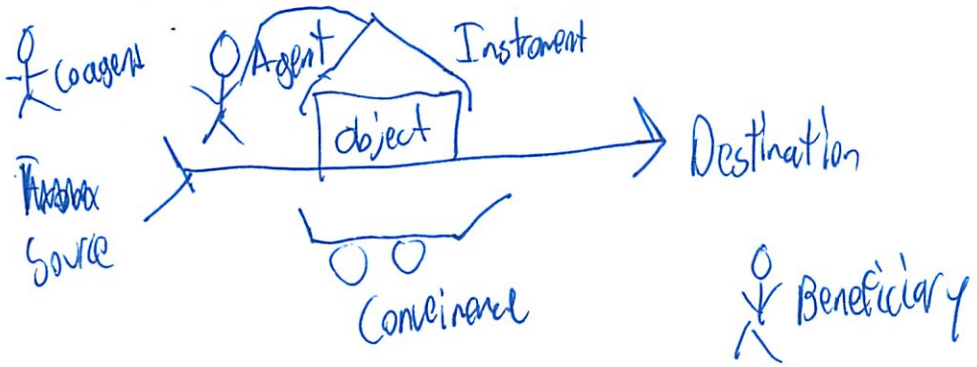
So this <u>exposes</u> what is happening

(6)

3. Trajectory

always talking about

"bird flu increase"



also called ~~the~~ role frames

in English: prepositions

to	destination
from	source
with	instrument
by	convenience
with	coagent
for	benificary
by	agent

But not unique

(7)

25% of sentances in the Wall St Journal have trajectories

A lot of MacBeth as well

---

example

Pat comforted Chris

Gender neutral names

Trajectory/Role frame

Action:

Agent: Pat

Object: Chris

Result:

← We don't know
- wrote big check
- gave narcotics

Transition

Chris's Mood ↑

(8)

Pat terrorized Chris

~~trajectory~~
Action ?
Agent Pat
Object chris
Result

Transition
Chris's Mood ↓

Pat kissed Chris

Trajectory
Object : Pat's lips
Destination : Chris's body

Trajectory/Role
Act
Agent Pat
Object Chris
Result

Pat Stabbed chris

Trajectory/Role
Action
Agent Pat
Object Chris
Result

Trajectory
Object : Pats' knife
Destination : Chris's Body

Transition
Chris' Mood ↓
Chris's Health ↓

(9)

Current project: put camera out into world
See what happening

Very easy for human
Still hard for computers

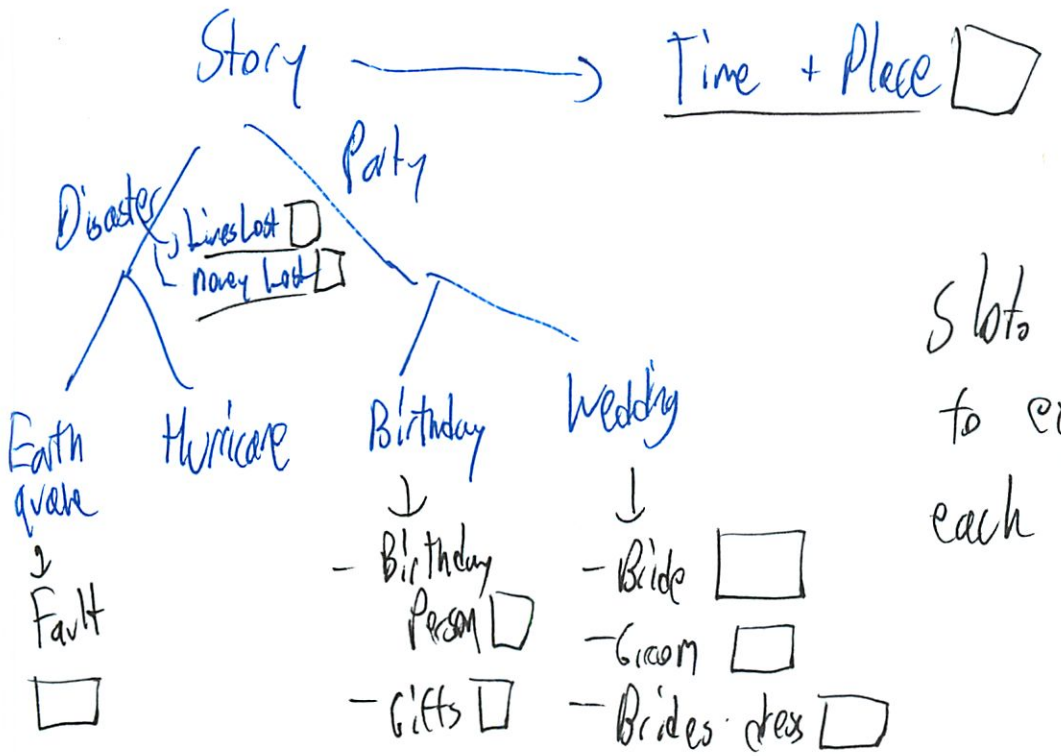
- fly
- run
- jump
- etc

Can use transition diagrams to think it out

But semantic nets undifferentiated
Can ~~[illegible]~~ put anything on label
And computer does not understand murder
- unless it has more data on that

Now can go higher: talk about frames of stories
~~Story~~

(10)



Slots we tend to expect for each story

But all those pronouns complicate it

A well written news story doesn't use pronouns

- fills in
- or other words like "Reliable Sources"

Same in technical writing

Pronouns put extra burden on lang processors

- Germans, Russians, Greeks think they can get away w/ pronouns
  - but they have genders - M, F, Neuter
  - divides possibilites in 3

(4)

Also no "former" or "later"

- People need to check

Also don't call a shovel a spade

- People don't want to use a word too many times
- call it by a consistent name each time
- in technical writing

* Don't use pronouns
* Dont use former or later
* Don't call a shovel a spade

6.034 Recitation

11/17

Quizzes back

2 handouts

1. Sum using k
   Special OH

2. Boosting

Breakdown

	Thorough	Adequate
P1	≥ 35	≥ 30
P2	≥ 35	≥ 29
P3	≥ 14	≥ 10
total	≥ 84	≥ 69

SVM, dot product kernal

kernal

- $k(\vec{U}, \vec{V}) = U \cdot V$
- linear
- how far apart or close they are

Notes handout page 1

2 key eq'n

like probability

uses data → to get intercept right

uses kernal → / inner product - trains system

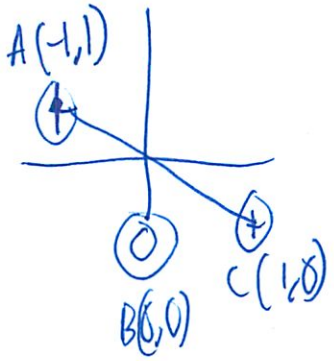
(1) $\sum \alpha_i y_i = 0$ partial of Lagrangian formation $\frac{\partial L}{\partial b}$ ← in notes

(2) $y = \sum_i \alpha_i y_i \cdot k(\bar{x}_i, \bar{x}) + b$ partial LaGrangian $\frac{\partial L}{\partial w_i}$ outputs ± 1

(2)

Want to find SVM values

Did graphically last time



i	j	kernel = distance similarity [dot product]
A	A	$(-1,+1)\cdot(-1,-1) = 2$
A	B	$(-1,+1)\cdot(0,0) = 0$
A	C	$(-1,+1)\cdot(1,0) = -1$
B	B	$(0,0)\cdot(0,0) = 0$
B	C	$(0,0)\cdot(1,0) = 0$
C	C	$(1,0)\cdot(1,0) = 1$

← just compute - don't need to transform pts to new space

Then

fill in

(1) $\alpha_A(1) + \alpha_B(-1) + \alpha_C(1) = 0$

(2) $y = \sum_i \alpha_i y_i k(\bar{x}_i, x) + b$

$+1 = \alpha_A k(x_A, x_A) - \alpha_B k(x_A, x_B) + \alpha_C k(x_A, x_C) + b$ ← y = 1 bc hidden

$-1 = \alpha_A k(x_B, x_A) - \alpha_B k(x_B, x_B) + \alpha_C k(x_B, x_C) + b$

$+1 = \alpha_A k(x_C, x_A) - \alpha_B k(x_C, x_B) + \alpha_C k(x_C, x_C) + b$

③

So just drop in current k() value
(From above table)

$$+1 = \alpha_A \cdot 2 - \alpha_C + b$$

$$-1 = b$$

since kernal fn is 0 for each term

$$+1 = -\alpha_A + \alpha_C + b$$

Now 4 eq 4 unknowns

$$\alpha_A = ? \quad \alpha_B = ? \quad \alpha_C = ? \quad b = ?$$

So have

$$b = -1$$

$$1 = 2\alpha_A - \alpha_C - 1$$

$$1 = -\alpha_A + \alpha_C - 1$$

$$2 = \alpha_A - 2$$

$$4 = \alpha_A$$

$$1 = -4 + \alpha_C - 1$$

$$6 = \alpha_C$$

(4)

Now use top eqn

$\alpha_B = 10$

Then write general solution

<u>pts</u>

A = -1, 1 (+)

B = 0, 0 (−)

C = 1, 0 (+)

<u>Boundry line</u>

$\boxed{\begin{matrix} +1 & >0 \\ -1 & <0 \end{matrix}}$

new data pt

$$h(x_1, x_2) = \mathrm{sgn}(\alpha_A k(x_A, x) + \alpha_B k(x_B, x) + \alpha_C k(x_C, x) + b)$$

So can write it out to get eqn

$$h(x_1, x_2) = \mathrm{sgn}\left(\alpha_A \begin{pmatrix} -1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \alpha_B \begin{pmatrix} 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \alpha_C \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - b\right)$$

$$= \mathrm{sgn}\left(4 \begin{pmatrix} -1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + 10 \begin{pmatrix} 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + 6 \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - 1\right)$$

5

$$= \operatorname{sgn}(-4x_1 + 4x_2 + 6x_1 - 1)$$

collect term

$$= \operatorname{sgn}(2x_1 + 4x_2 - 1)$$

This is boundry line

L "middle of road"

divides ⊕ from ⊖

$$2x_1 + 4x_2 - 1 = 0$$

$$x_2 = -\tfrac{1}{2}x_1 + \tfrac{1}{4}$$

But what is when ~~that~~ have diff kernal

- It just depends how we find kernal.

Can define

$$k(\vec{u}, \vec{v}) = (1 + \vec{u} \cdot \vec{v})^2$$

↑ parabolic

contour lines

(6)

Only thing that changes is computing difference

So for example

$$k(A,A) = (1 + A \cdot A)^2 \rightarrow 9$$

$$= (1 + 2)^2 \rightarrow 9$$

$$k(A,B) = (1 + A \cdot B)^2 \rightarrow 1$$

$$k(A,C) = (1 + A \cdot C)^2 \rightarrow 0$$

$$(1 + -1)^2 \rightarrow 0$$

$$k(B,B) = (1 + 0)^2 \rightarrow 1$$

$$k(B,C) = (1 + 0)^2 \rightarrow 1$$

$$k(C,C) = (1 + 1)^2 \rightarrow 4$$

(7)

Now same drill to solve

(not going to write)

└ in packet are examples

Boosting

- other handout - Boosting problem
- Classify Gandalf people
- Classify by species is very wrong
  - └ species is a wizard
  - └ makes 7 errors w/ 10 people → $\frac{7}{10}$ → very lousy

$$E = \frac{\#\text{ errors}}{\text{total data pts}}$$ ← initally all same weight

Boosting picks weak classifiers ($< \frac{1}{2}$)

└ not perfect

gets some right

but less $\frac{1}{2}$ wrong

8

$E = \frac{1}{2}$ ~~$E > \frac{1}{2}$~~

└ skip

$E > \frac{1}{2}$ – just flip

└ species is NOT a wizard

Asks for 2 other weak classifiers

$\sum$ weights $= 1$ $\quad$ $n = 10$ here (10 points)

initally all weights same $\frac{1}{n} \to \frac{1}{10}$

So
1. init weights
2. pick best stump (lowest error rate)
3. Find $\alpha = \frac{1}{2} \ln \frac{1-E}{E}$
4. Re weight data pts

(9)

In SVM we dropped most data points

↳ Here we keep them all

So then next classifier

↳ find error rate for each

Pick one w/ lowest error rate

↳ Classifier F

So now loop till perfect score or all $\frac{1}{2}$

Now find α

$$\alpha = \frac{1}{2} \ln_2 \left(\frac{1-E}{E} \right)$$ ← error rate

for f $\alpha = \frac{1}{2} \ln \left(\frac{1-\frac{2}{10}}{\frac{2}{10}} \right)$

$$= \frac{1}{2} \ln \left(\frac{\frac{8}{10}}{2/10} \right)$$

$$= \frac{1}{2} \ln (4)$$

↳ new weight

$$H(x) = \text{sgn} \sum_i \alpha_i h_i(x)$$

↳ stump

(10)

Now boost weights where classifier f got this ~~right~~ wrong
decrease " f " right

$$W'_{wrong_i} = \underbrace{\frac{1}{2} \cdot \frac{1}{E}}_{} \cdot W_i^{\text{original}}$$

new data pt weight

find factor to multiply old weight by to get new weight

E must always be $< \frac{1}{2}$

So $\frac{1}{E} > 2$

So $\frac{1}{2}\frac{1}{E} > 1$

So boosts weight

$$W'_{i\,correct} = \frac{1}{2} \cdot \frac{1}{1-E} \cdot W_i$$

So $\frac{1}{1-E} < 2$

So $\frac{1}{2}\frac{1}{1-E} < 1$

So decreases weight

(11)

Now try actual problem

Reweight data pts

So point 1

✓ was correct w/ f classifier

$$W_1' = \frac{1}{2} \cdot \frac{1}{1-E} \cdot W_1$$

$$= \frac{1}{2} \cdot \frac{1}{1-\frac{2}{10}} \cdot \frac{1}{10}$$

$$= \frac{1}{2} \cdot \frac{1}{\frac{8}{10}} \cdot \frac{1}{10}$$

$$= \frac{1}{16}$$ ← so weight went down

✗ was incorrect w/ f classifier

$$W_2' = \frac{1}{2} \cdot \frac{1}{E} \cdot W_2$$

$$= \frac{1}{2} \cdot \frac{1}{\frac{2}{10}} \cdot \frac{1}{10}$$

$$= \frac{1}{4}$$ ↗ weight boosted ↑

$$= \frac{4}{16}$$ ← write constant fractions

Do same for other pts

(12)

Check math

$= 2 \cdot \frac{4}{16} + 8 \cdot \frac{1}{16}$

$= \frac{1}{2} + \frac{1}{2}$ ← no significance

$= \underline{1}$

Now must recalculate error rates for B, D, F, I

See which has lowest error weight <u>given</u> new data weights

B gets 1, 5, 6, 9 wrong

└ add up these weights

└ all $\frac{1}{16}$

└ so total $\frac{4}{16}$

D gets 1, 4, 7, ?

$\frac{1}{16} + \frac{1}{16} + \frac{4}{16} + \frac{1}{16} = \frac{7}{16}$

etc

Pick lowest → B → find α for it ↓ <u>repeat</u>

error ~~rate~~ rate w/ reweighting

↳ update weights again

Massachvsetts Institvte of Technology
Department of Electrical Engineering and Computer Science
6.034 Recitation, Thursday, November 17

SVMs & Boosting, II — Prof. Bob Berwick, 32D-728

Part I. SVMs: solving for support vectors using the kernel function directly

Question 1. Instead of using the 'eyeball' technique, we can use the constraints of the SVM optimization problem to solve for the support vector 'alpha' values *directly*, and so find the decision boundary line. This is useful when we have no geometric picture, especially when we move to non-linear kernel functions. But first, let's do this for a simple case, to gain practice. Let us consider solving for the following SVM classifier:

$$h(\vec{x}) = \text{sgn}\left(\sum_i \alpha_i y_i K(\vec{x}_i, \vec{x}) + b\right)$$

where we will use first a **linear** kernel function $K(\vec{u}, \vec{v}) = \vec{u} \cdot \vec{v}$, i.e., the normal dot product, on the following data set with just 3 training data points, *A, B,* and *C*:

data point *A*: (–1, 1); classifier output: +1
data point *B*: (0, 0); classifier output: –1
data point *C*: (1, 0); classifier output: +1

Part A. Instead of doing this geometrically, we now explicitly compute the kernel function values for *each* of the 3 points, pairwise against one another (these are just dot product computations!)

i	j	$K(\vec{x}_i, \vec{x}_j)$
A	A	$(-1,+1)\cdot(-1,+1) = 2$
A	B	$(-1,+1)\cdot(0,0) = 0$
A	C	$(-1,+1)\cdot(1,0) = -1$
B	B	$(0,0)\cdot(0,0) = 0$
B	C	$(0,0)\cdot(+1,0) = 0$
C	C	$(+1,0)\cdot(+1,0) = +1$

Part B. Now we can find the system of linear equations governing the support vector alphas, α_A, α_B, and α_C that can be derived given the two key constraints given by the Lagrangian formulation of the SVM optimization problem:

Constraint 1. $\sum_i \alpha_i y_i = 0$

Which for the 3 points **with support** (ie, that are support vectors), *A, B, C,* we therefore have the equation:

$$\alpha_A(1) + \alpha_B(-1) + \alpha_C(1) = 0$$

Constraint 2. For all points **with support**, we have:

$$y = \sum_i \alpha_i y_i K(\vec{x}_i, \vec{x}) + b$$

For points *A, B, C,* we therefore have the summation over 3 terms implies 3 equations as follows (where we have switched the terms of $\vec{x}_i$ and $\vec{x}$ from the equation above, don't be deceived – the inner products are commutative):

$$+1 = \alpha_A K(x_A, x_A) - \alpha_B K(x_A, x_B) + \alpha_C K(x_A, x_C) + b$$
$$-1 = \alpha_A K(x_B, x_A) - \alpha_B K(x_B, x_B) + \alpha_C K(x_B, x_C) + b$$
$$+1 = \alpha_A K(x_C, x_A) - \alpha_B K(x_C, x_B) + \alpha_C K(x_C, x_C) + b$$

Part C. But we can now simplify these 3 equations, by plugging in the actual values for *K* as computed above, to get these 3 equations:

$$+1 = 2\alpha_A - \alpha_C + b$$
$$-1 = b$$
$$+1 = -\alpha_A + \alpha_C + b$$

Thus we immediately can solve for b=–1. Now we can solve for α_A by adding the first and third equations together, getting:

$$+2 = \alpha_A + 2(-1)$$
$$+4 = \alpha_A$$

Using this value for α_A, we can now find the value for α_C:

$$+1 = -4 + \alpha_C - 1$$
$$6 = \alpha_C$$

Finally, using *Constraint 1*, we can solve for α_B, since the sum from *Constraint 1* must be 0. So $\alpha_A + \alpha_C = \alpha_B$, therefore, $\alpha_B = 4 + 6 = 10$, and we have: α_A=4; α_B=10; α_C =6; b=–1.

Part D. Now, what is the SVM classifier equation for h given this answer? This follows directly from the boundary line definition in the SVM Winston notes:

$$\begin{aligned} h(x_1,x_2) &= \mathrm{sgn}(\alpha_A K(x_A,x) + \alpha_B K(x_B,x) + \alpha_C K(x_C,x) + b) \\ &= \mathrm{sgn}(4(-1,+1)\cdot(x_1,x_2) + 10(0,0)\cdot(x_1,x_2) + 6(+1,0)\cdot(x_1,x_2) - 1) \\ &= \mathrm{sgn}(4(-1,+1)\cdot(x_1,x_2) + 6(+1,0)\cdot(x_1,x_2) - 1) \\ &= \mathrm{sgn}(2x_1 + 4x_2 - 1) \end{aligned}$$

Note: the middle term with α_B has dropped out here because $K(x_B,x)$ is always 0 no matter what x is. This equation defines the boundary line midway between the gutters as well, all without having to draw anything:

$$2x_1 + 4x_2 - 1 > 0$$
$$x_2 = \frac{1}{2}x_1 + \frac{1}{4}$$

Part E. Let us see what happens when we *add* new training data points.
If we added a new point x_D at (2,0), with output y_D= +1, nothing would change, because this point would not have any support vector weight, and it would be correctly classified with what we have so far.
If instead y_D= –1, then everything would change, because this would force us to redefine where the decision boundary line goes.

Question 2. Non-linear kernels.
Now let's repeat this game with the *same* three data points, *but* with a **non-linear** kernel function, the **quadratic form**:

$$K(\vec{u},\vec{v}) = (1 + \vec{u}\cdot\vec{v})^2$$

Part A. Recall that this kernel will give parabolic shaped contour lines. So, first, we just have to go through the kernel computation as we did with the linear kernel, but now of course we are computing a different 'similarity' function, using the new kernel function. Note that we can make use of our old values of $u\cdot v$ from our previous table to speed up the computation, i.e., for point A, dot product of A with itself is just 2, from our table above:

i	j	$K(\vec{x}_i,\vec{x}_j) = (1+\vec{u}\cdot\vec{v})^2$
A	A	$(1+2)^2 = 9$
A	B	$(1+0)^2 = 1$
A	C	$(1+-1)^2 = 0$
B	B	$(1+0)^2 = 1$
B	C	$(1+0)^2 = 1$
C	C	$(1+1)^2 = 4$

Part B. As before, we can now set up a system of equations from the Lagrangian constraint solution.

Constraint 1. $\sum_i \alpha_i y_i = 0$

Constraint 2. For all points **with support**, we have:

$$y = \sum_i \alpha_i y_i K(\vec{x}_i, \vec{x}) + b$$

Just as before, these constraints work out to imply:

$$+1 = \alpha_A K(x_A, x_A) - \alpha_B K(x_A, x_B) + \alpha_C K(x_A, x_C) + b$$
$$-1 = \alpha_A K(x_B, x_A) - \alpha_B K(x_B, x_B) + \alpha_C K(x_B, x_C) + b$$
$$+1 = \alpha_A K(x_C, x_A) - \alpha_B K(x_C, x_B) + \alpha_C K(x_C, x_C) + b$$

Part C. But this time the *kernel* values are different! We plug in the K values from the quadratic 'inner product' as defined above. When we do this, and solve for the three alpha values (I shall leave this algebra to you this time), we get:

$$\alpha_A = \frac{8}{23}; \alpha_B = \frac{3}{23}; \alpha_C = \frac{18}{23}; b = -\frac{49}{23}$$

Part D. Now we can define the classifier function as we did before, with these new values for the alphas:

$$\begin{aligned} h(\vec{x} = (x_1, x_2)) &= \text{sgn}(\alpha_A K(x_A, x) + \alpha_B K(x_B, x) + \alpha_C K(x_C, x) + b) \\ &= \text{sgn}(\frac{8}{23}(1 + x_A \cdot x)^2 + \frac{3}{23}(1 + x_B \cdot x)^2 + \frac{18}{23}(1 + x_C \cdot x)^2 - \frac{49}{23}) \\ &= \text{sgn}(\frac{8}{23}(1 + \begin{bmatrix} -1 \\ +1 \end{bmatrix} \cdot x)^2 + \frac{3}{23}(1 + \begin{bmatrix} 0 \\ 0 \end{bmatrix} \cdot x)^2 + \frac{18}{23}(1 + \begin{bmatrix} +1 \\ 0 \end{bmatrix} \cdot x)^2 - \frac{49}{23}) \\ &= \text{sgn}\left(\frac{8}{23}(1 - x_1 + x_2)^2 + 0 + \frac{18}{23}(1 + \cdot x_1)^2 - \frac{49}{23}\right) \end{aligned}$$

3. Other spaces, other kernels

OK, now the other big win with SVMs has to do with the ease with which you can transform from one space to another, where the data may be more easily separable. The remaining questions all ask you about that, for **different** kinds of kernels (= different ways to compute inner products, or 'distance', aka, 'similarity' of two points). That is, we need to define $K(u,v) = \varphi(u) \cdot \varphi(v)$, the dot product in the transformed space. (You should try these out in Winston's demo program to see how the decision boundaries change.)

The basic kernels we consider are these:

1. Single linear kernel. These are just straight lines in the plane (or in higher dimensions). You should remember what perceptrons can and cannot 'separate' via cuts, and this tells you what linear kernels can do. (But see below under linear combination of kernels!!!)

$K(\vec{u}, \vec{v}) = (\vec{u} \cdot \vec{v}) + b$, e.g., $K(\vec{u}, \vec{v}) = (\vec{u} \cdot \vec{v})$ (ordinary dot product)

2. Polynomial kernel.

$K(\vec{u}, \vec{v}) = (\vec{u} \cdot \vec{v} + b)^n$, $n > 1$

eg., Quadratic kernel: $K(\vec{u}, \vec{v}) = (\vec{u} \cdot \vec{v} + b)^2$

In 2-D the resulting decision boundary can look parabolic, linear, or hyperbolic depending on which terms in the expansion dominate.

3. Radial basis function (RBF) or Gaussian kernel.

$$K(\vec{u},\vec{v}) = -\exp\left(-\frac{\|\vec{u}-\vec{v}\|^2}{2\sigma^2}\right)$$

In 2-D, the decision boundaries for RBFs resemble contour circles around clusters of positive and negative points Support vectors are generally positive or negative points that are closest to the opposing cluster. The contour space that results is drawn from the sum of support vector Gaussians. Try the demo to see.

When the variance or spread of the Gaussian curve σ^2 ('sigma-squared') is large, you get 'wider' or 'flatter' Gaussians. When it is small, you get sharper Gaussians. Hence, when using a small sigma-squared, the contour density will appear closer, or tighter, around the support vector points. In 2-D, as a point gets closer to a support vector, it will approach *exp*(0)=1, and as it gets farther away, it approaches *exp*(–infinity)= 0.

Note that you can **combine** several radial basis function kernels to get a perfect fit around **any** set of data points, but this will usually amount to a typical case of over-fitting – there are 2 free parameters for every RBF kernel function.

4. Sigmoidal (tanh) kernel. This allows for a combination of linear decision boundaries, like neural nets.

$$K(\vec{u},\vec{v}) = \tanh(k\vec{u}\cdot\vec{v}+b)$$

$$K(\vec{u},\vec{v}) = \frac{e^{k\vec{u}\cdot\vec{v}+b}+1}{e^{k\vec{u}\cdot\vec{v}+b}-1}$$

The properties of this kernel function: it is similar to the sigmoid function; it ranges from –1 to +1; it approaches +1 when $x >> 0$; and it approaches –1 when $x << 0$. The resulting decision boundaries are logical combinations of linear boundaries, not that different from second-layer neurons in neural nets.

5. Linear combinations of kernels (scaling or general linear combination).

Kernel functions are closed under addition and scaling by a positive factor.

Part II. Boosting and the Adaboost algorithm

0. The idea behind **boosting** is to find a weighted combination of s "weak" classifiers (classifiers that underfit the data and still make mistakes, though as we will see they make mistakes on less than ½ the data), h_1, h_2...,h_s, into a **single strong** classifer, $H(x)$. This will be in the form:

$$H(\vec{x}) = sign(\alpha_1 h_1(\vec{x}) + \alpha_2 h_2(\vec{x}) + \cdots + \alpha_s h_s(\vec{x})$$

$$H(\vec{x}) = sign\left(\sum_{i=1}^{s} a_i h_i(\vec{x})\right)$$

$$\text{where: } H(\vec{x}) \in \{-1,+1\}, h_i(\vec{x}) \in \{-1,+1\}$$

Recall that the *sign* function simply returns +1 if weighted sum is positive, and –1 if the weighted sum is negative (i.e., it classifies the data point as + or –).

Each training data point is **weighted.** These weights are denoted w_i for i=1, ..., n. **Weights** are *like* probabilities, from the interval (0, 1], with their **sum equal to 1. BUT** weights are **never 0**. This implies that **all data points have some vote** on what the classification shuld be, at all times. (You might contrast that with SVMs.)

The general idea will be to pick a single 'best' classifier h (one that has the lowest error rate when acting all alone), as an initial 'stump' to use. Then, we will **boost** the weights of the data points that this classifier **mis-classifies (makes mistakes on),** so as to focus on the next classifier h that does best on the re-weighted data points. This will have the effect of trying to fix up the errors that the first classifier made. Then, using this next classifier, we repeat to see if we can now do better than in the first round, and so on. In computational practice, we use the same sort of entropy-lowering function we used with ID/classifier trees: the one to pick is the one that lowers entropy the most. But usually we will give you a set of classifiers that is easier to 'see', or will specify the order.

In Boosting we always pick these initial 'stump' classifiers so that the error rate is strictly < ½. Note that if a stump gives an error rate greater than ½, this can always be 'flipped' by reversing the + and – classification outputs. (If the stump said –, we make it +, and vice-versa.) Classifiers with error exactly equal to ½ are useless because they are no better than flipping a fair coin.

1. Here are the definitions we will use.

Errors:

The error rate of a classifier s, E^s, is simply the sum of all the *weights* of the training points classifier h_s gets **wrong**.

$(1-E^s)$ is 1 minus this sum, the sum of all the *weights* of the training points classifier h_s gets **correct.**

By assumption, we have that:

$E^s < ½$ and $(1-E^s) > ½$, so $E^s < (1-E^s)$, which implies that $(1-E^s)/E^s > 1$

Weights:

α_s is **defined** to be ½ ln[$(1-E^s)/E^s$)], so from the definition of weights, the quantity inside the ln term is > 1, so all alphas must be positive numbers.

Let's write out the Adaboost algorithm and then run through a few iterations of an example problem.

2. Adaboost algorithm

Input: training data, $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$

1. **Initialize data point weights.**

   Set $w_i^1 = \frac{1}{n} \ \forall i \in (1, \ldots, n)$

2. **Iterate over all 'stumps':** for s=1, ..., T

   a. **Train base** learner using distribution w^s on training data.

   Get a base (stump) classifier $h_s(x)$ that achieves the lowest error rate E^s.

   (In examples, these are picked from pre-defined stumps.)

   b. **Compute the stump weight:** $\alpha_s = \frac{1}{2} \ln \frac{(1-E^s)}{E^s}$

   c. **Update weights** (3 ways to do this; we pick Winston's method)

   For points that the classifier **gets correct,** $w_i^{s+1} = \left[\frac{1}{2} \cdot \frac{1}{1-E^s}\right] \cdot w_i^s$

   (Note from above that $1-E^s > ½$, so the fraction $1/(1-E^s)$ must be < 2, so the total factor scaling the old weight must be < 1, i.e., the **weight of correctly classified points must go DOWN in the next round)**

   For points that the classifier **gets incorrect,** $w_i^{s+1} = \left[\frac{1}{2} \cdot \frac{1}{E^s}\right] \cdot w_i^s$

   (Note from above that $E^s < ½$, so the fraction $1/E^s$) must be > 2, so the total factor scaling the old weight must be > 1, i.e., the **weight of incorrectly classified points must go UP in the next round)**

3. **Termination condition:**

   If $s > T$ or if $H(x)$ has error 0 on training data or < some error threshold, exit;

   If there are no more stumps h where the weighted error is < ½, exit (i.e., all stumps now have error exactly equal to ½)

4. **Output final classifier:**

$H(\vec{x}) = sign\left(\sum_{i=1}^{s} a_i h_i(\vec{x})\right)$ **[this is just the weighted sum of the original stump classifiers]**

Note that test stump classifiers that are **never** used are ones that make more errors than some pre-existing test stump. In other words, if the set of mistakes stump X makes is a **superset** of errors stump Y makes, then Error(X) > Error(Y) is **always** true, no matter weight distributions we use. Therefore, we will **always** pick Y over X because it makes fewer errors. So X will **never** be used!

3. Let's try a boosting problem from an exam (on the other handout).

4. Food for thought questions.

1. How does the weight α^s given to classifier h_s relate to the performance of h_s as a function of the error E^s?
2. How does the error of the classifier E^s affect the new weights on the samples? (How does it raise or lower them?)
3. How does AdaBoost end up treating outliers?
4. Why is not the case that new classifiers "clash" with the old classifiers on the training data?
5. Draw a picture of the training error, theoretical bound on the true error, and the typical test error curve.
6. Do we expect the error of new weak classifiers to increase or decrease with the number of rounds of estimation and re-weighting? Why or why not?

Answers to these questions:

1. How does the weight α^s given to classifier h_s relate to the performance of h_s as a function of the error E^s?

Answer: The lower the error the better the classifier h is on the (weighted) training data, and the larger the weight α^t we give to the classifier output when classifying new examples.

2. How does the error of the classifier E^s affect the new weights on the samples? (How does it raise or lower them?)

Answer: The lower the error, the better the classifier h classifies the (weighted) training examples, hence the larger the increase on the weight of the samples that it classifies incorrectly and similarly the larger the decrease on those that it classifies correctly. More generally, the smaller the error, the more significant the change in the weights on the samples. Note that this dependence can be seen indirectly in the AdaBoost algorithm from the weight of the corresponding classifier α_t. The lower the error E^t, the larger α_t, the better h_t is on the (weighted) training data.

3. How does AdaBoost end up treating outliers?

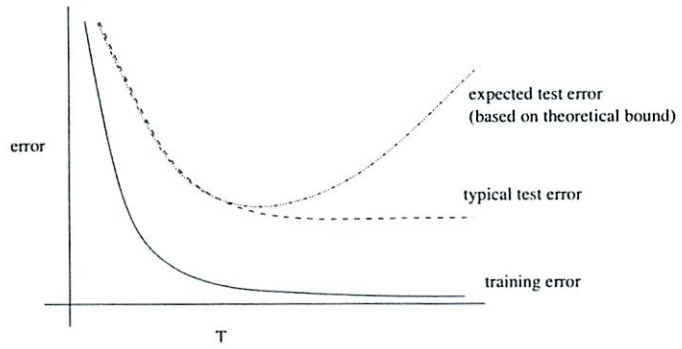
Answer: AdaBoost can help us identify outliers since those examples are the hardest to classify and therefore their weight is likely to keep increasing as we add more weak classifiers. At the same time, the theoretical bound on the training error implies that as we increase the number of base/weak classifiers, the final classifier produced by AdaBoost will classify all the training examples. This means that the outliers will eventually be "correctly" classified from the standpoint of the training data. Yet, as expected, this might lead to overfitting.

4. Why is not the case that new classifiers "clash" with the old classifiers on the training data?

Answer: The intuition is that, by varying the weight on the examples, the new weak classifiers are trained to perform well on different sets of examples than those for which the older weak classifiers were trained on. A similar intuition is that at the time of classifying new examples, those classifiers that are not trained to perform well in such examples will cancel each other out and only those that are well trained for such examples will prevail, so to speak, thus leading to a weighted majority for the correct label.

5. Draw a picture of the training error, theoretical bound on the true error, and the typical test error curve.

Answer:



6. Do we expect the error of new weak classifiers to increase or decrease with the number of rounds of estimation and re-weighting? Why?

Answer: We expect the error of the weak classifiers to increase in general since they have to perform well in those examples for which the weak classifiers found earlier did not perform well. In general, those examples will have a lot of weight yet they will also be the hardest to classify correctly.

11/17/11 Support Vectors (50 points)

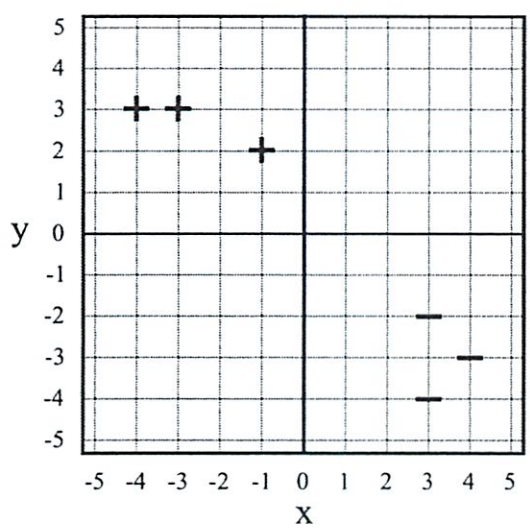
Part A (30 points)

The following three graphs represent the training data for two-dimensional, linear SVMs. + symbols indicate the coordinates of positive examples, which the classifier should give a value of 1 or more, and – symbols represent the coordinates of negative examples, which the classifier should give a value of -1 or less.

For each graph, you should do the following **three** things:

(a) Circle the support vectors
(b) Draw the "street" produced by the SVM, with solid lines where the classifier outputs 1 or -1 and a dashed line where the classifier outputs 0
(c) Describe the classifier function $h(\mathbf{x}) = \mathbf{w}\cdot\mathbf{x} + b$, by giving the values of $\mathbf{w}$ and b

Graph #1

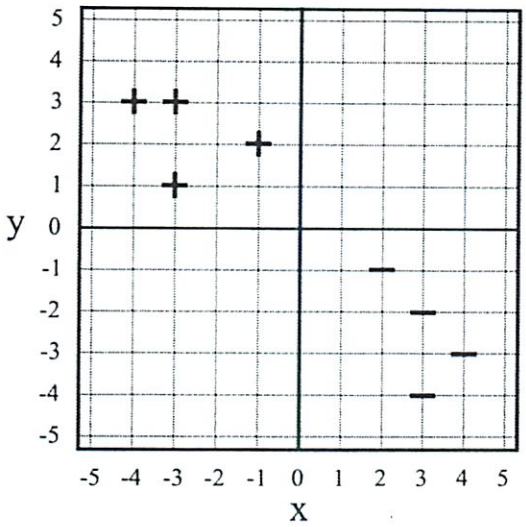


Step 1: circle support vectors (-1, 2) and ...(? , ?)
Step 2: use this to draw gutters & boundary line ($y \geq mx + b$)
$y \geq$ _____
Step 3: rewrite eqn of line so ≥ 0
Step 4: Find w values (2 of them) in matrix form, in general
Step 5: Use calculation of street width to solve for actual values for w and intercept b

$\mathbf{w} =$

$b =$

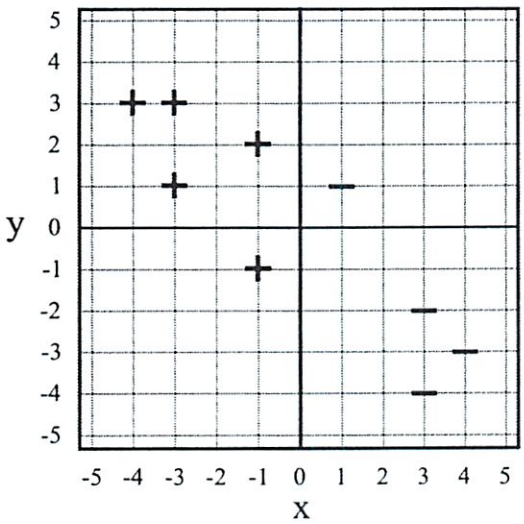
Graph #2



w =

b =

Graph #3

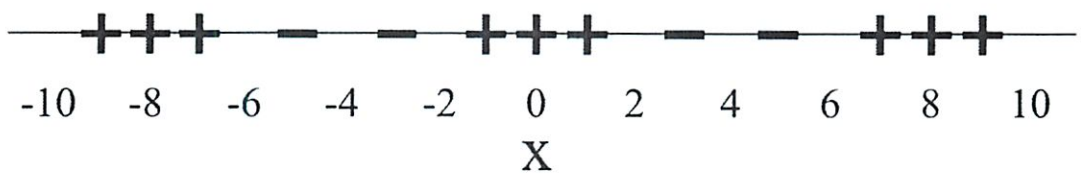


w =

b =

Part B (20 points)

Susan Q. Randomstudent is frustrated by the data she is supposed to classify with a support vector machine. It's all on the same line, defined only by an x-coordinate, and it switches back and forth from clusters of + points to clusters of – points several times, so it's not at all linearly separable.



Eventually, though, she figures out a kernel function that she can use to separate the data:

$$K(x_1, x_2) = \cos(\tfrac{\pi}{4}x_1)\cos(\tfrac{\pi}{4}x_2) + \sin(\tfrac{\pi}{4}x_1)\sin(\tfrac{\pi}{4}x_2)$$

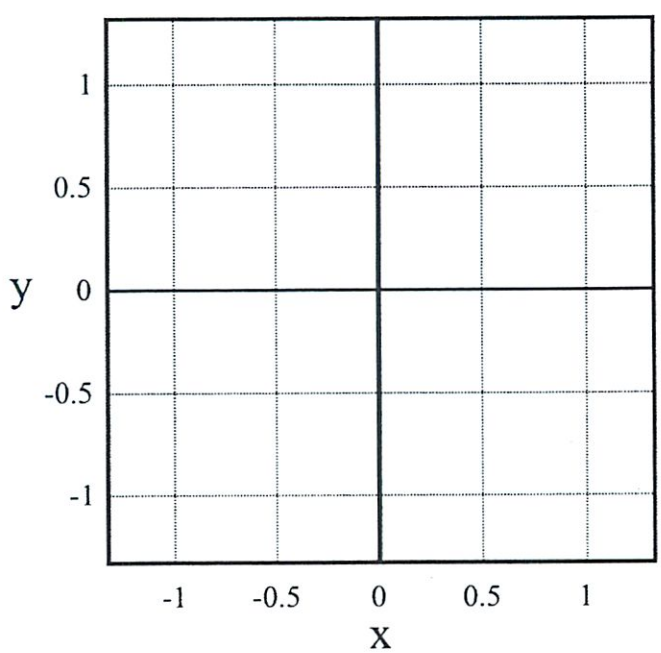
1. What is the Φ function that projects x into the new space defined by this kernel? (The function should take in x and output a vector with two components.)

   This means: WHAT function will yield the above form as its dot product?

   $\Phi(u) \cdot \Phi(v) = \cos(\text{blah})\cos(\text{blah}) + \sin(\text{blah})\sin(\text{blah})$

   $\Phi(x) =$

2. On the grid below, draw the + and – points in the new space, and draw the "street" that linearly separates them.



Problem 2: Boosting (50 points)

After wearing Sauron's ring for several months, Frodo is rapidly losing his sanity. He fears that the ring will interfere with his better judgement and betray him to an enemy. To ensure that he doesn't put his trust into enemy hands, he flees Middle Earth in search of a way to classify his enemies from his friends. In his travels he had heard rumors of the magic of Artificial Intelligence and has decided to hire you to build him a classifier, which will correctly differentiate between his friends and his enemies. Below is all of the information Frodo remembers about the people back in Middle Earth.

ID	Name	Friend	Species	Has Magic	Part of the Fellowship	Has/Had a ring of power	Length of hair (feet)
1	Gandalf	**Yes**	Wizard	Yes	Yes	No	2
2	Sarumon	**No**	Wizard	Yes	No	No	2.5
3	Sauron	**No**	Wizard	Yes	No	Yes	0
4	Legolas	**Yes**	Elf	Yes	Yes	No	2
5	Tree-Beard	**Yes**	Ent	No	No	No	0
6	Sam	**Yes**	Hobbit	No	Yes	No	0.25
7	Elrond	**Yes**	Elf	Yes	No	Yes	2
8	Gollum	**No**	Hobbit	No	No	Yes	1
9	Aragorn	**Yes**	Man	No	Yes	No	0.75
10	Witch-king of Angmar	**No**	Man	Yes	No	Yes	2.5

Part A: Picking Classifiers (10 points)

A1 (6 points)

The data has a high dimensionality and so rather than trying to learn an SVM in a high dimension space you think it would be a smart approach to come up with a series of 1 dimensional stubs that can be used to construct a boosting classifier. Fill in the classifier table below. Each of the different classifiers are given a unique ID and a test returns +1 (friend) if true and -1 (enemy) if false.

Classifier	Test	Misclassified
A	Species is a Wizard	2, 3, 4, 5, 6, 7, 9
B	Species is an Elf	1, 5, 6, 9
C	Species is **not** a Man	2, 3, 8, 9
D	Does **not** have magic	1, 4, 7, 8
E	Is **not** part of the Fellowship	1, 2, 3, 4, 6, 8, 9, 10
F	Has **never** owned a ring of power	2, 7
G	Hair <= 1ft	1, 3, 4, 7, 8
H	Hair <= 2 ft	3, 8
I	Friend	2, 3, 8, 10
J	Enemy	1, 4, 5, 6, 7, 9

A2 (4 points)

Looking at the results of your current classifiers, you quickly see two more good weak classifiers (make fewer than 4 errors). What are they?

Classifier	Test	Misclassified
K		1, 8, 10
L		

Part B: Build a Strong Classifier (30 points)

B1 (25 points)

You realize that many of your tests are redundant and decide to move forward using only these four classifiers:{**B**, **D**, **F**, **I**}. Run the Boosting algorithm on the dataset with these four classifiers. Fill in the weights, classifiers, errors and alphas for three rounds of boosting. In case of ties, favor classifiers that come first alphabetically. Note: initial weights are set to be EQUAL and so 1/10 (they must add up to 1)

	Round 1		**Round 2**		**Round 3**	
w1	1/10	h_1 = F (why?)	F correct:	h_2 =		h_3 =
w2	1/10	Err =2/10	F incorrect:	Err =		Err =
w3	1/10	α =		α =		α =
w4	1/10					
w5	1/10					
w6	1/10					
w7	1/10					
w8	1/10					
w9	1/10					
w10	1/10					
Err(B)	/10					
Err(D)	/10					
Err(F)	2/10 WHY?					
Err(I)	/10					

So we pick F as our first 'stump' - why?

B2 (5 points)

What is the resulting classifier that you obtain after three rounds of Boosting?

H(x) = Sign[(1/2 ln) * F(x) + (1/2ln) * + (1/2ln)*

Part C: Boost by Inspection (10 points)

As you become frustrated that you must have picked the wrong subset of classifiers to work with, one of the 6.034 TA's, Martin, happens to walk by and sees your answer to part A1. He reminds you why the boosting algorithm works and then tells you that there is no reason to actually run boosting on this dataset. A boosted classifier of the form:

$$\mathbf{H(x) = Sign[h_1(x) + h_2(x) + h_3(x)]}$$

can be found which solves the problem. What three classifiers $\{h_1, h_2, h_3\}$ is Martin referring to, and why is the resulting H(x) guaranteed to classify all of the points correctly?

6.034 Mega-Recitation

11/18

Silver Star Ideas

* Elimination
* Numerator stays the same Method

* $\alpha = \frac{1}{2} \ln \frac{1-E}{E}$
* if $\epsilon = \frac{1}{n} \rightarrow \frac{1}{2} \ln(n-1)$

Boosting

- hardest thing in class
- have a bunch of week classifiers

Vampires again

- often hard to determine if someone is a vampire at 1st glance

Can't use isVampire - just tells us

Step 1 - figure out possible classifiers for Vampire = Yes

A Evil = Y
B - Emo = Y
C - Transforms = Y.
D Sparkly = Y
E # Rom Interests ≥ 2
F " > 4
G True ← everyone vampire

H Evil = N
I Emo = N
J Transforms = N
k Sparkly = N
L # Rom Interests < 2
M " < 4
N False ← no one vampire

(2)

14 is a lot to check!

First write all the ones that are illegally classified

matches

A	2, 3, 4, 5 ← gotten wrong	H	1, 6, 7, 8, 9, 10
B	1, 6, 7, 9	I	2, 3, 4, 5, 8, 10
C	3, 4, 5, 8	J	1, 2, 6, 7, 9, 10
D	1, 2, 4, 5, 6, 7, 8	K	3, 9, 10
E	3, 10	L	1, 2, 4, 5, 6, 7, 8, 9
F	3, 4, 10	M	1, 2, 5, 6, 7, 8, 9
G	8, 9, 10	N	1, 2, 3, 4, 5, 6, 7

↑ the complement don't need to look at chart

Now we can ignore 6 of them

Which ones are good?

G, E, A, C, D

Having very few got wrong is very good, but no guarentee

↳ F looks good next - but always 1 more wrong than E

(3)

Also k is nice - but I more wrong than E

Classifiers to throw out - can never be good classifiers

- not just ones you don't like
  or pick pairs that appears to cover it

In fact you circle D!

- But it gets a bunch wrong
- But is only one that gets 3, 4, 10

Never selected to boost up front

But could work in long run

We were told to eliminate 6

- But could check all to see which are strictly worse than others
  - 3, 4, 5 strictly worse than 3, 5

(4)

Boosting

1. Assign everything $\frac{1}{N}$ weights

1	$\frac{1}{10}$
2	$\frac{1}{10}$
3	$\frac{1}{10}$
4	$\frac{1}{10}$
5	$\frac{1}{10}$
6	$\frac{1}{10}$
7	$\frac{1}{10}$
8	$\frac{1}{10}$
9	$\frac{1}{10}$
10	$\frac{1}{10}$

2. Look which classifier has smallest error

E 3, 10

$$\frac{1}{10} + \frac{1}{10} = \frac{1}{5}$$

3. Find α.

$$\alpha = \frac{1}{2} \ln\left(\frac{10}{n} - 1\right)$$

$$= \frac{1}{2} \ln(9)$$

(5)

4. Now write ones got wrong

1	2	3	4	5	6	7	8	9	10
$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$

His strategy: erase denominators (only if all the same denom) - then add up all the integers got wrong and ~~distribute~~ multiply by 2

1	2	3	4	5	6	7	8	9	10
		$\frac{1}{4}$							$\frac{1}{4}$

add all numerators of ones got right and multiply by 2

change to common denom

1	2	3	4	5	6	7	8	9	10
$\frac{1}{16}$	$\frac{1}{16}$	$\frac{4}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{4}{16}$

Now look again for which one gives you lowest error rate

If something $> \frac{1}{2}$ - do the opposite!

Can't select classifier 2x in a row (will be 50-50)

but can select later

(6)

Now next one B

$$\epsilon = \frac{1}{4}$$

$$\alpha = \frac{1}{2} \ln 3$$

1	2	3	4	5	6	7	8	9	10
$\frac{1}{8}$	$\frac{1}{24}$	$\frac{4}{24}$	$\frac{1}{24}$	$\frac{1}{24}$	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{24}$	$\frac{1}{8}$	$\frac{4}{24}$

Could change to common denom

Now next one

Can't be B

Have 3 = choices

But pick in alpha order → A

$$\epsilon = \frac{7}{24}$$

$$\alpha = \frac{1}{2} \ln\left(\frac{12}{7}\right)$$

↑ can use trick since ~~$\alpha \neq \frac{1}{n}$ everywhere~~

Other qu

$$\text{sign}\left(\frac{1}{2}\ln 4\,[E] + \frac{1}{2}\ln 3\,[B] + \frac{1}{2}\ln\frac{12}{24}\,[A]\right)$$

↑ 1 or −1 depending on what it is

then sum if ⊕ is yes
⊖ is no

↑ if 0 - edge case - by def of sun

(7)

Pick final classifier

When 3 classifiers

- One has α bigger than other 2 combined
  - pick that
- otherwise
  - best of majority vote (?)
  - whatever 2 of classifiers say

So run through each vampire canidate

- so if 2 say Y, 1 say N - then Y
  - & N Y - then N

Some still wrong

- Edward gets No
- The other $\frac{4}{10}$ gets it right

(8)

Then long concept qu of OR, AND

- more quickly + ~~[illegible]~~ less rounds of Boosting

- Would save rounds
- But get quadratically more classifiers to check
- And half # rounds
- So takes more time!

Lab 5

From 6.034 Fall 2011

Contents

General

Yeah!

This is the last problem set in 6.034! It's due Monday, November 21st at 11:59 PM.

To work on this problem set, you will need to get the code:

- You can view it at: http://web.mit.edu/6.034/www/labs/lab5/
- Download it as a ZIP file: http://web.mit.edu/6.034/www/labs/lab5/lab5.zip
- Or, on Athena, `add 6.034` and copy it from `/mit/6.034/www/labs/lab5/`

do later

You will need to **download and install** an additional software package called Orange (http://www.ailab.si/orange/) for the second part of the lab. Please download Orange first so that you get the problems worked out early. If you have downloaded and installed it, you should be able to run `orange_for_6034.py` and get the Orange version number. Once you've filled in the boosting part, you should be able to run `lab5.py`, and see the output of several classifiers on the vampire dataset. If you get errors, email us.

- Orange is available for Linux (Ubuntu), Windows, and OS X.
- For Ubuntu users: Please follow the install instruction here (http://www.ailab.si/orange/nightly_builds.html#linux) (see section on Linux). Note: just running `apt-get orange` on Ubuntu will *NOT* install orange but a completely different software package!
- ADVICE: If you can't get orange to work directly on your machine, try working on Athena instead. It may not be a wise use of your time trying to get the Orange software to work on your machine, as it is only needed for half of lab 5. If orange works right out of the box great, but if it doesn't then follow the Athena instructions to do the lab.

- To check that your Orange is properly installed, run:

```
python orange_for_6034.py
```

and you should get a version string and no errors.

To work on this lab on Athena you'll need to:

1. If you use bash: Setup your environment to pickup the class installation of Orange by adding the following:

```
export LD_LIBRARY_PATH=/afs/athena.mit.edu/course/6/6.034/lib/python2.5/dist-packages/orange:$LD_LIBRARY_PATH
```

to your .bashrc.mine file. If you are using tcsh or csh run:

```
setenv LD_LIBRARY_PATH /afs/athena.mit.edu/course/6/6.034/lib/python2.5/dist-packages/orange:$LD_LIBRARY_PATH
```

instead. You may omit `:$LD_LIBRARY_PATH` if you get complaints about it not being set. The above settings tell python where to look for shared libraries required for running Orange. NOTE: If you don't know what the shell you are running, type "echo $SHELL".
2. Log into linux.mit.edu: `ssh -X <user>@linux.mit.edu`
3. For your convenience, we've provided a script, `run-orange-gui.sh` that will run the Orange GUI on linux.mit.edu. Note that some Orange Widgets (like ROC charting) may not appear in GUI in the Athena version. Don't worry, you won't need the GUI to answer the questions for this lab.

Your answers for the problem set belong in the main file `lab5.py` as well as `neural_net.py` and `boost.py`.

Neural Nets

In this part of Lab 5, you are to complete the API for a Neural Net. Then afterwards, you are to construct various neural nets using the API to solve abstract learning problems.

Completing the implementation

We have provided you a skeleton of the Neural Net in `neural_net.py`, you are to complete the unimplemented methods.

The three classes, `Input`, `PerformanceElem`, and `Neuron` all have incomplete implementation of the following two functions:

```
def output(self)
def dOutdX(self, elem)
```

Your first task is to fill in all 6 functions to complete the API.

Output

The function `output(self)` produces the output of each of these elements.

Be sure to use the sigmoid and ds/dz functions as discussed in class:

```
o = s(z) = 1.0 / (1.0 + e**(-z))
ds(o)/dz = s(z) * (1 - s(z)) = o * (1 - o)
```

and also performance function and its derivative as discussed in class:

```
P(o) = -0.5 (d - o)**2
dP(o)/dx = (d - o)
```

Derivatives

The function `dOutDx(self, elem)` generates the value of the partial derivative, given a weight element.

Recall, neural nets update a given weight by computing the partial derivative of the performance function with respect to that weight. The formula we have used in class is as follows:

```
wi' = wi + rate * dP / dwi
```

In our code this is represented as (see `def train()` -- you don't have to implement this):

```
w.set_next_value( w.get_value() + rate * network.performance.dOutdX(w) )
```

The element passed to the dOutdX function is always a weight. Namely it is the weight that we are doing the partial over. Your job is to figure out how to define dOutdX() in terms of recursively calling dOutdX() or output() over the inputs + weights of a network element.

For example, consider the Performance element P, `P.dOutdX(w)` could be defined in the following recursive fashion:

```
dP/d(w) = dP/do  * do/dw          # applying the chain rule
        = (d-o)  * o.dOutdX(w)
```

Here o, is the output of the Neuron that is directly connected to P.

For Neuron units, there would be more cases to consider. Namely, 1) The termination case where the weight being differentiated over is one of the (direct) weights of the current neuron. 2) The recursive case where the weight is not one that is directly connected (but is a descendant weight).

right→left

This implementation models the process of computing the chain of derivatives recursively. This top-down approach works from the output layer towards the input layers. This is in contrast to the more conventional approach (taught in class) where ~~you com~~puted a per-node little-delta, and then recursively computed the weight updates bottom-up, from input towards output layers.

If you are confused about how the top-down recursive chaining of derivatives work, first read the course notes (http://courses.csail.mit.edu/6.034f/ai3/netmath.pdf) to review. If you are still confused, ask 6034tas for hints and clarifications.

About the API Classes

Most of the classes in the Neural Network inherit from the following two abstract classes:

`ValuedElement`

This interface class allows an element to have a settable value. `Input` (e.g. i1, i2) and `Weight` (e.g. w1A, wAB) are subclasses of `ValueElement`

Elements that are subclassed all have these methods:

- `set_value(self,val)` - set the value of the element
- `get_value(self)` - get the value of the element
- `get_name(self)` - get the name of the element

`DifferentiableElement`

This abstract class defines the interface for elements that have outputs and are involved in partial derivatives.

- `output(self)`: returns the output of this element
- `dOutdX(self,elem)`: returns the partial derivative with respect to another element

`Inputs`, `Neurons`, and `PerformanceElem` are the three subclasses that implement `DifferentiableElements`. You will have to complete the interface for these classes.

`Weight(ValuedElement)`

Represents update-able weights in the network. In addition to ValueElement functions are the following methods, which are used for the training algorithm (you will not need them in your implementation):

- `set_next_value(self,val)`: which sets the next weight value in self.next_value
- `update(self)`: which sets the current weight to the value stored in self.next_value

done for us

`Input(DifferentiableElement, ValuedElement)`

Represents inputs to the network. These may represent variable inputs as well as fixed inputs (i.e. threshold inputs) that are always set to -1. `output()` of Input units should simply return the value they are set to during training or testing.

`dOutdX(self, elem)` of an Input unit should return 0, since there are no weights directly connected into inputs.

`Neuron(DifferentiableElement)`

Represents the actual neurons in the neural net. The definitions for output and dOutdX already contains some code. Namely, we've implemented a value caching mechanism to speed up the training / testing process. So instead of implementing `output` and `dOutdx` directly you should instead implement:

- `compute_output(self):`
- `compute_doutdx(self,elem):`

) for caching

`PerformanceElem(DifferentiableElement)`

Represents a Performance Element that allows you to set the desired output.

- `set_desired` which sets my_desired_val

To better understand back-propagation, you should take a look at the methods **`train`** and **`test`** in `neural_network.py` to see how everything is put together.

Unit Testing

← not really unit testing each piece!

Once you've completed the missing functions, we have provided a python script `neural_net_tester.py` to help you unit test. You will need to pass the tests in this unit tester before you can move on to the next parts.

To check if your implementation works, run:

```
python neural_net_tester.py simple
```

This makes sure that your code works and can learn basic functions such as AND and OR.

Building Neural Nets

Once you have finished implementing the Neural Net API, you will be tasked to build three networks to learn various abstract data sets.

Here is an example of how to to construct a basic neural network:

```
def make_neural_net_basic():
    """Returns a 1 neuron network with 2 variable inputs, and 1 fixed input."""
    i0 = Input('i0',-1.0) # this input is immutable
    i1 = Input('i1',0.0)
    i2 = Input('i2',0.0)

    w1A = Weight('w1A',1)
    w2A = Weight('w2A',1)
    wA  = Weight('wA', 1)

    # the inputs must be in the same order as their associated weights
    A = Neuron('A', [i1,i2,i0], [w1A,w2A,wA])
    P = PerformanceElem(A, 0.0)

    # Package all the components into a network
    # First list the PerformanceElem P
    # Then list all neurons afterwards
    net = Network(P,[A])

    return net
```

Naming conventions

IMPORTANT: Be sure to use the following naming convention when creating elements for your networks:

Inputs:

- Format: `'i' + input_number`
- Conventions:
  - Start numbering from 1.
  - Use the same `i0` for all the fixed -1 inputs
- Examples: `'i1'`, `i2`.

Weights:

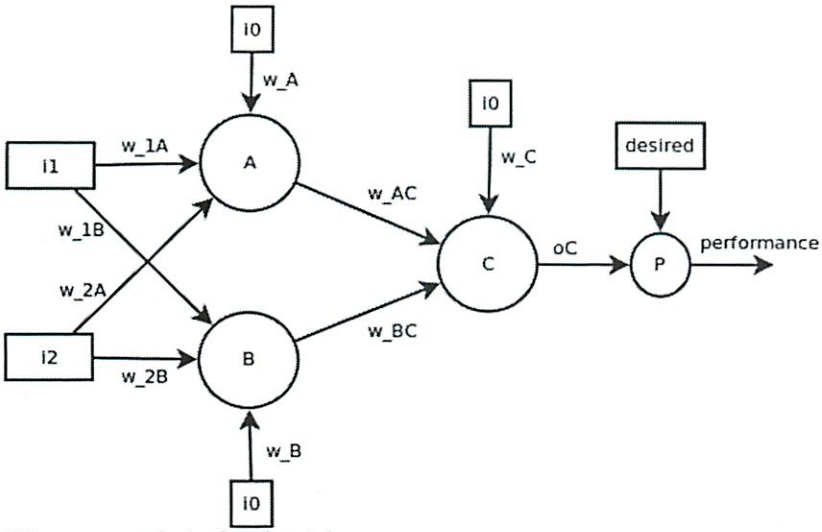
- Format `'w' + from_identifier + '.' + to_identifier`
- Examples:
  - `w1A` for weight from Input `i1` to Neuron `A`
  - `wBC` for weight from Neuron `B` to Neuron `C`.

Neurons:

- Format: `alphabet_letter`
- Convention: Assign neuron names in order of distance to the inputs.
- Example: `A` is the neuron closest to the inputs, and on the left-most (or top-most) side of the net.
- For ties, order neurons from left to right or top to bottom (depending on how you draw orient your network).

Building a 2-layer Neural Net

Now use the Neural Net API you've just completed and tested to create a two layer network that looks like the following.



Fill your answer in the function stub:

```
def make_neural_net_two_layer()
```

in `neural_net.py`.

Your 2-layer neural net should now be able to learn slightly harder datasets, such as the classic non-linearly separable examples such as NOT-EQUAL (XOR), and EQUAL.

When initializing the weights of the network, you should use random weights. To get deterministic random initial weights so that tests are reproducible you should first seed the random number generator, and then generate the random weights.

```
seed_random()

wt = random_weight()
...use wt...
wt2 = random_weight()
...use wt2...
```

Note: the function `random_weight()` in neural_net.py uses the python function `random.randrange(-1,2)` to compute initial weights. This function generates values: -1, 0, 1 (randomly). While this may seem like a mistake but what we've found empirically is that this actually performs better than using random.uniform(-1, 1). Be our guest and play around with the `random_weight` function. You'll find that Neural Nets can be quite sensitive to initialization weight settings. Recall what happens if you set all weights to the same value?

To test your completed network run:

```
python neural_net_tester.py two_layer
```

Your network should learn and classify all the datasets in the `neural_net_data.harder_data_set` with 100% accuracy.

Designing For More Challenging Datasets

Now it's your turn to design the network. We want to be able to classify more complex data sets.

Specifically we want you to design a new network that should theoretically be able to learn and classify the following datasets:

1. The letter-L.

```
4 + -
3 + -
```

```
2 + -
1 + - - - -
0 - + + + +
  0 1 2 3 4
```

2. This moat-like shape:

```
4 - - - - -
3 -       -
2 -   +   -
1 -       -
0 - - - - -
  0 1 2 3 4
```

3. This `patchy` shape:

```
4 - -   + +
3 - -   + +
2
1 + +   - -
0 + +   - -
  0 1 2 3 4
```

We claim that a network architecture containing 5 neuron nodes or less can fully learn and classify all three shapes. In fact, we require it!

Construct a new network in:

```
def make_neural_net_challenging()
```

that can (theoretically) perfectly learn and classify all three datasets.

To test your network on the first 2 of these shapes, run

```
python neural_net_tester.py challenging
```

To pass our tests, your network must get 100% accuracy within 10000 iterations.

Now try your architecture on the third dataset `patchy`. Run:

```
python neural_net_tester.py patchy
```

Depending on your architecture and your initial weights, your network may either easily learn `patchy` or get stuck in a local maxima. Does your network completely learn the dataset within 10000 iterations? If not, take look at the weights output at the end of the 10000 iterations. Plot the weights in terms of a linear function on a 2D graph. Do the boundaries tell you why there might be a local maxima?

Manually Setting Weights

You can have your network learn the dataset `patchy` perfectly and very quickly if the proper weights are set.

You can use either of these strategies to determine the proper weights.

1. You can experimentally determine the right weights by running your network until it perfectly learns the dataset. You will probably need to increase the max-iterations parameter, or playing around with different initial weight settings.
2. You can try to solve for the weights analytically. This is a good chance to put into practice the methods for solving network weights you learned in tutorial. You can review the example given in these TA notes (https://docs.google.com /View?docID=0AbO0KikZO04vZGhxaG0yYnFfMTA4dzducTl6YzM&revision=_latest) .

In either case, we want you to find preset weights to the same network that you built in the last part. Your new weight-preset network should be able to learn the `patchy` problem with only 1000 additional iterations of training.

After you've found the optimal weights, fill in:

```
def make_neural_net_with_weights()
```

To test, run:

```
python neural_net_tester.py weights
```

If everything tests with an accuracy of 1.0, then you've completed the Neural Networks portion of lab5. Congrats!

Now on to Boosting!

Boosting

You're still trying to use AI to predict the votes of politicians. ID-Trees were great, but you've heard about these other magnificent learning algorithms like SVMs and Boosting. Boosting sounds easier to implement and had a pretty good reputation, so you decide to start with that.

To make sure that you interpret the results without letting your political preconceptions get in the way, you dig up some old data to work with: in particular, the data from the 4th House of Representatives, which met from 1796 to 1797. (According to the records on voteview.com (http://www.voteview.com) , this is when the two-party system first emerged, with the two parties being designated "Federalists" and "Republicans".)

You experiment with that data before going on to the 2007-2008 data, finding that Congressmen in 1796 were much more clear about what they were voting on than in 2008.

The framework for a boosting classifier can be found in `boost.py`. You need to finish coding it, and then use it to learn some classifiers and answer a few questions.

The following resources will be helpful:

- The documentation for the boosting code, which you can find embedded in `boost.py` in the documentation strings.
- The Shapire paper on boosting (http://courses.csail.mit.edu/6.034f/ai3/msri.pdf) , or the notes (http://courses.csail.mit.edu/6.034f/ai3/boosting.pdf) that summarizes it.
- The Lab 4 writeup, if you need to refer to how `data_reader.py` represents legislators and votes.

A (clever|cheap) trick

The boosting code uses a trick that means it only has to try half the number of base classifiers.

It turns out that AdaBoost does not really care which side of its base classifier is +1 and which side is -1. If you choose a classifier that is the *opposite* of the best classifier -- it returns -1 for most points that should be +1, and returns +1 for most points that should be -1, and therefore has a high error rate -- it works the same as if you had chosen the negation of that classifier, which is the best classifier.

If the data reweighting step is implemented correctly, it will produce the same weights given a classifier or its opposite. Also, a classifier with a high error rate will end up with a *negative* alpha value, so that in the final "vote" of classifiers it will act like its opposite. So the important thing about a classifier isn't that its error rate is *low* -- it's that the error rate is *far from 1/2*.

In the boosting code, we take advantage of this. We include only classifiers that output +1 for voting YES on a particular motion, or +1 for voting NO on a particular motion, and as the "best classifier" we choose the classifier whose error rate is *farthest from 1/2*. If the error rate is high, then the result will act like a classifier that outputs +1 for "voting NO or abstaining", or +1 for "voting YES or abstaining", respectively. This means we don't have to include these classifiers in the base classifier list, speeding up the algorithm by a factor of 2.

Completing the code

Here are the parts that you need to complete:

- In the `BoostClassifier` class, the `classify` method is also undefined. Define it so that you can use a trained BoostClassifier as a classifier, outputting +1 or -1 based on the weighted results of its base classifiers. Complete the very similar `orange_classify` method as well.
- In the `BoostClassifier` class in `boost.py`, the `update_weights` method is undefined. You need to define this method so that it changes the data weights in the way prescribed by the AdaBoost algorithm. Note: There are two ways of implementing this update; they happen to be mathematically equivalent.)
- In `lab5.py`, the `most_misclassified` function is undefined. You will need to define it to answer the questions.

Remember to use the supplied `legislator_info(datum)` to output your list of the most-misclassified data points!

Questions

Answer the two questions `republican_newspaper_vote` and `republican_sunset_vote` in `lab5.py`.

When you are asked how a particular political party would vote on a particular motion, disregard the possibility of abstaining. If your classifier results indicate that the party *wouldn't* vote NO, consider that an indication that the party would vote YES.

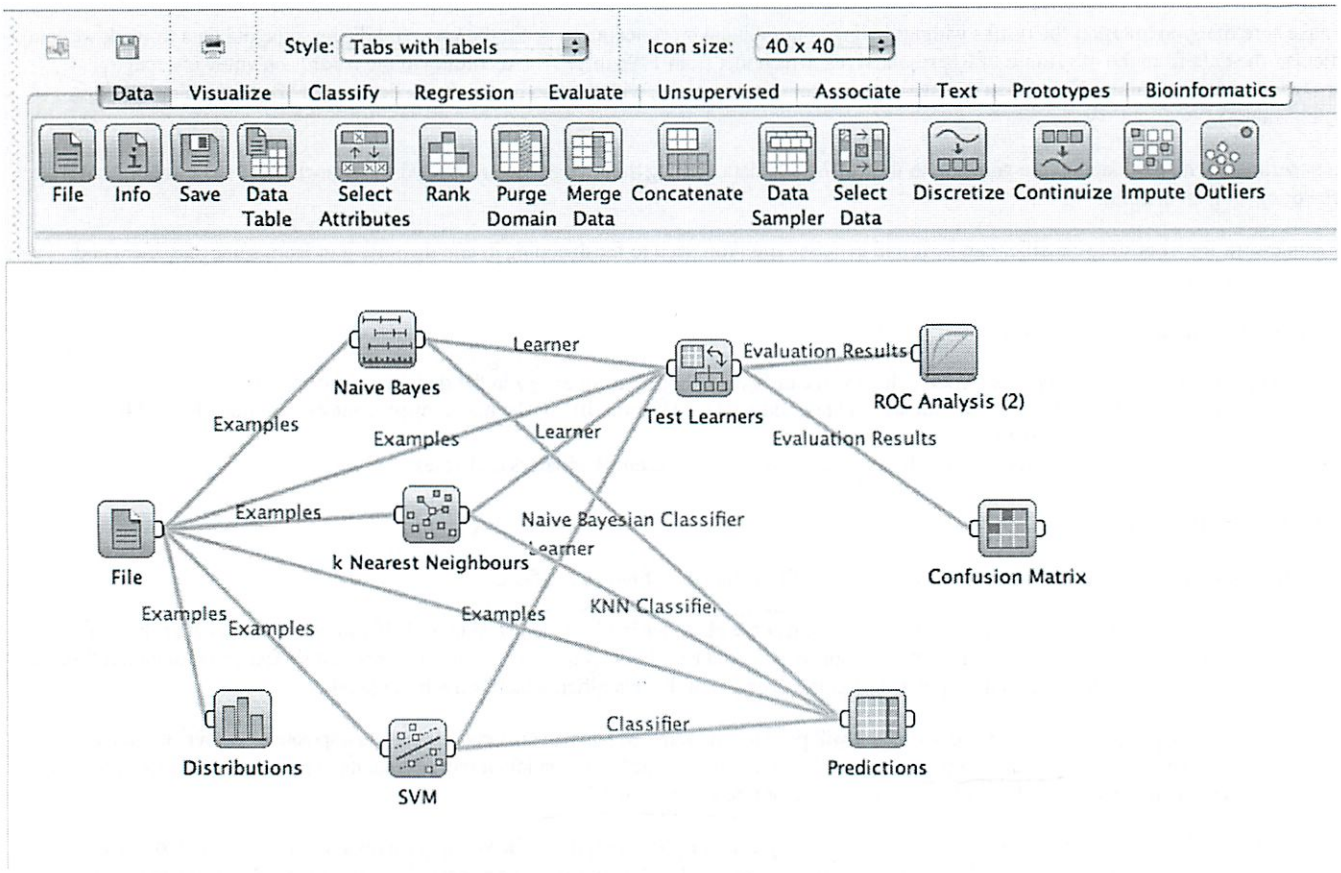
Orange you glad someone else implemented these?

First things first: Have you installed Orange yet?

Now that you've installed Orange, when you run lab5.py, does it complain about Orange, or does it show you the outputs of some classifiers on vampire data?

Getting familiar with the Orange GUI

This part is to get you familiar with the Orange GUI to do a little machine learning without doing any programming. We've given you some data files (vampires.tab, H004.tab, adult.tab, titanic.tab, breast-cancer.tab, etc.) that you can play with. Try making something like the following:



Then take a look at the performance, and look at the actual predictions.

Using Orange from Python

We have given you a function called `describe_and_classify` that trains a bunch of classifiers that Orange provides. Some of them will be more familiar than others.

First it trains each classifier on the data, shows its output on each data point from the training data, and shows the accuracy on the training data. You know from class that the accuracy on the training data should be 1 for these classifiers. It is less than one because each classifier comes with built-in regularization to help prevent overtraining. That's what the pruning is for the decision tree, for example. We didn't specify regularization parameters to most of the learners because they have fine defaults. You can read more about them in the Orange documentation.

You'll notice that we do one extra thing with the decision tree. We print it out. For most classifiers, their internal representations are hard for humans to read, but the decision tree's internal representation can be very useful in getting to know your data.

Then `describe_and_classify` passes the untrained learners, without having seen the data, to cross-validation. Cross-validation hides part of the training data, trains the learner on the rest, and then tests it on the hidden part. To avoid accidentally picking just one terrible subset of the data to hide (an irreproducible subset), it divides the data evenly into some number of folds, successively tests by hiding each fold in turn, and averages over the results. You will find with cross-validation that the classifiers do much better on identifying political parties than they do on vampires. That's because the vampire dataset has so few examples, with so little redundancy, that if you take away one example, it is very likely to remove the information you actually need in order to classify that example.

To ensure that you've gotten the right idea from each part of `describe_and_classify`, there are **six questions** just below the function.

Boosting with Orange

You may be able to do better by using AdaBoost over your Orange classifiers than you could do with any of those Orange classifiers by themselves. Then again, you may do worse. That AdaBoost "doesn't tend to overfit" is a somewhat conditional statement that depends a lot on the particular classifiers that boosting gets its pick of. If some of the underlying classifiers tend to overfit, then boosting will greedily choose them first, and it will be stuck with those choices for testing.

We have set up a learner that uses the BoostClassifier from the first part, but its underlying classifiers are the Orange classifiers we were just looking at. When you combine classifiers in this way, you create what's called an "ensemble classifier". You will notice, as you run this new classifier on the various data sets we've provided, that the ensemble frequently performs worse in cross-validation than some (or most) of its underlying classifiers.

Your job is to find a set of classifiers for the ensemble that get **at least 74% accuracy on the breast-cancer dataset**. You may use any subset of the classifiers we've provided. Put the short names of your classifiers into the list `classifiers_for_best_ensemble`. Classifier performance appears to be architecture dependent, so you might be able to get to 74% with just one classifier on your machine, but that won't be enough on the server -- in this case, try to get even better performance at home.

Hints

Problems with the tester

If you try running the tester on Athena (without ssh'ing into linux.mit.edu, as mentioned near the top of this page), you're likely to get errors like

```
Test 22/22 (classifiers_for_best_ensemble): Error.
While running the following test case:
        classifiers_for_best_ensemble
Your code encountered the following error:
Traceback (most recent call last):
  File "tester.py", line 199, in test_offline
    answer = run_test((index, type, fn_name, getargs), get_lab_module())
  File "tester.py", line 101, in get_lab_module
    raise ImportError, "Cannot find your lab; or, error importing it.  Try loading it by running 'python labN.py' (for the appropriate valu
ImportError: Cannot find your lab; or, error importing it.  Try loading it by running 'python labN.py' (for the appropriate value of 'N').
```

for every test case. If this is the case, use ssh -X <user>@linux.mit.edu (where <user> is your Athena username) and then cd to the directory containing your lab5 code and run the tester there.

Neural Nets

If you are having problems with getting your network to convergence on certain problems, try the following:

1. Change your random weight function to use random.randrange(-1,1) instead of random.uniform(-1,1). We've found that the former produced weights that are more conducive to network convergence.
2. Order your weight initialization (i.e. calls to random_weight()) from the bottom-most weights to the top-most weights in you network. While this ordering theoretically irrelevant, we've found that this ordering worked well in practice (in conjunction with 1 above). NN are unfortunately quite sensitive to initial weight settings. silly
3. Play around with the seed_random function to try different starting random seeds, although seeding the random function with 0 is what worked for us.
4. If none of these work, try setting weights that are close to what you want in terms of the final expected solution.

could do for last one

Errata

If you find problems in the lab description or the code, please email 6034tas@csail.mit.edu to let us know!

Doing 6.034

Lab 5

11/13

Final lab!

Do 1st half today

11/18

So do each part of neural net

Input

Performance Elem

Neuron

Output

└ so the out for input is just it?

Is valued Element object stored inside it?

But leave it for now

for neuron sum weights

derivative

deriv ~~th~~ of input w/ respect to ~~input~~ element??

~~oh so w/ respect~~

Ok so take partial deriv of performance fn w/ respect to that weight

(2)

$$w_i' = w_i + \text{rate} \cdot \frac{\partial P}{\partial w_i}$$

So have that in notes

$$\frac{\partial P}{\partial w_r} = \underbrace{(d - o_r)}_{\text{Perf}} \cdot \underbrace{o_r(1 - o_r)}_{\text{Sigmoid}} \cdot \underbrace{i_r}_{\text{input}}$$

↑ so same i_i

So deriv of Node calls output and/or dout/dx recursively on inputs

So

$$o_r(1 - o_r)$$

↑ need output

Well

$$\frac{\partial P}{\partial w_i} = (d - o_r) \times o_r(1 - o_r) \times w_r \times \underbrace{o_l(1 - o_l) \times i_i}$$

So basically this?

③

See multiple

$$\delta_{int} = o(1-o) \times \sum_j w_{ji} \rightarrow_{rj} \times \delta_{rj}$$

$o_f(1-o)(d-o)$

for final

or repeat

¿ just returns that section?

File not running -s says can't be found

└ the tester

Python IDLE wont load

Try Ubuntu

└ that will be delay to get it to load

¿ Or need orange to work?

Python is not importing anything

└ Orange screw it up?

③ 6

Python 2.6 on laptop

└ hmm same problem

Emailed in

¿Oh separate neural net tester?

Ok that gives me legit error

└ Go through that for now

So input to performance el is what?

So its a neuron

So I call get its output

So each part calls output of previous part

Oh valued element wrong - duh

See how input works

└ save?

¿Overflow error?

└ It works a lot of the time!

(4)

Oh how same issue finding e^{710} ↑ same exact value!

Decimal value?

↳ yeah that did it

Doing lots of training now

✓ Passes half tests

↳ Did I do something wrong?

↳ I guess

I wish we were doing unit tests

↳ read whole p-set 1st!

They want $\frac{dP}{d(w)} = \frac{dP}{dO} \cdot \frac{dO}{dw}$

$(d-O)$ ↑ it has to return next value?

~~↑ actually~~

~~$O(1-O)$?~~

no got wrong

O, dat dx(w)

(5)

Ok back prob was never right
So it was never fully used

(I wish had types!)

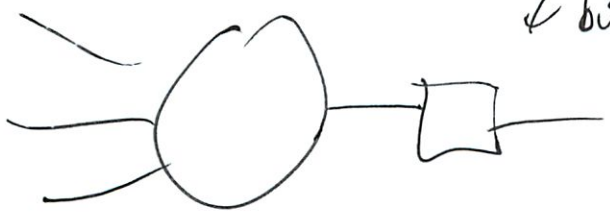
Ok fixed a bunch of syntax things

It still does not seem to change performance over time
↳ But all 4 are correct on OR ✓
2 correct on AND ↳ up from 2

So what is still wrong?
Special cases?
↳ when weight is weight of current neuron
↳ when dependent neuron

So what should I do differently?

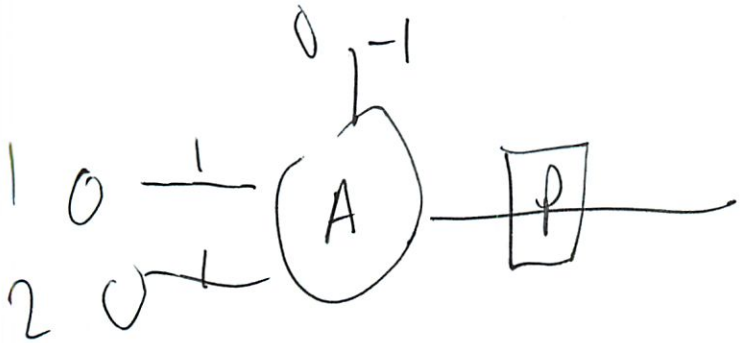
Follow examples



← but this isn't what testing

(6)

No thats it



So backward

$$\frac{\partial P}{\partial w_1} = d - o_r \cdot o_r (1 - o_r) \cdot i_r$$

$?_w$ that trying

Perf

neuron

calls on this

for input just value

calls $\frac{d}{dx}$ recursively

: does not do $o_r(1-o_r)$ left

So add that

Now values changing

(see just need to stop + think!)

Stuff is still wrong!

Set next value is done for us

(7)

Doing right on multi line?

~~(2)~~

Still $o(1-o) \times \sum$ _____

? So only do once!

(x) Still no good

? So <u>last</u> neuron does not need call get input

↳ but we need to get inst values

When are we using respective weight?

But this example just layer!

(so many formulas!)

So what am I doing wrong? 11/19

So weight · other inputs

↳ otherwise not using weight

Which weight?

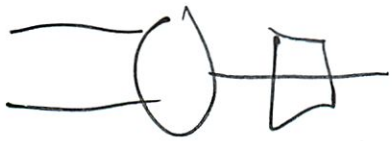
↳ the weight for that unit

When taking deriv ~~for 2 but 2~~ - w/ respect to weight

(8)

What does that mean?

So for our current system



$i_r \cdot \sigma(1-r) \cdot d-0$

when just one

but - <u>that's weight</u> times value!

$\left.\begin{matrix} \sigma_\ell \; w_\ell \\ \sigma_\ell \; w_\ell \end{matrix}\right\} \text{sum} \cdot \sigma_r(1-r)$

; Only go up to that weight + stop;

└ so that is termination

└ need to build that in

So possible elms

$\left.\begin{matrix} -w_A \\ -w_{1A} \\ -w_{2A} \end{matrix}\right\}$ what is difference b/w?

So wr is just up to that node?

w_{1A} w_{2A} - ignore?

(9)

So what to do for each one?
└description somewhere?

So in recitation example - we did not use w_r?
- just the input ones?

No cant w_A as one of them??
└ Does not seem to be any difference
? But when up to that node
└ only take <u>its</u> input

$\frac{\partial P}{\partial w_r}$ $d\text{-}o \cdot o_r(1-o_r) \times i_r$
? no weight!

(✓) Bingo thats it for 1 layer

Reading about API classes
└ would have been good earlier

Oh dOut/dx of input = 0
└ since we are treating specially

(10)

Now building neural Nets

example w/ basic

Certain conventions

Now build this 2 layer thing

Seed w/ random weights

(What happens again when all same value?)

└ I think something like a monoculture

I0 always −1 right?

Or random

└ well random weight

So inputs + desired tested later?

X Fails early

Why is elem 1?

Oh not weight get value

└ So my indirect way was not working

(11)

Now Neuron does not have get_value
output instead?
Or should this never have Neuron
Switching to output produces a lot of printing
so perhaps right

So it runs now but a lot of tests coming back wrong

And performance not updating
it should only run a few times to train

And + Or still right

=, ≠, horizontal not working
taking a long time
+ performance changing

W|A;B,—
seem crazy
~1200
- but ~~and~~ ~~OR~~
works w/ 86

(12)

So now back to reading

So should be

$$\frac{\partial P}{\partial \ell} = \underbrace{(d-o)}_{\text{Perf}} \times \underbrace{o_r(1-o_r)}_{\text{base Neuron}} \times w_r \times o_\ell(1-o_\ell)\, i_\ell$$

2 layer

its weight

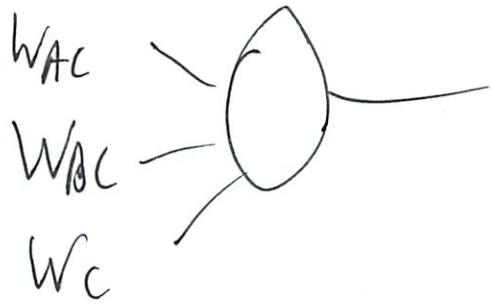
We want our weight

Oh fns on it decendent weight of

on the sum

$$\delta_{\ell_i} \quad \sum_j w_{\ell_i} \to r_i$$

So for C



So call over get_decendent_weights()

- only immediately decendent weights

Same as my_weights; for 1st level!

(3)

Are there cases when its <u>not</u> a decendent weights?

So saving them as normal?

well input line d.dx ~~cat~~ that weight

So for $\frac{dP}{dW_C}$

just 1 layer like normal

$\frac{dP}{dW_A}$? fid ~~dP~~ recurse into A

"Always C, A in print at"

C's nodes are wrong

L always B

L which has A in sets?

Oh never changed names!

That might have been it

L does it look at names

Yeah visiting B now

(14)

Now OR () training is ~~failing~~ going slow
└ But still right

Before I was essentiall had 
like before!

(✓) AND, OR
↑
But takes longer

= → 1 wrong

Oh but very high treshhold

Right weight?

So Wc is WAC if we do that to $\frac{dP}{w_{1a}}$
↑ multiply this!

Oh changing elem!
└ must be same!

for that elem – do we go there
or collect all?

Oh we have summ of all neurons to right
└ but we are coming from right

(15)

So really we just want to go to designated elem value

look to have right idea now

✓ Training much faster OR, AND

✗ Equal still wrong

So when say $\frac{dP}{dW_{BC}}$ I have output of neron B

So $(d-O_c)\, O_c(1-O_c) \times i_c$

input to this

O_c output of Neron B

for 3

$(d-O) \times O_c(1-O_c) \times w_c \times O_e(1-O_e) \times w_e \times O_a(1-O_a) \times i_a$

we don't have a 3 layer

So that should be right

Unless forward prop wrong...

(16)

Looked at ~~input~~ forward prop - seems right

└ emailed in

Should work - out of idea!

Why bad at = data, but good at 1 layer data?

Work on Boosting for now

└ get orange working

Need to add path

And Need 32 bit version

Oh for neural net - still need $(1-O)_O$

└ no makes things worse! AND, OR slower

Oh but everything else is correct ✓

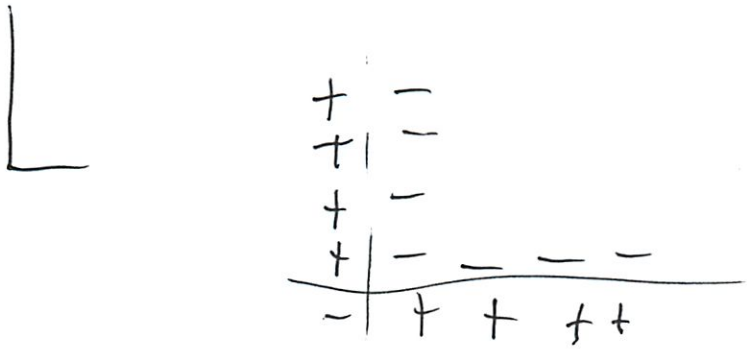
So I figured it out

X urg one still wrong

└ but almost all right

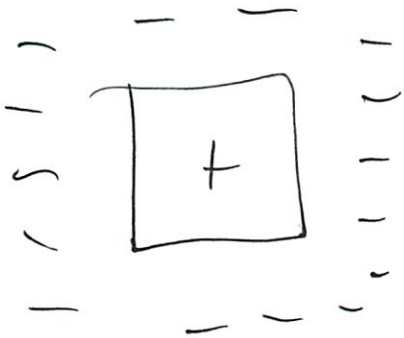
(17)

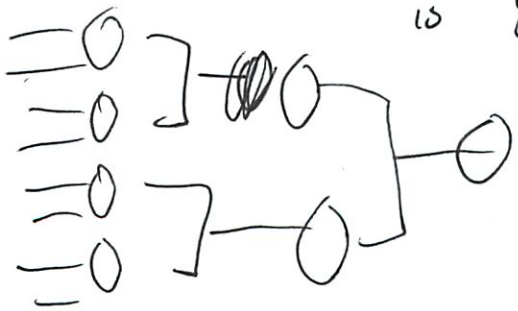
Build other neural nets



so 2 cuts horiz

for not shape 4 cuts





is should do it over do it

(18)

But at most 5 neuron units

So base

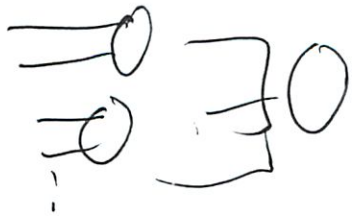
2-layer

can make 2 cuts any line?

So can do any 1 line

can do any 2 lines

we need 4 lines



(19)

That is working for a lot of it

(✓) I think for everything

Pathy

try running as is

Worked - save those weights!

2000 training

2227 now

save and ten se

(✓) should be done

11/19 lab

Boosting

(in Athena lab)

why are we ssh to linux.mit.edu ?

ahh I see GI!

(20)

implement Boost Classify

└ so whole thing?

or it does most of the work?

No pretrained - so add weighted total?

$$\text{Sign}\left(\sum_{i=1}^{S} a_i h_i(x)\right)$$

(oh on last page it tells you to run test code by ssh'ing into Athena?)

How do you test?

What is a_i, h_i?

What is classifier object?

? Oh built in sigmoid fn?

Need most misclassified

└ don't get how to test!!!

└ so annoying!!!

└? run lab5.py?

get some error

(21)

Must run through ssh I think....

So ones w/ highest weights should be last

how to do that?

Oh no - stupid in there

Lots of bugs in this print fn

Ah seems to work now....

How to print boosted items?

How to test online w/ linux?

Loh IDLE - but that is not SSH

Oh add "submit"

↳ Got 4/23

Should see some vampire data on lab 5

↳ I see senate data

(22)

Can you SSH GUI apps?

↳ Oh needs x to bind to

Ie Athena

How to open?

↳ File

How to run?

↳ Report?

Describe + classify

So what to do next - where?

Ohh they are doing a lot of classifiers

How to actually run???

So in Py 2.6 Decimal won't read float

↳ Does in 2.7 - do something to get it to work

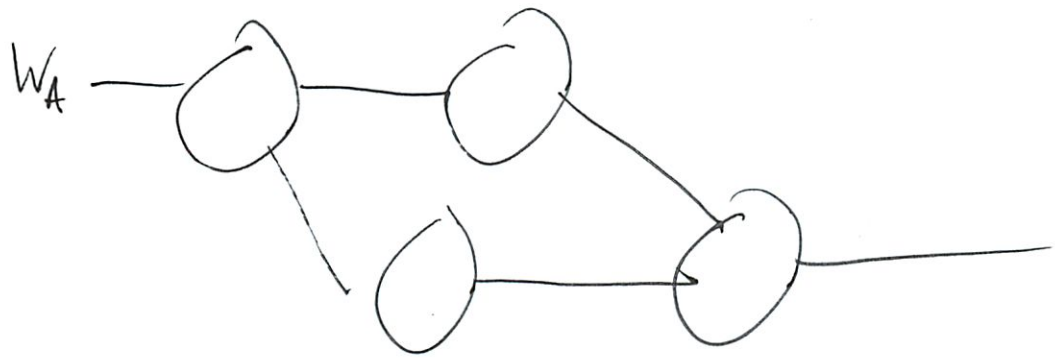
↳ Fixed - Neural net works

(23)

Neural nets failing in some places

TAs wrote back: need to take into account
an elem can feed into multiple inputs
See delta formula

So



So $W_{l_i} \rightarrow W_{r_i}$
Since that goes $l \rightarrow r$
we go $r \rightarrow l$

So need to add all items w/ that as dependent node

(✓) Pass 13/22 tests off line before fix

try neural net tester
└ wrong!

(24)

Oh did D-O before

Neural net very wrong
- worked better before-ish

So is it summing?
or my decimal fails w/ str state

Copy ~~my~~ ~~of~~ the other sigmoid fn
└ works much faster!

And accurately on 2.6

So still correct ~~for~~ except for 1

Ah things a lot more correct!

Before 2.71 online - took hr to run!

Tests run soooo much faster ~~online~~ now
└ when things right

Classifyer tests wrong now
and └ N.N. not implemented?
20/24 4.0! woot
└ oh never committed...

(25)

I have no clue how score ↑ that test

So neural net should work now

fix the other things → boosting and most misclassified

I think my classifyer is off
↳ throwing off top 5

On return -1 not 0

Most misclassified still wrong

Other stuff right

(✓) All except one question was not right offline

(✓) 5.0/5.0 online Jow!

6.034 Lecture

11/21

- ☐ Architectures
- ☐ General Problem Solver
- ☐ State Operator And Result
- ☐ Emotion Machine
- ☐ Subsumption
- ☐ Genesis

* Methods enabled by Constraints exposed by Representations supporting Models of thinking

Turing → Minsky
Turing → McCarthy
Turing → Newell
Turing → Simon

What it takes to figure this out?

We talked about representations + models

But how can you put it all together?

(2)

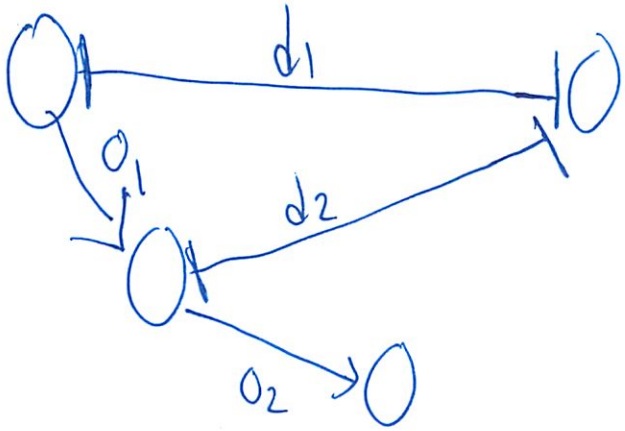
All these famous AI guys are gone now
Except I

How they thought?

How to put this stuff together?

General Problem Solver

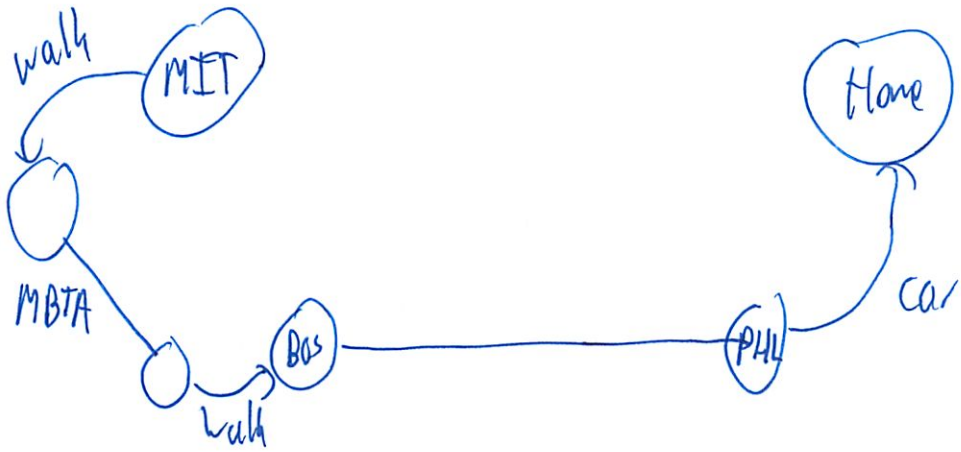
So want to reduce difference b/w here and goal



Difference \ Operators				
	✓			
		✓		
			✓	
				✓

③

Like if want to go here to home



This was Newell + Simon

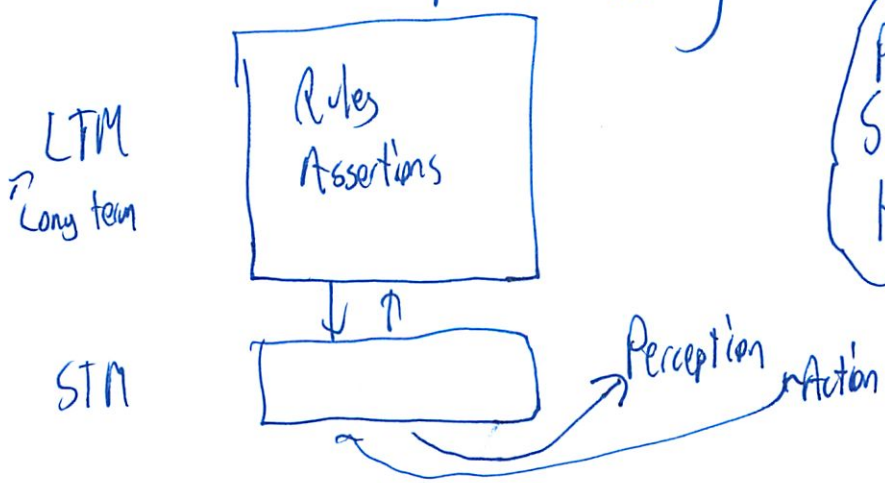
Chain reaction - things would be smaller + smaller

SOAR

GPS was not working, so new architecture

acronym does not mean anything now

How do we do problem solving



Physical Symbol Hypothesis

(4)

Convinced humans are Symbol Hypothesis processes

System has

1. STMs + LTMs
2. Rules + Assertions
3. Preference ↳ if one rule > other
4. Problem Space
   ↳ if had an impasse
5. Subgoal
6. Chunking ↳ puting rules together for macro rule

match

It seems like everything here was done already

1. Cognitive Science

2+3. AI

4+5. GPS

6. Macro Operators @ Stanford

(5)

In <u>Emotion Machine</u> book, a few levels

Self - Conscias Emotions - ask how actions conform to her ideals
Self - Reflective Thinking - thinking about ya thinking about inner world
Reflective Thinking - think about inner world
Deliberative Thinking - Planning or Reasoning
Learned Reaction - look left at street
Intitive Reactions - turn when hear noise

Words that are <u>Pormanteaux</u>
- can stuff a lot into the world
- aka <u>suitcases</u>
- ie love, creativity

- leads to multiplicity - lots of levels to think on
- if ya only have 1 way to solve problem ya're screwed - so always >1 way to think about things

(6)

What are all these things are about?

- Problem solving - Turing saw this as the big goal

Each of these is a crystalization of what we do all the time

<u>SOAR</u> - A cognitive theory equipped w/ a programming language

<u>Emotion Machine</u> - Philosopy
- its never been implemented

These things are still being worked on - but also new things

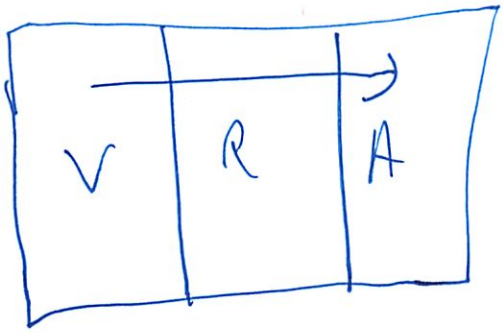
<u>Subsumption Architecture</u>

Rodney Brooks

interested in building creatures
- that can wonder arand and pick up tole cans

(7)

Before people wer stuck in



- but complex - adding 1 thing breaks another

So build layers

Seek	VRA
Explore	VRA
Wander	VRA
Avoid	VRA

- Once you get something working - don't touch it!
- layers interact w/ other layers

Features

1. Layers
2. No Representation
   World is the model
   highly reactive
   no map

(8)

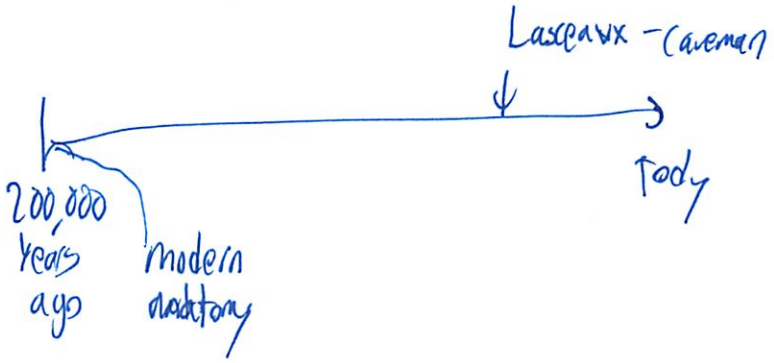
Old robots ~~small~~ slow

Just made movies
└ didn't take stats

But these advancements did not seem to make much ~~change~~ progress to make a "smart" robot

---

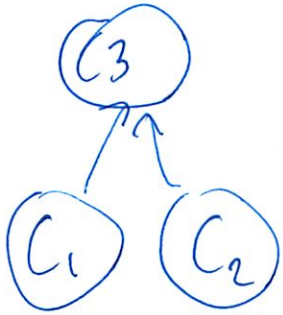
Timeline



were anatomically modern
but we were not smart

cave paintings + Jewelery - symbolic

Chomsky



combine concepts w/o distorting earlier concepts and w/o limits

Development Experiments

White Room

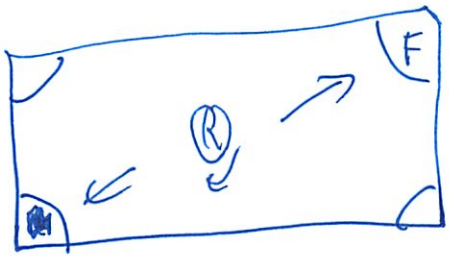
Rat placed in center on turntable

In each corner - uniform cloths

put food under one - have rat watch

Turn the turntable

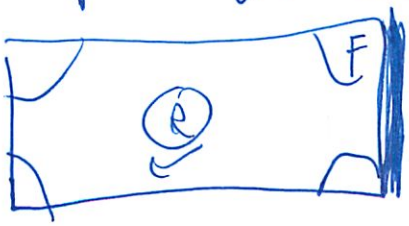
Rat will go either



Same w/ child + toy

Same w/ adult + food

But if paint one wall green



(10)

Rat + small child still go to either corner
└ neither is color blind

But adult gets it right every time

When does a child become an adult?
└ around 5 or 6
What does it correlate with?
└ When child uses words left + right in their own descriptions of world

---

~~Reading~~

~~Speed~~ Copying while reading - said I was pretty good
but can't do food thing
after a while can do it

But this use of langage that make it possible for adults to do things humans can't
Chimps squirm too much to do this

Key difference! Ability to combine concepts
humans over chimps! w/o limit

(11)

Wed: Business of AI
└ not on quiz

Post

All recognized short wall / long wall difference in corners

6.034 Tutorial

11/21

Quiz Dec 7 or So SVM + Boosting
Then after that Probabilistic systems
2 only on Final

Quiz 3 #3 Near Miss

Scope was supposed to include it
but everyone was unclear
I think they should provide a "kanban"
of topics on exam

Lots of people got it wrong
Even people who claimed they studyed it
("Much harder than examples we studied"

This is natural lang - so harder than blocks

---

(2)

Picking at ~~near~~ near-miss should be easy

<u>near miss</u> A miss involving 1 thing

wrong
different from model up to this point

If >1 thing don't know which one contributed to miss

1st one not ~~near~~ near miss
1st is murders and kills
2nd leans more stuff that would be revenge

Last one is near mis
Only difference is <u>leads</u> to capitalized

capitalized means <u>must</u> be present

(3)

Extending set - now have another set of possibilities

Want to make it so new rule is consistent w/ all previous rules
- also look ahead to next line what is already filled in

Problem 3s are also on finals
- Some TAs said to look on old finals problem 5s it focuses on stuff that has not been focused on before

Read Near Miss chap of book
- should have read

Are people doing worse this year?
- Winston trying to change
  - wants Lectures to matter

Plus were some bugs in qr
- What should be capitalized wrong in a lot of places

Lot of 0s

(4)

Some error in expectations

Lectures not clear about what it meant

This week's plan is SVM

instead think about 20 pt questions

Since can do 48 pt questions practice exams

and done in recitation + mega recitation

TAs don't really know what can be problems

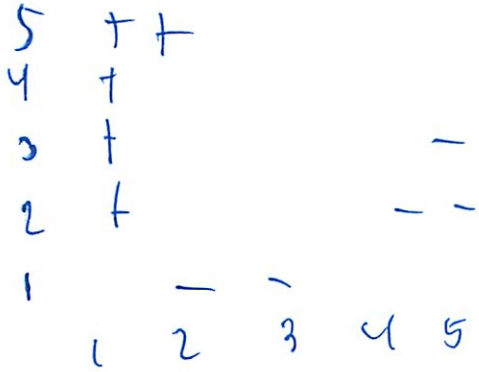
Where does problem in class can be easily extended

Could you get a problem from today?

Just multiple choice about subsumption

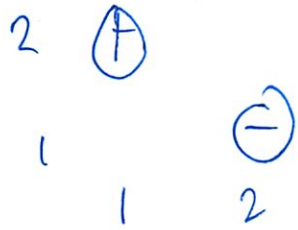
Doodle on possible ~~total~~ topics

SVMs

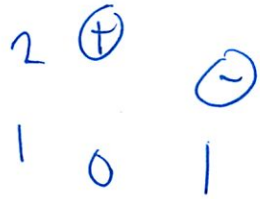


(5)

Circle support vectors



I #ed a bit differently. Their #s

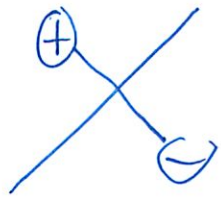


So what is classifier?

$y = x + 1$

So $y = mx + b$

line always between SV M



Write eqn of line

If you can remove point and line stays the same – then it's not a support vector (SV)!
Sometimes can choose one or other to remove

But need to turn line to classifier

$>$ or $<$

Write w/ one side $= 0$

$-x + y - 1 \quad 0$

blank

Then try $>$ or $<$

$-x + y - 1 > 0$

So $\oplus \geq 1$, which is what we defined, good!

Next choosing α

Equations
$$\begin{cases} y_i(\bar{w} \cdot x_i + b) = 1 \\ \bar{w} = \sum \alpha_i y_i \bar{x}_i \\ \sum \alpha_i y_i = 0 \end{cases}$$

← support vector 1 away from line

← comes from doing Lagrange w/ st. w

← Lagrange w/ st b

(8)

Our Lagrangian

$$L = \frac{1}{2}\|\bar{w}\|^2 - \sum \alpha_i [y_i(\bar{w} \cdot \bar{x}_i + b) - 1]$$

We only want support vectors

$\alpha = 0$ for Not SVs

$$\alpha_1 - \alpha_2 = 0$$

$$\boxed{\alpha_1 = \alpha_2}$$

$$\bar{w} = \alpha_1 \bar{x}_1 - \alpha_2 \bar{x}_2$$

$$x_1 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

$$x_2 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$\oplus = 1$

$\ominus = -1$

unless told otherwise

$$(\alpha_1 \bar{x}_1 - \alpha_1 \bar{x}_2)\, x_1 + b = 1$$

$$\alpha_1 \bar{x}_1 \bar{x}_1 - \alpha_1 \bar{x}_2 \bar{x}_1 + b = 1$$

(8)

$\alpha_1 4 - \alpha_1 2 + b = 1$

Now stuck

2 unknowns

$-1(\alpha_1 \bar{x}_1 \bar{x}_2 - \alpha_1 \bar{x}_2 \bar{x}_2 + b = 1$

$-\alpha_1 2 + \alpha_1 2 - b = 1$

Now 2 eq'n 2 un-knowns

$2\alpha_1 + = 2$

$2\alpha_1 = 2$

$\alpha_1 = 1$ ✓

Now can solve problem

Can keep going if want

↳ find b by plugging in α

6.034 Lecture

11/23

The AI Biz

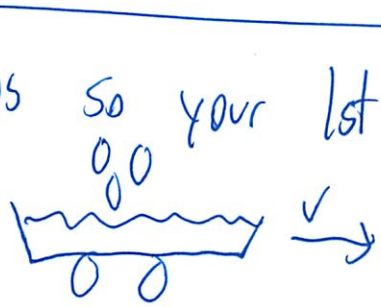
(optional)

- ☐ Three Powerful Ideas
- ☐ Three More Powerful Ideas

*
*
*

* Nobody gives a damn about saving $
* The superposition principle
* All big organizations are stupid

Tips so your 1st startup does not go broke



8.01 physics test

~~F=ma~~ wrong since mass is changing
(tend to ~~mem~~ remember what got wrong)

$2,000,000 economic impact of cos started by MIT alumni

(2)

Since MIT students rather win an argument than convince people

Often objective starting cos is to show people are wrong
└ then to make ~~$~~

* Olson founded DEC: Said his one goal as a company is to survive

* Fred Adler gave keynote at AI Biz conference:
I will never invest in ~~$~~AI cos
I will only invest in cos that solve peoples problems

* Erza Dyson was an IT pundit - very powerful:
AI would not amount to anything important unless embedded in main stream systems
He started a biz. He had no clue what to do.
1st contract: rewrite program in Lisp
Then someone they knew Contenental CEO
└ needed to know where to park airplanes at terminal

③

Built system for Continental

Gatekeeper software

Everyone loved it - but nobody bought it

They had 600 competitors - each airlines IT dept

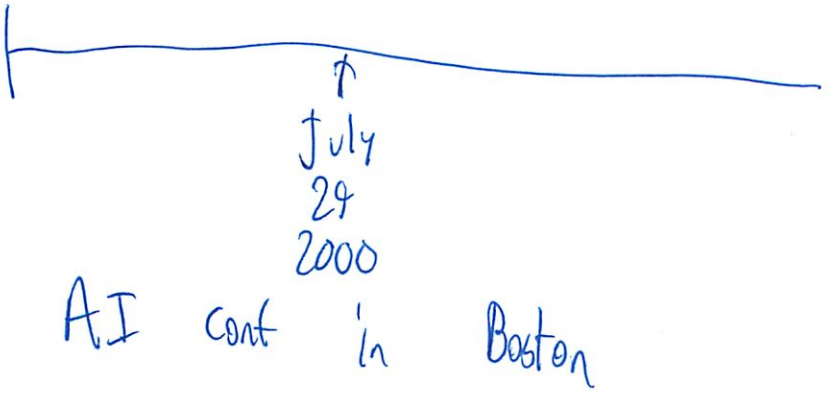
How many customers you have now? 0

This was 1989

He was at AI lab - head

DARPA PM now

They looked at his co



(4)

Aug 2
2000

Demonstration of Lotus - againts UI patents
- run by Richard Stallman
- no press
- same day Kuwait invaded

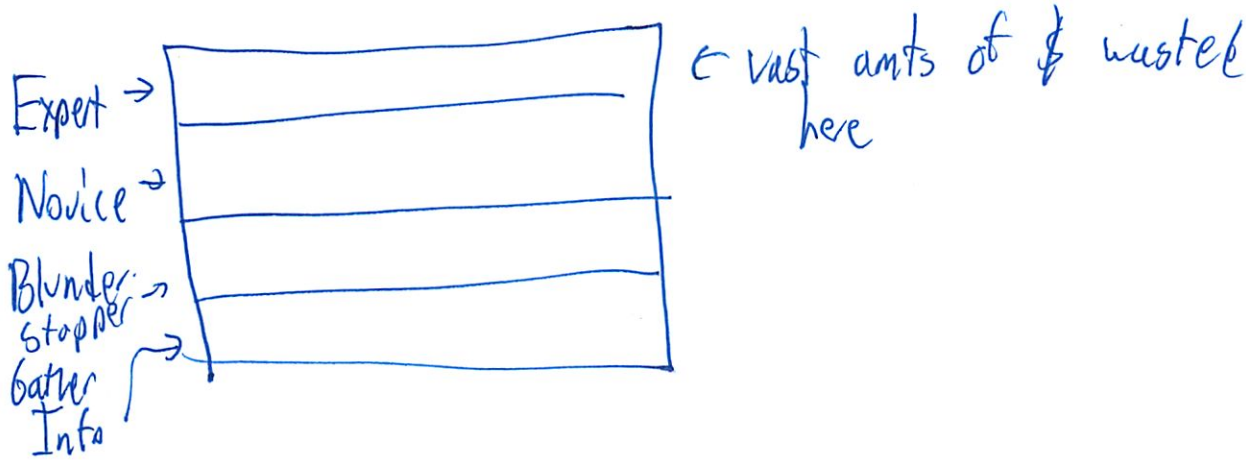
Aug 6
2000

Military guy called back
Wanted to make moving plans
→ Was used in Iraq invasion
DARPA guy said it justified every penny on AI
Now went back to Delta
Did a deal

(5)

Is the commerical ~~[illegible]~~ reason for AI to replace people?

- like experts



What else could it be?

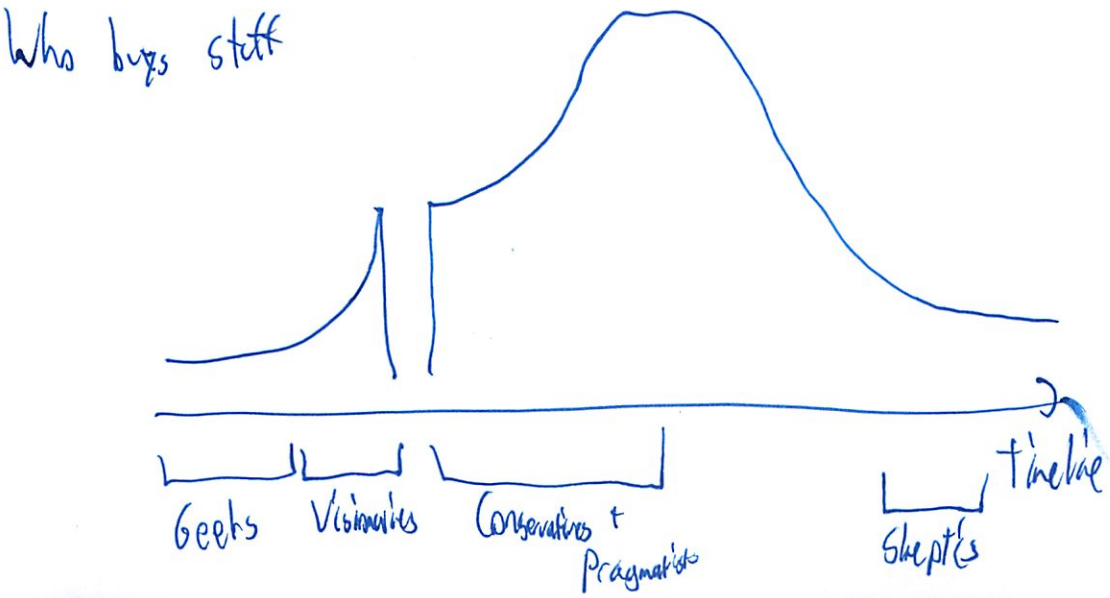
New Revenue

Lor for military i opportunity

Their system was in Blunder Stopper + Gater Info

Its Combo People + system

Who buys stuff



(6)

What are each groups interests

Visionaries - New Revenue

Skeptics - Save $

AI people were trying to sell to Skeptics/Save $ early on

Video from Boston Dynamics

- working accross grand
- surgery simulation
- DiGuy

- improving surgery - no new revenue
  - Surgons hate it: would be measured
  - put in last mtg before Thanksgiving

Cos not interested in papers

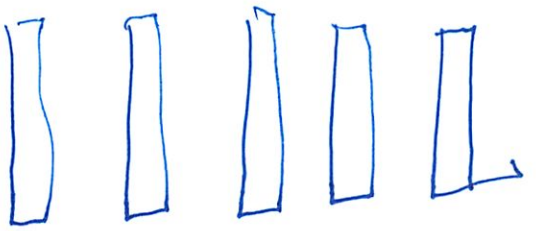
- interested who else uses
- Some things matter more then others

(7)

Big orgs behave stupidly

- even w/ smart people - like MIT
  - as an org do stupid things

Atlanta airport



took 3 months to buy the new software

The people were not at the top of their class at MIT

MITs come in w/ highest self esteem at Top 20 chs
leave w/ lowest self esteem

<u>Mega</u> ~~Only~~ You can do it
Only you can do it
You can't do it alone

- Collectively people are smarter

6.034 Tutorial 12/05

No tutorial last week

Coverage

SVMs

Boosting

Part 3: Representation, Architecture,
before T After T

Today's lecture: on Final, not this Quiz

(I really need to review)

Boosting - straightforward

SVMs - can have a few tricks

Boosting Q

how many errors

23 45

12

2 can be misclassified

Check w/ weights to see if actually misclassified

(2)

SVM

(handout)

Match kernals to graph

Can see demonstraitions online
- link was sent out
- under demonstrations section

key is reading directions

Only 5 distinct pictures

$(V1 \cdot V_2 + 1)$ linear → so E

$(V1 \cdot V_2 + 1)^2$ quadratic → C

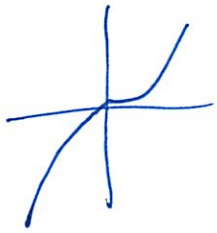
A, B, D look similar
- can't be polynomial

$(V_1 \cdot V_2 + 1)^3$ could be C - they can be the same

What happens when ↑ order - more degrees of freedom
- can get better - or can over fit

③

- 3rd order polynomials still can't make circles
- eq'n does not look like a 3rd order polynomial



- could get something that looks like C
- more complicated terms can always be simpler

- all of the polynomials can't be circles

So the best are C

Can get an intuition from doing the demonstrations

$$e^{\frac{-\|V_1 - V_2\|^2}{2\sigma}}$$

σ (gamma)

Remember this is about the distance b/w items

So how does this matter?

how much the fn cares about pts that are far away

make parallels to Gausian - mean

(e)

Small λ = small circle
large large

last one $\frac{-\|V_1 - V_2\|^2}{.06}$ = smallest circle - B

Then next one A

Remember C have to be the same
4 Cs - have already

C is so big - can't see ~~center~~ edges of circles
So last one is C

So largest two are λ

Note the 6 - are indistinguishable together
not that some combo of the 6 will be same in pairs, etc

(5)

Part C chandat

~~Not C, B, A, D~~

The linear one is excluded

Want one that are correct - but worst for future pts

(B) - since very close to training data

adding new points likly to be wrong

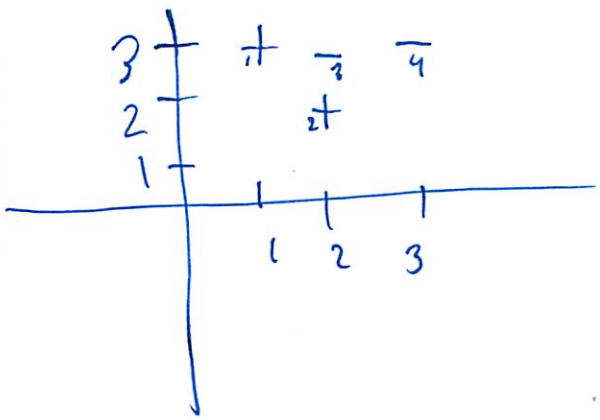
highest prob of all of the possibilites

can't definitely say overfitting unless try some testing data

---

Are there problems w/ >2 SVMs?

- 2006 Final

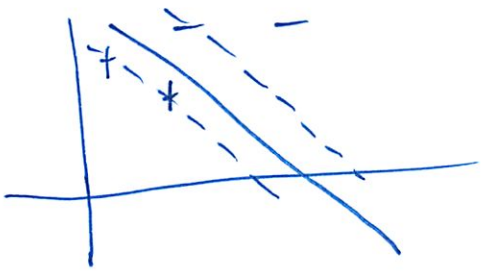


(6)

1. Sketch decision boundry

- must be a line (continuous)

  L unless very weird kernal



Now find

$w =$

$\alpha_1 =$

$\alpha_2 =$

$\alpha_3 =$

$\alpha_4 = 0$ ← since not a SVM

<u>Equations</u>

$$\sum \alpha_i y_i = 0$$

$$\sum \alpha_i y_i x_i = w$$

$$y_i(\bar{w}\,\bar{x}_i + b) = 1$$

$1 = \oplus$

$-1 = \ominus$

L if support vector

7

$$L = \frac{1}{2}\|w\|^2 - \sum \alpha_i \left(y_i\left[wx_i + b\right] - 1\right)$$

↑ magnitude
$\left(\sqrt{x^2+y^2}\right)$

Now do

$$\alpha_1 + \alpha_2 - \alpha_3 = 0$$

↳ not as nice as before - but ok

$$\alpha_1 x_1 + \alpha_2 x_2 - \alpha_3 x_3 = w$$

↑ knowns
vectors
of the points
(just their coords)

Sub ~~[illegible]~~ in for α_3

$$\alpha_1 x_1 + \alpha_2 x_2 - \alpha_1 x_3 - \alpha_2 x_3 = w$$

$$\alpha_1 (x_1 - x_3) + \alpha_2 (x_2 - x_3) = w$$

~~[illegible]~~ have 2 unknowns

Now

$$w^{\cdot} x_1 + b = 1$$

$$w^{\cdot} x_2 + b = 1$$

$$w^{\cdot} x_3 + b = -1$$

(8)

Now plug in one for the other

$$\alpha_1(x_1 x_1 - x_1 x_3) + \alpha_2(x_1 x_2 - x_1 x_3) + b = 1$$

$$\alpha_1(x_2 x_1 - x_2 x_3) + \alpha_2(x_2 x_2 - x_2 x_3) + b = 1$$

$$\alpha_1(x_3 x_1 - x_3 x_3) + \alpha_2(x_3 x_2 - x_3 x_3) + b = -1$$

Now solve for

3 unknows $\alpha_1, \alpha_2, \alpha_3$

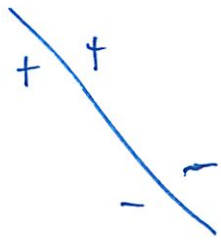
Solve middle parts

Easier if 2

Does $\alpha_1 = \alpha_2$?

- αs define the margin
- if 2 support vectors at opposite sides will be same
  - ↳ at least 2 will be there usually

<u>Not</u>

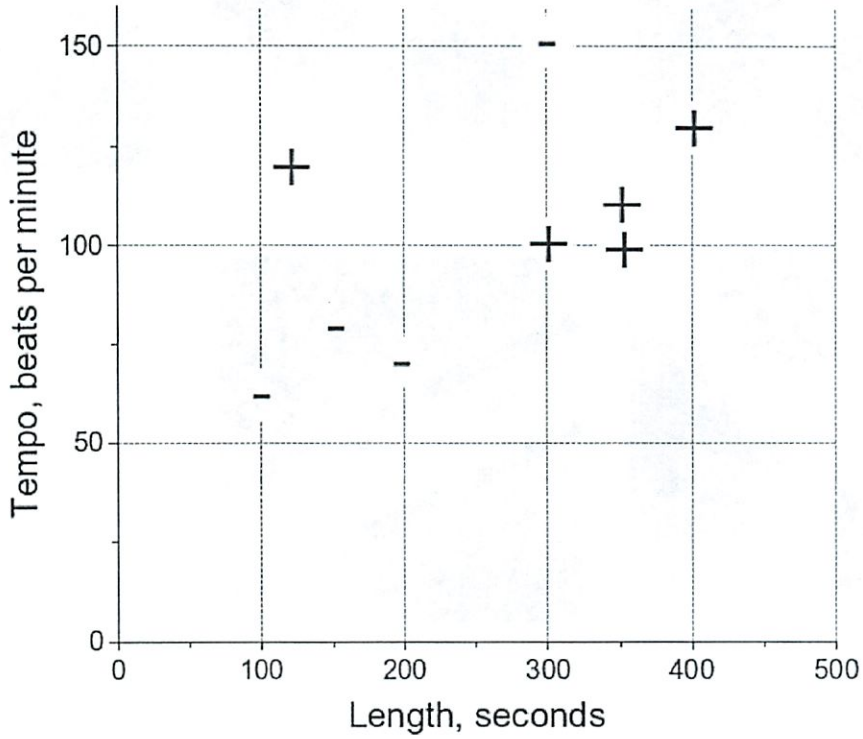


Tutorial Handout

12/5

Part B (15 points)

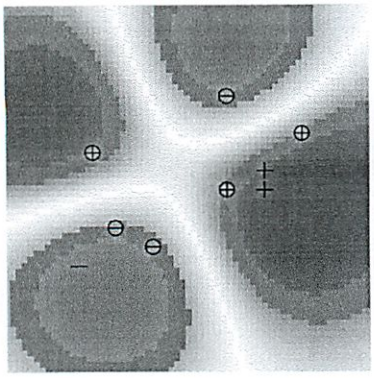
You decide to try an alternative approach with the following data:



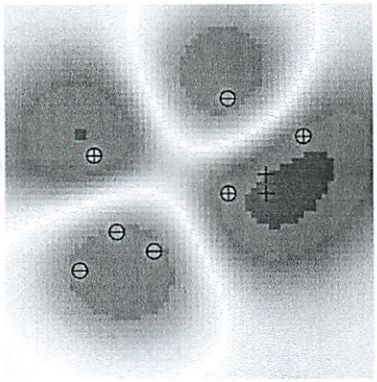
indistinguishable together

To get a better understanding of how various kernels handle your new approach, you try all ten provided in the demonstration program. You discover, interestingly, that **6 of the 10 produce results that are just about indistinguishable, so only five different pictures are shown in the accompanying diagram.** Fill in the table below with the appropriate diagram label from the separate sheet containing SVM outputs. **Note that 6 of your answers in the 10 boxes will be the same and 4 will be different from all the others.**

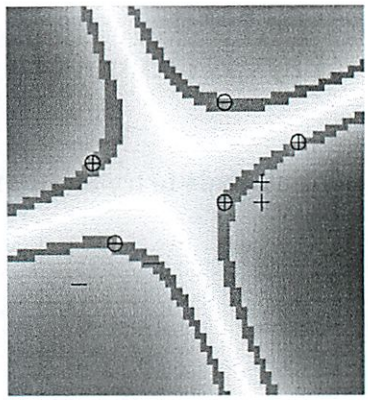
$(V1 \cdot V2+1)$	E	$e^{\frac{-\|V1-V2\|^2}{2.0}}$	C
$(V1 \cdot V2+1)^2$	C	$e^{\frac{-\|V1-V2\|^2}{0.5}}$	C
$(V1 \cdot V2+1)^3$	C	$e^{\frac{-\|V1-V2\|^2}{0.22}}$	A
$(V1 \cdot V2+1)^4$	C	$e^{\frac{-\|V1-V2\|^2}{0.12}}$	D
$(V1 \cdot V2+1)^5$	C	$e^{\frac{-\|V1-V2\|^2}{0.06}}$	B



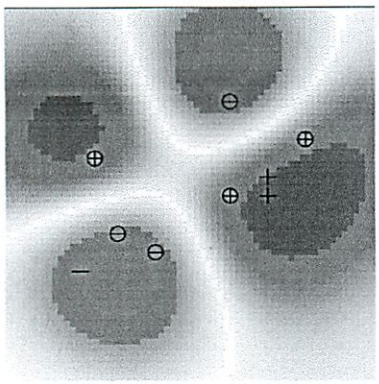
A



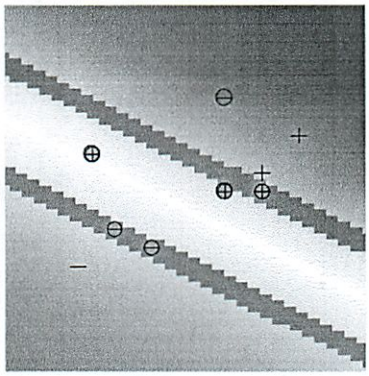
B



C



D



E

Part C (5 points)

Of the kernels tried in part B, one does not succeed in properly classifying all the examples. Among the rest, identify the kernel that is likely to provide the worst results when applied to test data:

Why?

Reference material and playlist

From 6.034 Fall 2011

Most of the readings come from Patrick Winston's AI textbook (third edition), which exists as a physical book (http://www.amazon.com/Artificial-Intelligence-3rd-Winston/dp/0201533774/) , but is also available on the internet (http://courses.csail.mit.edu/6.034f/ai3/) (and there's a table of contents here (http://people.csail.mit.edu/phw/Books/AITABLE.HTML)).

Topics and Playlist 2011

September	Day	Topic	Quiz #	Playlist
7	Wed	What it's all about	1	This could be the last time, Stones
12	Mon	Goal trees and symbolic integration (http://courses.csail.mit.edu /6.034f/ai3/saint.pdf)	1	You can get it if you really want it, Jimmy Cliff
14	Wed	Goals and rule-based systems (pp.53-60) (http://courses.csail.mit.edu/6.034f/ai3/ch3.pdf)	1	Engineer's Song, Chorallaries
19	Mon	Basic search (http://courses.csail.mit.edu/6.034f/ai3/ch4.pdf)	1	Searchin', Stones
23	Fri	Optimal search (http://courses.csail.mit.edu/6.034f/ai3/ch5.pdf)	1	Route 66, Stones
26	Mon	Games (http://courses.csail.mit.edu/6.034f/ai3/ch6.pdf)	2	It's Only Rock and Roll, Stones
28	Wed	Quiz 1	-	-
October	**Day**	**Topic**	**Quiz #**	**Playlist**
3	Wed	Constraints in drawings (http://courses.csail.mit.edu/6.034f /ai3/ch12.pdf)	2	I Can't Get No Satisfaction, Stones
5	Wed	Constraints in maps and resource allocation	2	Paint it Black, Stones
12	Wed	Constraints in object recognition (http://courses.csail.mit.edu/6.034f /ai3/ch26.pdf)	2	The First Time I Saw your Face, Presley
14	Fri	Nearest neighbor learning (http://courses.csail.mit.edu/6.034f /ai3/ch19.pdf) /Sleep (http://courses.csail.mit.edu/6.034f/sleep.pdf)	3	ABC song, Ray Charles et al.
17	Mon	Identification tree learning (http://courses.csail.mit.edu/6.034f /ai3/ch21.pdf)	3	Romanian national anthem, Desteapta-te române!
19	Wed	Neural net learning (http://courses.csail.mit.edu/6.034f /ai3/netmath.pdf)	3	19th Nervous Breakdown, Stones

24	Mon	Genetic algorithms (http://courses.csail.mit.edu/6.034f/ai3/ch25.pdf)	3	Let's spend the night together, Stones
26	Wed	Quiz 2	-	-
31	Mon	Learning in sparse spaces (http://courses.csail.mit.edu/6.803/pdf/yip.pdf)	3	You talk too much, Peas
November	**Day**	**Topic**	**Quiz #**	**Playlist**
2	Wed	Support-vector machines (http://courses.csail.mit.edu/6.034f/ai3/SVM.pdf) , SVM (and Boosting) Notes (http://ai6034.mit.edu/fall11/images/SVM_and_Boosting.pdf)	4	Get a little help from my friends, Beatles
7	Mon	Learning from near misses (http://courses.csail.mit.edu/6.034f/ai3/ch16.pdf)	3	Imma Be Rocking that Body, Peas
9	Wed	Boosting (Winston and Ortiz notes) (http://courses.csail.mit.edu/6.034f/ai3/boosting.pdf) , Boosting (Shapiri paper) (http://courses.csail.mit.edu/6.034f/ai3/msri.pdf)	4	Workin' together, Ike and Tina Turner
14	Mon	Frames and representation (http://courses.csail.mit.edu/6.034f/ai3/ch9.pdf)	4	Selections from the Black Watch, aka The Ladies from Hell
16	Wed	Quiz 3	-	-
21	Mon	GPS, SOAR (http://courses.csail.mit.edu/6.034f/ai3/SOAR.pdf) , Subsumption (http://courses.csail.mit.edu/6.034f/ai3/Subsumption.pdf) , Society of Mind (http://web.media.mit.edu/~minsky/eb5.html) , Genesis (http://courses.csail.mit.edu/6.034f/ai3/Submitted.pdf)	4	Thus spake Zarathustra, Strauss
23	Wed	The AI Business	-	Money, money, ABBA
28	Mon	Probabilistic inference I (http://courses.csail.mit.edu/6.034f/ai3/bayes.pdf)	5	Oh No, Not You Again, Stones
30	Wed	Probabilistic inference II (http://courses.csail.mit.edu/6.034f/ai3/bayes.pdf)	5	Tumbling Dice, Stones
December	**Day**	**Topic**	**Quiz #**	**Playlist**
5	Mon	Watching the brain at work, less than you want to know (http://courses.csail.mit.edu/6.034f/ai3/Kanwisher2010.pdf) Watching the brain at work, more than you want to know (http://web.mit.edu/bcs/nklab/publications.shtml)	5	Happy, Stones
7	Wed	Quiz 4	-	-

6.034 Exam 4 Prep

12/6

Coverage (from web)

- SVM
- Boosting
- Frames + Representation
- GPS, Soar, Subsumption, Society of the mind, Genesis

Oh you can run his software - nice!

Lecture 11/2 SVMs

Look for widest "street"

Points beyond street don't matter

~~Other~~ Points on street = "support vectors"

$\vec{w} \cdot \vec{v} + b > 0$ for decision rule

$\|\vec{w}\| \|\vec{v}\| \cos\theta$

$\sqrt{x^2+y^2}$

angle b/w

②

Constraints

$$\bar{w} \cdot \bar{x}_+ + b \geq 1$$

$$\bar{w} \cdot \bar{x}_- + b \leq 1$$

New variable

$$y = \begin{cases} 1 & \oplus \\ -1 & \ominus \end{cases}$$

Multiply constraints by y

$$y_+ (\bar{w} \cdot \bar{x}_+ + b) \geq 1$$

$$y_- (\bar{w} \cdot \bar{x}_- + b) \geq 1$$ ← correct

$$\boxed{y_i (\bar{w} \cdot \bar{x}_i + b) - 1 \geq 0}$$ important

↑ if at side street will get 1 or −1
if just inside street will get < 1

Figure out how wide street is

$$(\bar{x}_+ - \bar{x}_-) \cdot \frac{\bar{w}}{\|\bar{w}\|} = \text{width}$$

(3)

So want to maximize $\frac{2}{||\bar{w}||}$

Can invert $\frac{||\bar{w}||}{2}$ to minimize

Set up Lagrangian

$$L = \frac{1}{2}||\bar{w}||^2 - \sum_i \alpha_i \left[y_i (\bar{w} \cdot \bar{x}_i + b) - 1 \right]$$

WP! Lagrangian (L) function that summarizes the dynamics of a system

$L = T - V$

T: kinetic energy, V: potential energy

important in mechanics

Can easily read conservation etc

(I don't get it...)

Think saw in 8.02 - and I was reading wrong WP article

its λ thing - for find min + max

(4)

Anyway

$$\frac{\partial L}{\partial \bar{w}} = \bar{w} - \sum_i \alpha_i y_i \bar{x}_i = 0$$

? differentiating vector

(This stuff does not really matter - Burwick's notes good)

$$\boxed{\bar{w} = \sum_i \alpha_i y_i \bar{x}_i}$$

Now differentiate w/ respect to b

$$\frac{\partial L}{\partial b} = -\sum \alpha_i y_i = 0$$

Can plug $\bar{w}$ back into Lagrangian

$$L = \frac{1}{2}\left(\sum_i \alpha_i y_i x_i\right)\left(\sum_j \alpha_j y_j x_j\right) - \sum_i \alpha_i y_i x_i \sum \alpha_j y_j x_j - b \sum \alpha_i y_i + \sum \alpha_i$$

← weird ...?

5

$$= \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i x_j -$$

$$- \sum_i \sum_j \alpha_i \alpha_j y_i y_j \bar{x}_i \bar{x}_j + \sum_i \alpha_i$$

Can rewrite decision rule

$$\sum \alpha_i y \bar{x}_i \bar{v} + b \geq 0$$

Can seperate some things in 3D

Swap in kernal functions

$$\bar{z} = \phi(\bar{x})$$

$$\phi[\bar{x}_i] \cdot \phi(\bar{x}_j) = k(\bar{x}_i, \bar{x}_j)$$

$$\phi[\bar{x}_i] \cdot \phi(\bar{u}) = k(\bar{x}_i, \bar{v})$$

Don't need ϕ, just need k

↑ kernal function

$$k = e^{\frac{\|x_i - x_j\|^2}{\sigma}}$$

↑ constant – the distance each sample has influenced

confused what influence is if others are 0

(6)

Mega Recitation

Hardest topic in class

* Avoid $\frac{1}{||w||} = d$

* k is new dot product if ϕ is applied to all vectors

Represent each pt w/ α score

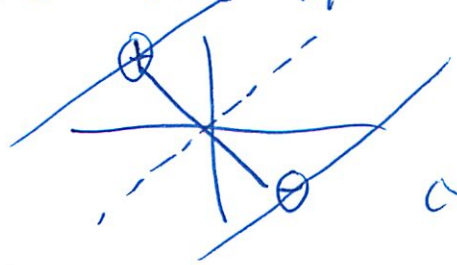
non SV = 0 = not related at all

α = how much a point is constraining the width of the road

high = very constrained

low = not constrained

Draw line b/w support vectors



linear ~~kern~~ kernal

(7)

Dotted line divides new points

A few eq'ns to solve for

$W_1 x + W_2 y + b = 0$ for dotted line

x, y are any pt on dotted line

eq'n for line

Quiz often asks to solve for w, b

$W_1 x + W_2 y + b = 1$ for ⊕ solid line SVM

$= -1$ for ⊖ " " "

Can plug in x, y + solve

$\sum \alpha_{\oplus} = \sum \alpha_{\ominus}$

$\bar{w} = \sum \alpha_{\oplus} \bar{x}_+ - \sum \alpha_- \bar{x}_-$

can solve for weights like this

Usually no more than 2 pts

8

<u>Shortcut</u> d is half of perpendicular bisector

$$d = \frac{1}{\|w\|}$$

midpoint

$$W_1 X_1 + W_2 Y_1 + b = 0$$

$$W_1 X_2 + W_2 Y_2 + b = 0$$

⊕ SVM

~~Multiples~~ Multiply vectors

$$\bar{W} \cdot \bar{X}_1$$

$$\bar{W} \cdot \underbrace{(\bar{X}_1 - \bar{X}_2)}_{P} = -1$$

$$(\bar{X}_1 - \bar{X}_2) = \frac{-1}{\|w\|}$$

$$d = \frac{1}{\|w\|}$$

9

Dotted line $y = x - 1$

Infinite # of w, b - lots are multiples of 1 another

(Confused - do Recitation 1st)

Recitation 11/10 SVM

+/- Classifier

tries to fit as many points as possible

so can seperate regions as much as possible

w/o compromising split

Maximize $\frac{2}{\|w\|}$

$$\|w\| = \sqrt{\sum_{i=1}^{n} w_i^2}$$

Such that $y_i(w_i x_i + b) \geq 1$

$$\frac{\partial L}{\partial b} =$$

$$\frac{\partial L}{\partial w} =$$

(10)

(or perhaps I remember boasting more)

(Read handout)

avoids overfitting
train on points
then test it w/ seperate points

width of road $= m = \frac{2}{\|\bar{w}\|}$

$\sqrt[n]{\sum_i w_i^2}$ ie the lenght of w

$\sqrt{x^2+y^2}$ for y

Want to maximize street width
and minimize $\|\bar{w}\|$

Subject to contraint that we classify pts as ⊕ ⊖

$$Y_i(\bar{w} \cdot \vec{x}_i + b) \geq 1$$

dot product

So this is our modifier

$Y_i \oplus = 1$

$Y_i \ominus = -1$

(11)

~~Resah~~ Resulting Lagrange multiplier

$$L = \frac{1}{2}\|\bar{w}\|^2 - \sum_i \alpha_i (y_i (\bar{w} \cdot \bar{x}_i) + b) - 1)$$

Then take partials w/r to b, w

Then solve for w, b, α_i

to get max road width m

α = amt of influence

└ the closer to opposite, the narrower the street, the higher the α

<u>kernal</u>

here dot product $w\ x$

└ distance b/w (since vector)

But can just read distance on graph...

ϕ = transform

$k(\bar{u}, \bar{v}) = \phi(u) \cdot \phi(v)$

↑ two vectors

(12)

Useful equations

A_i eqn from partials

1. Partial of Lagrangian w/s to b

$$\frac{\partial L}{\partial b} = 0$$

$$\boxed{\sum_i \alpha_i y_i = 0}$$

2. $$\frac{\partial L}{\partial W} = 0$$

Case 1: linear kernels

$$\boxed{\sum_i \alpha_i y_i \vec{x}_i = W}$$

Case 2: transformations

$$\sum_i \alpha_i y_i \phi(x_i) = \vec{w}$$

B_i eqns from +/- boundries + constraints

3. Decision boundry (to make new pts ⊕ or ⊖)

Case 1 linear $k(\bar{x}_i, x) = \bar{x}_i \cdot x$

$$\boxed{h(x) = \sum_i \alpha_i y_i \cdot \vec{x}_i \cdot \vec{x} + b \geq 0}$$

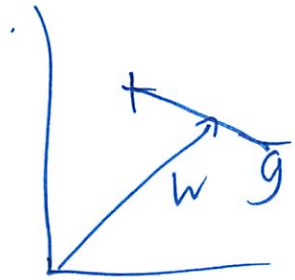
$$= \vec{w} \cdot \vec{x} + b \geq 0$$

(13)

What is w again

A vector

The normal to the g line connecting the 2 sums (I believe)



Case 2 General kernel

$$h(x) = \sum_i \alpha_i y_i k(\vec{x}_i, \vec{x}) + b \geq 0$$

4. The positive gutter

Case 1 Linear kernel

$$h(\vec{x}) = \sum_i (\alpha_i y_i \vec{x}_i \cdot x] + b = 1$$

$$= \vec{w} \cdot \vec{x} + b = 1$$

Case 2 Generic kernel

$$h(\vec{x}) = \sum_i \alpha_i y_i k(\vec{x}_i, x) + b = 1$$

(14)

5. Netive Gutter

$$h(\vec{x}) = \sum_i \alpha_i y_i \vec{x}_i \cdot x + b = -1$$

$$= \vec{w} \cdot \vec{x} + b = -1$$

etc

6. Width of Road

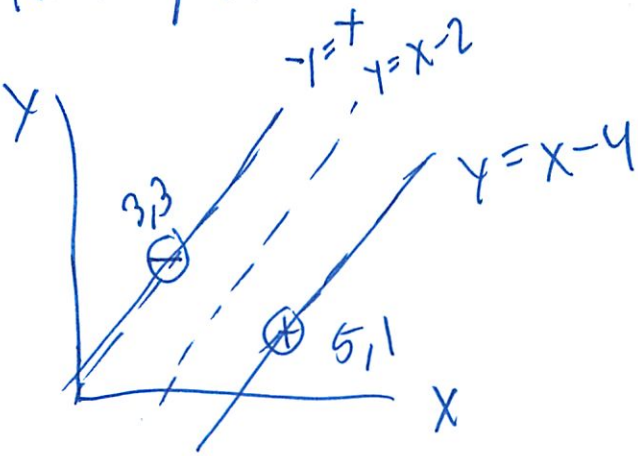
$$m = \frac{2}{\|\vec{w}\|} \quad \|\vec{w}\| = \sqrt{\sum_i w_i^2}$$

if just 2 svms

$$m = \frac{2}{\|\vec{w}\|}(x_+ - x_-)$$

Ok I think I have a good overview of can
Now how to actually do a problem

<u>Question</u> from packet



15

Now manipulate line into st form

$h(x) = W_1 x + W_2 y + b \geq 0$

1. $y \leq x - 2$
2. $x - y - 2 \geq 0$

Ok spend a minute here

So everything less than $x - 2$ is ⊕

Since ⊕ points are below line

So if eq is true, pt is ⊕

Then convert

$$x - 2 - y \geq 0$$

$$x - y - 2 \geq 0$$

Then fill in

$W_1 = 1$

$W_2 = -1$

$b = -2$

(16)

try plugin in

$$\frac{2}{\sqrt{1^2+(-1)^2}} = \frac{2}{\sqrt{2}}$$

(x) No not right

¿ Don't get what they mean by street width ?

¿ from center to edge ?

3. So realize any multiple of c is still on eq

$$cx - cy - 2c \geq 0$$

Then solve w/ other constraints

└ Set = to street width

$$\frac{2}{\sqrt{(c)^2+(-c)^2}} = \frac{2}{\sqrt{2c^2}} = 2\sqrt{2}$$

Street width from visual inspection

$$\frac{4}{2c^2} = 8$$

$$c = \frac{1}{2}$$

(17)

Now can read off values

$$cx - cy - 2c \geq 0$$

$$\tfrac{1}{2}x - \tfrac{1}{2}y - 1 \geq 0$$

So $\vec{w} = \begin{bmatrix} +1/2 \\ -1/2 \end{bmatrix}$ $b = -1$

So $\vec{w} = \begin{bmatrix} cx \\ cy \end{bmatrix}$

Then need to compute α for each training pt

$$\sum_i \alpha_i y_i \vec{x}_i = w$$

So Fill in 0 for all non-support vectors

So

$$\alpha_3 \overset{\downarrow \text{since } 0}{(-1)} \underset{\begin{bmatrix} x \\ y \end{bmatrix}}{\begin{bmatrix} 3 \\ 3 \end{bmatrix}} + \alpha_4 (1) \begin{bmatrix} 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/2 \\ -1/2 \end{bmatrix}$$

I learned this in 18.03!

$$-3\alpha_3 + 5\alpha_4 = 1/2$$

$$-3\alpha_3 + \alpha_4 = -1/2$$

2 eq, 2 unknowns Find α_3, α_4

So if add new pt
α = usually 0 - since not SVM

Move support vector

- α changes inversly to road width m

Diff kernals

1. Linear

   Remember Perceptrons

2. Polynomial

   $(\vec{u} \cdot \vec{v} + b)^n \quad n > 1$

   - parapolic, linear, or hyperbolic
   - not circles!

   what does this mean?

3. Radial / Gaussian
   - Contour circles around ⊕ ⊖ points
   - when σ^2 is large get wider Gaussians

Can combine several Gaussians ~~both~~ to perfect fit
but will overfit

4. <u>Sigmodial (tan h) kernal</u>

Allows for combos of linear decision boundries
L like neural nets 2nd neuron

$$k = \tanh(k\vec{u}\cdot\vec{v} + b)$$

$$= \frac{e^{k\vec{u}\cdot\vec{v}+b} + 1}{e^{k\vec{u}\cdot\vec{v}+b} - 1}$$

5. <u>Linear Combo of kernals</u>

"are closed"

(Ok I think I got this...)
(Is a section of gory details...)
Oh just same a lecture
I hope don't need to actually do Lagrange multipliers...

(2)

Boosting

Goal is to find a weighted combo of weak classifiers (that underfit data + make mistakes) into a single strong classifier $H(x)$

$$H(x) = \text{sign ex}\left(\alpha_1 h_1(\vec{x}) + \alpha_2 h_2(\vec{x}) + \dots\right)$$

$$= \text{Sinex}\left(\sum_{i=1}^{S} a_i h_i(\vec{x})\right)$$

where $H(\vec{x}) = \{-1, +1\}$

$h(\vec{x}) = \{-1, +1\}$

$$\text{Sigrex} = \begin{cases} \geq 0 & = +1 \\ < 0 & = -1 \end{cases}$$

W_i = weights - like probabilities

$$\sum_i W_i = 1$$

(21)

Pick the single best (lowest error rate) classifier h

Then boost weights of next datapoints that classifier misses - so next classifier does not make same mistake on

E^s is sum of all weights classifier s got wrong

$(1-E^s)$ is all correct

$E^s < \frac{1}{2}$ or else it would be otherwise way around

α_s is defined as $\frac{1}{2} \ln\left(\frac{(1-E^s)}{E^s}\right)$

|| weights

(22)

1. Set all datapoints $W_i = \frac{1}{n}$

2. Iterate over all classifiers

   Add the ones it gets wrong
   weights

   b. Compute

   $$\alpha_s = \frac{1}{2} \ln \left(\frac{1-E^s}{E^s} \right)$$

   c. Update weights

   For correct

   $$W_i^{s+1} = \left[\frac{1}{2} \frac{1}{1-E^s} \right] W_i^s$$

   incorrect

   $$W_i^{s+1} = \left[\frac{1}{2} \cdot \frac{1}{E^s} \right] W_i^s$$

   Review exactly what that means

3. Termination

If $S > T$ or $H(x)$ has error 0

Exit

(usually they only tell you x rounds)

4. Add up final classifier

Back to Lecture: Boosting

Adding together weak classifiers

Not that elegant

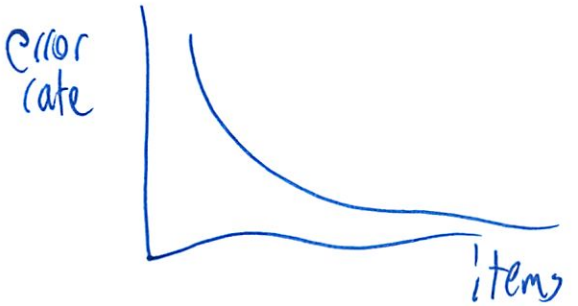
Z is normalizer - so it all adds to 1

$$Z^t = \underbrace{\sqrt{\frac{1-\epsilon^t}{\epsilon^t}} \sum_i w_i^t}_{\text{Misclassified}} + \underbrace{\sqrt{\frac{\epsilon^t}{1-\epsilon^t}} \sum_i w_i^t}_{\text{Classified correct}}$$

$$= 2\sqrt{\epsilon^t}\sqrt{1-\epsilon^t}$$

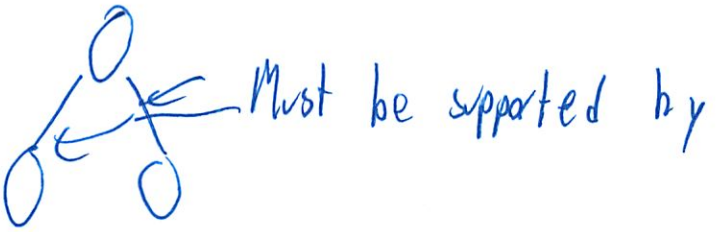
(this was already factored in above)

This seems not to overfit



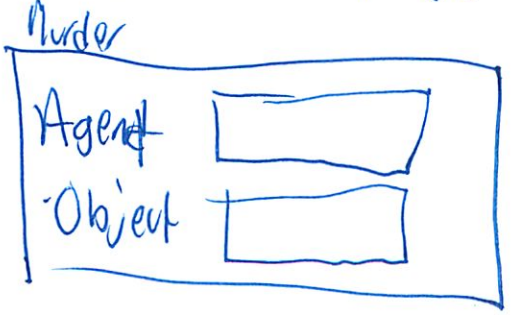
11/14 Lecture Frames + Representations

Semantic Nets



take something abstract + make it real

Can change frame we are looking at for knowledge



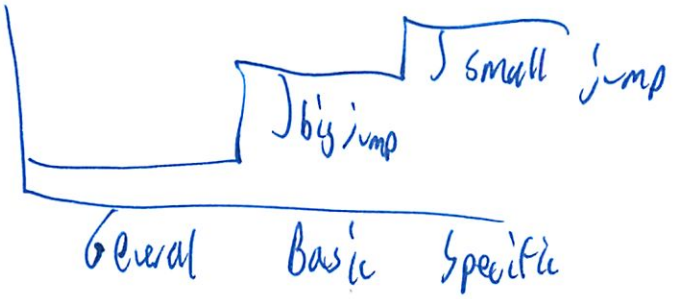
Can also sequence

$$O \rightarrow O \rightarrow O \rightarrow O$$

Classification, Transition, Trajectory

1. Classification

if you know someones class
You know a lot about someone something



Musical instrument
↑
Piano
↑
Bosendopler

(26)

2. Transition

		opposits
increasing	↑	~~↑~~
decreasing	↓	~~↓~~
changing	Δ	~~Δ~~
appearing	A	~~A~~
disappearing	D	~~D~~

Break into time slots

	slowing down	hitting wall	sitting there crashed
Distance b/w car + wall			
Speed of car L			
Condition of car			

(27)

3. Trajectory

bird flu increase

in English the prepositions

to, from, with, by, with, for, by

25% of sentances in WSJ have trajectories

Pat comforted Chris

Trajectory/Role Frame

Action;
Agent
Object
Result

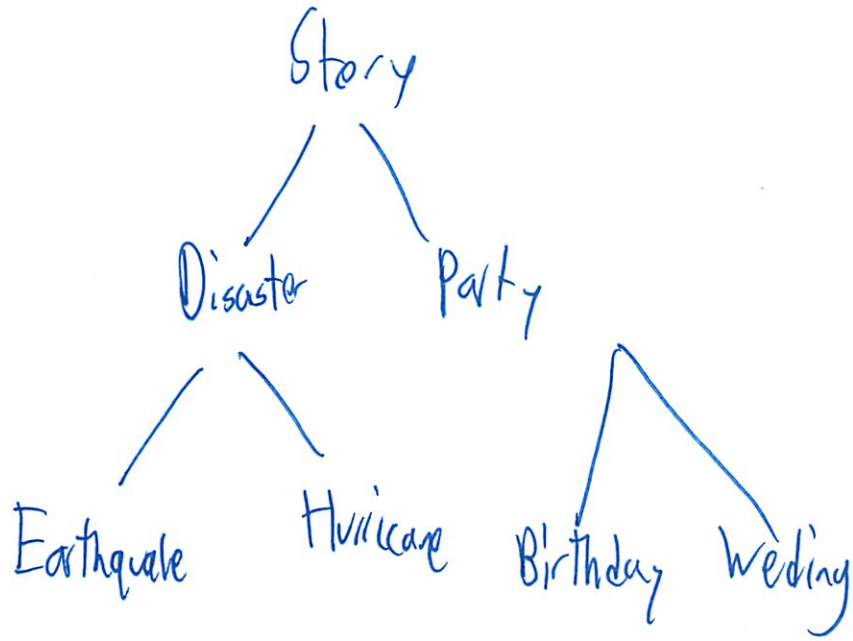
Transition

Chris Mood ↑

(28)

More complex

Can talk about frame of stories



SVM Key Eqns

1. $\sum \alpha_i y_i = 0$
2. $y = \sum_i \alpha_i y_i \cdot k(\bar{x}_i, \bar{x}) + b$

Boosting Mega Recitation

- Vampires example

1. Write down all the ones illegally classified
   - there can throw out the duplicates

(2 ϕ)

2. Assign $\frac{1}{N}$ to all W_i's

3. Look at classifyer w/ smallest error w/ weight ~~[illegible]~~.

4. Find α

5. His strategy (where denom all same)

1. erase denoms
2. Add up all the ones wrong
3. Multiply by 2
4. Do for ones right
5. Change for common denom

So all $\frac{1}{10}$

2 wrong

So $2 \cdot 2 = 4$

8 right $8 \cdot 2 = 16$

So $\frac{4}{16}$ $\frac{1}{16}$

(I kinda get that - can use real way)

6. Repeat

11/21 Lecture GPS, SOAR, Subsumption, Society of Mind, Genesis

How do you figure this out?
How do you put it together?

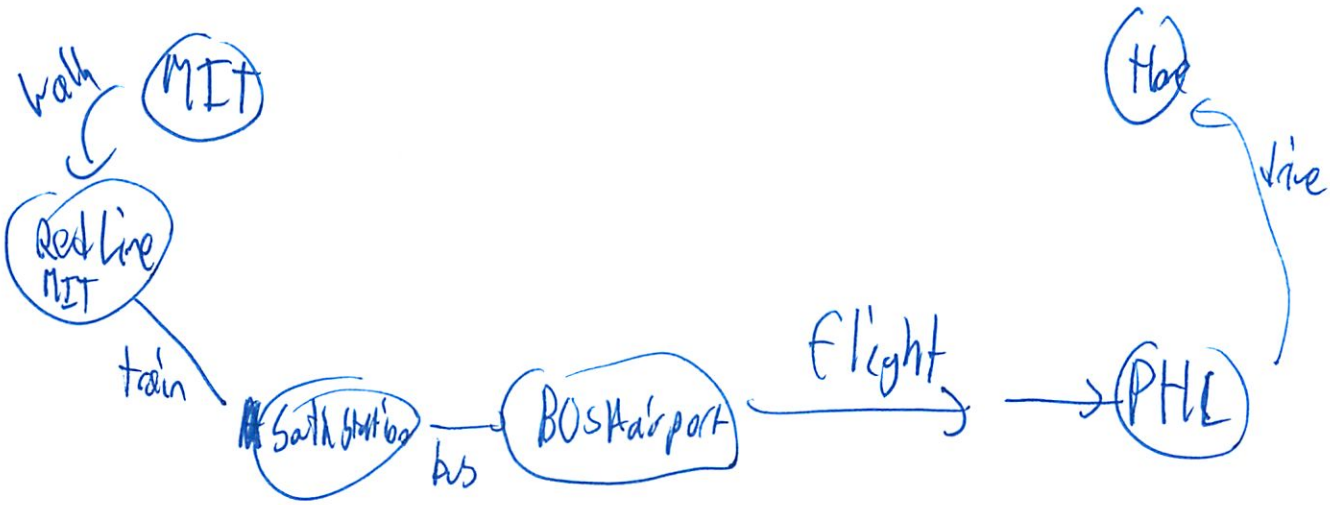
General Problem Solver

State

Goal

What to ↓ distance b/w here + goal

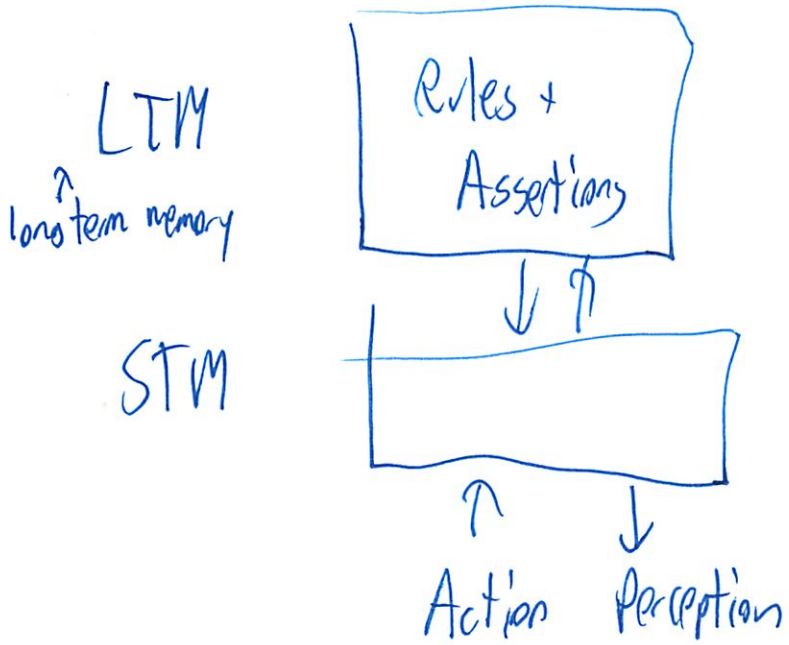
Like to home



SOAR

L New architecture

State, Operator And Result



Convinced humans are symbol + hypothesis processors.

System has

1. STMs + LTMs
2. Rules + Assertions
3. Preferences
4. Problem Space
5. Sup Goals
6. Chunking

(a collection of things done already)

Emotion Machine

- philosophy based system

Self-Conscious Emotions
Self-Reflective Thinking
Reflective Thinking
Deliberative Thinking
Learned Reaction
Intuitive Reactions

Subsumption Architecture

- building system to pick up coke cans

Not

V	R	A

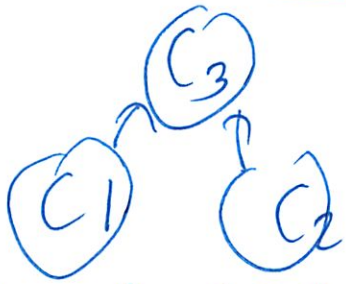
But

Seek	VRA
Explore	VRA
Wander	VRA
Avoid	VRA

No representation – world is the model

Chomsky

intellegence is



Combining concepts w/o disturbing original earlier concepts w/o limit

---

Development Experiments

White Room
Rat on turn table
Food under cloth
For children painting wall green does not matter

6.034 Quiz 4
December 2, 2009

Name	
email	

Circle your TA and recitation time **(for 1 point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

TAs
Erica Cooper
Matthew Peairs
Lisa Fisher
Mark Seifter
Yuan K. Shen
Jeremy Smith
Olga Wichrowska

Thu	
Time	Instructor
12-1	Gregory Marton
1-2	Berwick/Marton
2-3	Berwick/Marton
3-4	Berwick/Marton

Fri	
Time	Instructor
1-2	Randall Davis
2-3	Randall Davis
3-4	Randall Davis

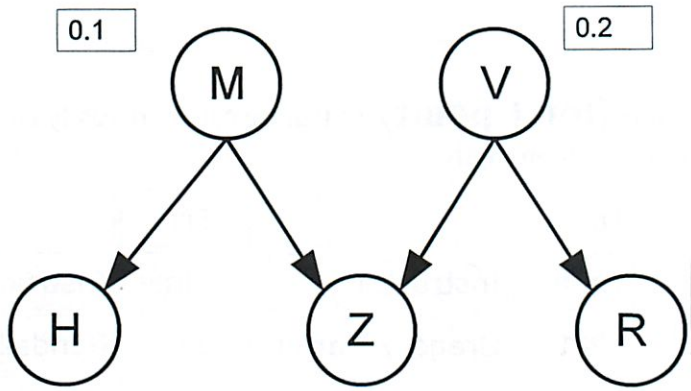
Problem number	Maximum	Score	Grader
1	49		
2	50		
Total	100		

There are 10 pages in this quiz, including this one. In addition, tear-off sheets are provided at the end with duplicate drawings and data. As always, open book, open notes, open just about everything.

This isn't supposed to be on the quiz!

Problem 1: Bayesian Inference (49 points)

Part A: Bayes Nets (30 Points)



M	H
F	0.7
T	0.8

V	R
F	0.1
T	0.3

M	V	Z
F	F	0.01
F	T	0.4
T	F	0.5
T	T	0.6

You decide to use Bayes Nets to help you explain recent sightings of Zombies on campus. In your net you define the following variables:
M = A **M**agical spell was cast { T, F }
V = There is a **V**iral outbreak of H1Z1 {T, F}
H = You see a **H**ippogriff (a magical creature) { T, F }
Z = You see a **Z**ombie { T, F}
R = You see Time t**R**avelers in hazmat suits trying to contain the viral outbreak. {T, F }
Using this Bayes Net, answer the following questions:

A1 (3 points): What is the probability of NOT seeing a Zombie given that NO magic spell has been cast and there is NO viral outbreak? **Write down the probability you are computing and give an numeric answer.**

A2 (3 points): Joint Probability: What is the probability of the following state of the world? P(M=T, H=T, Z=T, V=T, R=T) **Give an expression in terms of the probabilities that can be directly read from the network. You need not compute the final numeric answer.**

A3 (4 points): What is the probability of seeing a Zombie given that there is a viral outbreak? P(Z=T | V=T)

Give an expression in terms of probabilities read from the network. You need not compute a final numeric answer.

A4 (6 points): What is the probability of seeing a Zombie? P(Z=T) **Give an expression in terms of probabilities read from the network. You need not compute a final numeric answer.**

A5 (8 points): What is the probability of a Viral outbreak if you see a Zombie? P(V=T | Z=T): **Give an expression in terms of probabilities read from the network and/or probability computed from a previous questions. You need not compute a final numeric answer.**

A6 (6 points): Write inequalities to give an ordering to these probabilities from smallest to largest
P(V=T | Z = T), P(V=T | Z=T, M=F), P(V=T | Z=T, M=T).

Part B: Naïve Bayes (19 Points)

You decide to create a Naïve Bayes classifier to distinguish Zombies from MIT students.
You survey Zombies and Healthy MIT students based on the following true and false questions:

W – Do you wear tattered clothes?
S – Do you get very little sleep?
N – Do you often venture out at night?
B – Do you enjoy a healthy diet of human brains?

Your survey returned the following data set:

	W = True	S = True	N = True	B = True	# surveyed
Zombie	6	8	9	9	10
Non-Zombie	10	15	12	1	20

B1 (9 points):
Draw the Bayes net (with all CPT tables filled in) for your Naïve Bayes Classifier.

B2 (10 points): You decide to test your classifier on Eric. He has the following characteristic:

He is a night owl (likes to go out late into the night). He wears old and tattered clothes, and because of his classes he gets very little sleep. He admits liking the occasional Kidney Pie, but he would never ever eat human brains.

What is Eric according to your classifier? (Circle one)

Healthy **Zombie** **Can't be determined**

To get full credit, you must show your calculations:

Ahh did in Megu

Question 2: Boosting (50 Points)

After graduating MIT, you get a job working for Van Helsing and Summers, a famous vampire hunter consulting agency. Gabriel Van Helsing, one of the two founders, once attended several 6.034 lectures as a guest, and he remembers Professor Winston's Vampire Identification Tree lecture. He assigns you the task of creating a superior classifier for vampires by using boosting on the following data.

Vampires:

ID	Name	Vampire	Evil	Emo	Transforms	Sparkly	# Romantic Interests	ID
1	Dracula	**Y**	Y	N	Y	N	5	1
2	Angel	**Y**	N	Y	Y	N	5	2
3	Edward Cullen	**Y**	N	Y	N	Y	1	3
4	Saya Otonashi	**Y**	N	Y	N	N	3	4
5	Lestat de Lioncourt	**Y**	N	Y	N	N	5	5
6	Bianca St. Claire	**Y**	Y	N	Y	N	5	6
7	Mircalla Karnstein	**Y**	Y	N	Y	N	5	7
8	Sailor Moon	**N**	N	N	Y	Y	1	8
9	Squall Leonhart	**N**	N	Y	N	N	1	9
10	Circe	**N**	N	N	N	N	5	10

Part A (10 points)

To make it easier for yourself, you first determine for each possible classifier (including the classifiers True and False which mean that everyone is a Vampire or not a Vampire, respectively), which of the data points would be misclassified. For example, for the first classifier, the one that says a person is a vampire if he is Evil (and not a vampire if not evil) the classifier is wrong on sample 2, 3, 4, and 5. We have given each classifier a name to make it easier for you to refer to them.

Classifier	Test	Value	Misclassified
A	Evil	Y	2, 3, 4, 5
B	Emo	Y	1, 6, 7, 9
C	Transforms	Y	3, 4, 5, 8
D	Sparkly	Y	1, 2, 4, 5, 6, 7, 8,
E	# of Romantic interests	> 2	3, 10
F	# of Romantic interests	> 4	3, 4, 10
G	TRUE		8, 9, 10
H	Evil	N	
I	Emo	N	skipping ...
J	Transforms	N	
K	Sparkly	N	
L	# of Romantic interests	< 2	
M	# of Romantic interests	< 4	
N	FALSE		

where not sure long + boring

9: can auto eliminate

Part B (5 points)

Again to help yourself later, only 6 of these classifiers would ever be used because the other 8 make all the same errors as one of the other classifiers and then make additional errors. **Circle the 6 above** that you would consider using. **If you are sure you have the right 6, you don't have to waste your time with any of the others in Part C.**

Part C (15 points)

Now you are ready to perform Boosting on the collected Vampire dataset. Fill in the following blanks for the weights, classifiers, errors, and alphas of the first three rounds of Boosting. If there is ever a tie, break it by choosing the classifier that is higher on the list in Part A. Remember to only use the classifiers you circled in Part B.

Pick lowest

	Round 1	Round 2	Round 3
w1	1/16	1/8 4/22 was 1/24	
w2	1/16	1/22 1/24	
w3	4/16	1/22 1/6	
w4		1/22 1/24	
w5		1/22 1/24	
w6	1/16	4/22 1/8	
w7		4/22 1/8	
w8		1/22 1/24	
w9		4/22 1/8	
w10	4/16	1/22 1/6	
h	E	B 1,6,7,9	A
ε	2/16	4/16	
α	$\frac{1}{2}\ln\left(\frac{1-2/10}{2/10}\right)$	$\frac{1}{2}\ln\left(\frac{1-4/16}{4/16}\right)$	

Oh opps → I considered round 1 as round 0

So not keep →

applies here

applies here

2345

? Should reduce

? not for others add up error
Lnot much room here

Part D (10 points)

Rest 12 · 2 = 24 then $\frac{4}{16} \cdot \frac{2}{3} = \frac{1}{6}$

What is the final classifier you find when performing three rounds of Boosting:

Add up αs

Sign ex $\left[\frac{1}{2}\ln 4\,[E] + \frac{1}{2}\ln[B] + \frac{1}{2}\ln\frac{17}{7}\,[A]\right]$

wait $\frac{4}{24}$ So just change denom!

How many of the data points does your final classifier classify correctly?

9

" So what are we actually doing here?

$\frac{4}{16} \cdot 1 = \frac{4}{24}$

So for each point 1 if that classifier is yes (-1 no)
Sum it up. If >0 then yes is vampire
<0 no not "

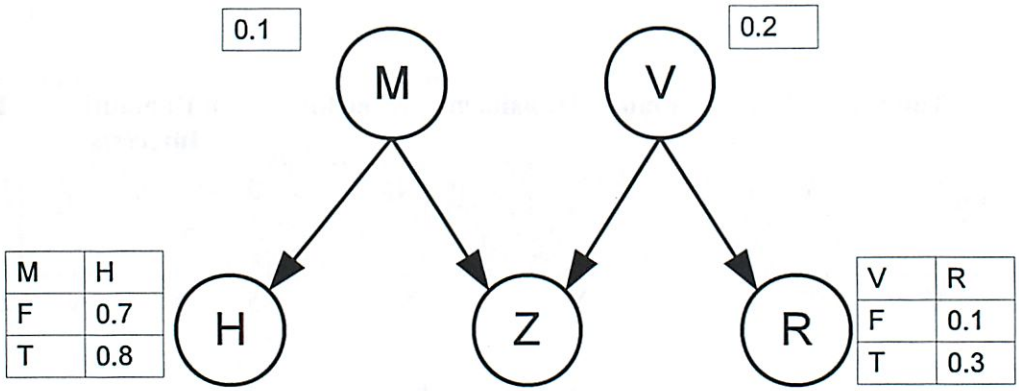
? the mean recitation way

Part E (10 points)

Wesley Windham-Pryce, a fellow consultant, has noticed a few correlations between some of the classifiers you used, and so he suggests using a new set of weak classifiers consisting of each pair of your original weak classifiers logically ANDed or ORed together (so for instance, two of the new classifiers would be "Emo=Y OR Evil=Y" and "Sparkly=N AND Transforms=Y"). He believes that you will be able to classify large vampire datasets more quickly and with fewer rounds of boosting using his system. Do you agree or disagree with Wesley? Explain your argument briefly and precisely.

Tear off sheet—you need not hand this in.

ID	Name	Vampire	Evil	Emo	Transforms	Sparkly	# Romantic Interests	ID
1	Dracula	Y	Y	N	Y	N	5	1
2	Angel	Y	N	Y	Y	N	5	2
3	Edward Cullen	Y	N	Y	N	Y	1	3
4	Saya Otonashi	Y	N	Y	N	N	3	4
5	Lestat de Lioncourt	Y	N	Y	N	N	5	5
6	Bianca St. Claire	Y	Y	N	Y	N	5	6
7	Mircalla Karnstein	Y	Y	N	Y	N	5	7
8	Sailor Moon	N	N	N	Y	Y	1	8
9	Squall Leonhart	N	N	Y	N	N	1	9
10	Circe	N	N	N	N	N	5	10



M	H
F	0.7
T	0.8

V	R
F	0.1
T	0.3

M	V	Z
F	F	0.01
F	T	0.4
T	F	0.5
T	T	0.6

6.034 Quiz 4
December 2, 2009

Name OLD SCRATCH

email [redacted]

Circle your TA and recitation time **(for 1 point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

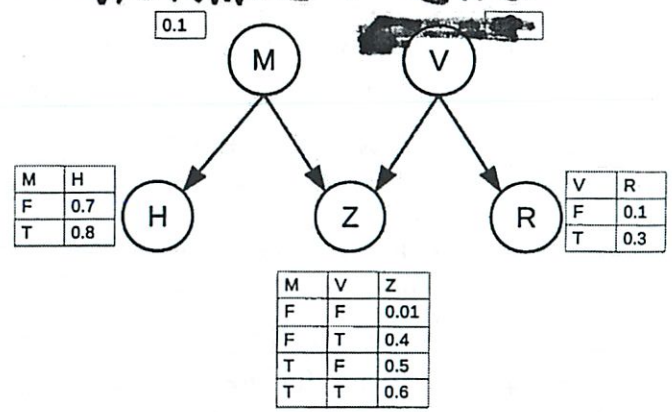
TAs	Thu Time	Thu Instructor	Fri Time	Fri Instructor
Erica Cooper	12-1	Gregory Marton	1-2	Randall Davis
Matthew Peairs	1-2	Berwick/Marton	2-3	Randall Davis
Lisa Fisher	2-3	Berwick/Marton	3-4	Randall Davis
Mark Seifter	3-4	Berwick/Marton		
Yuan K. Shen				
Jeremy Smith				
Olga Wichrowska				

Problem number	Maximum	Score	Grader
1	49		
2	50		
Total	100		

There are 10 pages in this quiz, including this one. In addition, tear-off sheets are provided at the end with duplicate drawings and data. As always, open book, open notes, open just about everything.

Problem 1: Bayesian Inference (49 points)

Part A: Bayes Nets (30 Points)



You decide to use Bayes Nets to help you explain recent sightings of Zombies on campus. In your net you define the following variables:
M = A Magical spell was cast { T, F }
V = There is a Viral outbreak of H1Z1 {T, F}
H = You see a Hippogriff (a magical creature) { T, F }
Z = You see a Zombie { T, F}
R = You see Time tRavelers in hazmat suits trying to contain the viral outbreak. {T, F }
Using this Bayes Net, answer the following questions:

A1 (3 points): What is the probability of NOT seeing a Zombie given that NO magic spell has been cast and there is NO viral outbreak? **Write down the probability you are computing and give an numeric answer.**

$P(Z=F \mid M=F, V=F) = 1 - 0.01 = \boxed{0.99}$

A2 (3 points): Joint Probability: What is the probability of the following state of the world? P(M=T, H=T, Z=T, V=T, R=T) **Give an expression in terms of the probabilities that can be directly read from the network. You need not compute the final numeric answer.**

$$P(M=T, H=T, Z=T, V=T, R=T) =$$
$$= P(M=T)\, P(V=T)\, P(H=T \mid M=T)\, P(Z=T \mid M=T, V=T)\, P(R=T \mid V=T)$$
$$= 0.1 \cdot 0.2 \cdot 0.8 \cdot 0.6 \cdot 0.3$$
$$= 0.0028$$

A3 (4 points): What is the probability of seeing a Zombie given that there is a viral outbreak? P(Z=T | V=T)

Give an expression in terms of probabilities read from the network. You need not compute a final numeric answer.

$$P(Z=T \mid V=T) = P(Z=T \mid M=T, V=T)\, P(M=T)$$
$$+ P(Z=T \mid M=F, V=T)\, P(M=F)$$
$$= 0.6 \cdot 0.1 + 0.4 \cdot (1-0.1) = 0.42$$

A4 (6 points): What is the probability of seeing a Zombie? P(Z=T) **Give an expression in terms of probabilities read from the network. You need not compute a final numeric answer.**

$$P(Z=T) = P(Z=T \mid M=F, V=F)\, P(M=F)\, P(V=F)$$
$$+ P(Z=T \mid M=F, V=T)\, P(M=F)\, P(V=T)$$
$$+ P(Z=T \mid M=T, V=F)\, P(M=T)\, P(V=F)$$
$$+ P(Z=T \mid M=T, V=T)\, P(M=T)\, P(V=T)$$
$$= 0.01\,(0.99)(1-0.2) + 0.4(1-0.1)(0.2) + 0.5(0.1)(1-0.2) + 0.6 \cdot 0.1 \cdot 0.2$$
$$\doteq 0.1312$$

A5 (8 points): What is the probability of a Viral outbreak if you see a Zombie? P(V=T | Z=T): **Give an expression in terms of probabilities read from the network and/or probability computed from a previous questions. You need not compute a final numeric answer.**

$$P(V=T \mid Z=T) = \frac{P(Z=T \mid V=T)\, P(V=T)}{P(Z=T)} = \frac{0.42 \quad 0.2}{0.1312} = 0.6402$$

$P(Z=T \mid V=T)$ [from A3] $P(V=T)$ from CPT.

$P(Z=T)$ [from A4]

A6 (6 points): Write inequalities to give an ordering to these probabilities from smallest to largest P(V=T | Z = T), P(V=T | Z=T, M=F), P(V=T | Z=T, M=T).

$$P(V=T \mid Z=T, M=T) < P(V=T \mid Z=T) < P(V=T \mid Z=T, M=F)$$

Intuition: "Explaining away" – we have two competing causes for Zombies (Z), V and M, knowing that M=T makes the viral cause less likely (less than not knowing about it)

Similarly, knowing that M=F makes the Viral cause more likely because M is now the single remaining cause.

You can also prove this numerically, but the intention of this question is to reason the answer out intuitively using "Explaining Away".

Numerical sol'n:

$$P(V=T \mid Z=T, M=F) = P(Z=T \mid M=F, V=T)\, P(V=T \mid M=F) \,/\, P(Z=T \mid M=F)$$
$$0.4 \quad 0.2 \,/\, 0.088 = 0.9091$$

$$P(V=T \mid Z=T) = P(Z=T \mid V=T) \cdot P(V=T) \,/\, P(Z=T)$$
$$= 0.42 \quad 0.2 \,/\, 0.1312 = 0.6402$$

$$P(V=T \mid Z=T, M=T) = P(Z=T \mid V=T, V=T)\, P(V=T \mid M=T) \,/\, P(Z=T \mid M=T)$$
$$= 0.6 \cdot 0.2 / 0.52 \doteq 0.2307$$

Part B: Naïve Bayes (19 Points)

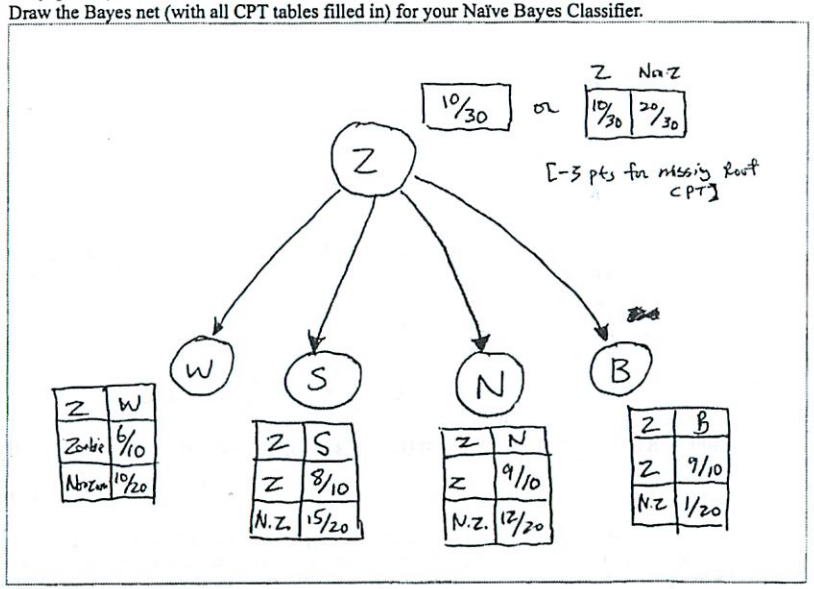
You decide to create a Naïve Bayes classifier to distinguish Zombies from MIT students.
You survey Zombies and Healthy MIT students based on the following true and false questions:

W – Do you wear tattered clothes?
S – Do you get very little sleep?
N – Do you often venture out at night?
B – Do you enjoy a healthy diet of human brains?

Your survey returned the following data set:

	W = True	S = True	N = True	B = True	# surveyed
Zombie	6	8	9	9	10
Non-Zombie	10	15	12	1	20

B1 (9 points):
Draw the Bayes net (with all CPT tables filled in) for your Naïve Bayes Classifier.



B2 (10 points): You decide to test your classifier on Eric. He has the following characteristic:

He is a night owl (likes to go out late into the night). He wears old and tattered clothes, and because of his classes he gets very little sleep. He admits liking the occasional Kidney Pie, but he would never ever eat human brains.

What is Eric according to your classifier? (Circle one)

Healthy **Zombie** **Can't be determined** (2 pts)

To get full credit, you must show your calculations:

$$P(Z=T \mid S=T, N=T, W=T, B=F) \propto P(S=T \mid Z=T)\, P(N=T \mid Z=T)\, P(W=T \mid Z=T)\, P(B=F \mid Z=T)\, P(Z=T)$$

$$= \frac{6}{10} \cdot \frac{8}{10} \cdot \frac{9}{10}\left(1 - \frac{9}{10}\right) \cdot \frac{10}{30}$$

(6pts for math)

$$\cong 0.01438$$

Same as Non Zombie

$$P(Z=F \mid S=T, N=T, W=T, B=F) \propto P(S=T \mid Z=F)\, P(N=T \mid Z=F)\, P(W=T \mid Z=F)\, P(B=F \mid Z=F)\, P(Z=F)$$

$$= \frac{10}{20} \cdot \frac{15}{20} \cdot \frac{12}{20} \cdot \left(1 - \frac{1}{20}\right) \cdot \frac{20}{30}$$

$$\cong 0.1415$$

Argmax Z $\Rightarrow$ $\boxed{Z=F}$ which implies Eric is a Non Zombie and hence healthy.

(2pts for argmax or ratio based conclusion)

Question 2: Boosting (50 Points)

After graduating MIT, you get a job working for Van Helsing and Summers, a famous vampire hunter consulting agency. Gabriel Van Helsing, one of the two founders, once attended several 6.034 lectures as a guest, and he remembers Professor Winston's Vampire Identification Tree lecture. He assigns you the task of creating a superior classifier for vampires by using boosting on the following data.

Vampires:

ID	Name	Vampire	Evil	Emo	Transforms	Sparkly	# Romantic Interests	ID
1	Dracula	Y	Y	N	Y	N	5	1
2	Angel	Y	N	Y	Y	N	5	2
3	Edward Cullen	Y	N	Y	N	Y	1	3
4	Saya Otonashi	Y	N	Y	N	N	3	4
5	Lestat de Lioncourt	Y	N	Y	N	N	5	5
6	Bianca St. Claire	Y	Y	N	Y	N	5	6
7	Mircalla Karnstein	Y	Y	N	Y	N	5	7
8	Sailor Moon	N	N	N	Y	Y	1	8
9	Squall Leonhart	N	N	Y	N	N	1	9
10	Circe	N	N	N	N	N	5	10

Part A (10 points)

To make it easier for yourself, you first determine for each possible classifier (including the classifiers True and False which mean that everyone is a Vampire or not a Vampire, respectively), which of the data points would be misclassified. For example, for the first classifier, the one that says a person is a vampire if he is Evil (and not a vampire if not evil) the classifier is wrong on sample 2, 3, 4, and 5. We have given each classifier a name to make it easier for you to refer to them.

Classifier	Test	Value	Misclassified
A	Evil	Y	2, 3, 4, 5
B	Emo	Y	1,6,7,9
C	Transforms	Y	3,4,5,8
D	Sparkly	Y	1,2,4,5,6,7,8
E	# of Romantic interests	> 2	3,10
F	# of Romantic interests	> 4	3,4,10
G	TRUE		8,9,10
H	Evil	N	1,6,7,8,9,10
I	Emo	N	2,3,4,5,8,10
J	Transforms	N	1,2,6,7,9,10
K	Sparkly	N	3,9,10
L	# of Romantic interests	< 2	1,2,4,5,6,7,8,9
M	# of Romantic interests	< 4	1,2,5,6,7,8,9
N	FALSE		1,2,3,4,5,6,7

Why this

not

Part B (5 points)

Again to help yourself later, only 6 of these classifiers would ever be used because the other 8 make all the same errors as one of the other classifiers and then make additional errors. **Circle the 6 above** that you would consider using. **If you are sure you have the right 6, you don't have to waste your time with any of the others in Part C.**

Part C (15 points)

Now you are ready to perform Boosting on the collected Vampire dataset. Fill in the following blanks for the weights, classifiers, errors, and alphas of the first three rounds of Boosting. If there is ever a tie, break it by choosing the classifier that is higher on the list in Part A. Remember to only use the classifiers you circled in Part B.

	Round 1	Round 2	Round 3
w1	1/10	1/16	1/8
w2	1/10	1/16	1/24
w3	1/10	1/4 (= 4/16)	1/6
w4	1/10	1/16	1/24
w5	1/10	1/16	1/24
w6	1/10	1/16	1/8
w7	1/10	1/16	1/8
w8	1/10	1/16	1/24
w9	1/10	1/16	1/8
w10	1/10	1/4	1/6
h	E	B	A
ε	1/5	1/4	7/24
α	$\frac{1}{2}\ln 4$	$\frac{1}{2}\ln 3$	$\frac{1}{2}\ln\left(\frac{17}{7}\right)$

Part D (10 points)

What is the final classifier you find when performing three rounds of Boosting:

$$\text{Sign}\left[\tfrac{1}{2}\ln 4\,[E] + \tfrac{1}{2}\ln 3\,[B] + \tfrac{1}{2}\ln\tfrac{17}{7}\,[A]\right]$$

How many of the data points does your final classifier classify correctly?

9

Part E (10 points)

Wesley Windham-Pryce, a fellow consultant, has noticed a few correlations between some of the classifiers you used, and so he suggests using a new set of weak classifiers consisting of each pair of your original weak classifiers logically ANDed or ORed together (so for instance, two of the new classifiers would be "Emo=Y OR Evil=Y" and "Sparkly=N AND Transforms=Y"). He believes that you will be able to classify large vampire datasets more quickly and with fewer rounds of boosting using his system. Do you agree or disagree with Wesley? Explain your argument briefly and precisely.

Fundamentally, this kind of combination is what boosting does anyway, using a weighted linear combination of the classifiers, as happens with a perceptron. Giving these combinations to boosting lets it effectively do two rounds in one, but doing each round costs $\binom{n}{2}$ classifier evaluations for each operator. So "fewer rounds" yes, "more quickly" no.

6.034 Quiz 4
1 December 2010

Name	
email	

Circle your TA and recitation time, so that we can more easily enter your score in our records and return your quiz to you promptly.

TAs
Martin Couturier
Kenny Donahue
Bobby Keys
Gleb Kuznetsov
Kendra Pugh
Mark Seifter
Yuan Shen

Thu	
Time	Instructor
1-2	Bob Berwick
2-3	Bob Berwick
3-4	Bob Berwick

Fri	
Time	Instructor
1-2	Randall Davis
2-3	Randall Davis
3-4	Randall Davis

Problem number	Maximum	Score	Grader
1	50		
2	50		
Total	100		

There are 9 pages in this quiz, including this one, but not including blank pages and tear-off sheets. Tear-off sheets are provided at the end with duplicate drawings and data. **As always, open book, open notes, open just about everything, including a calculator, but no computers.**

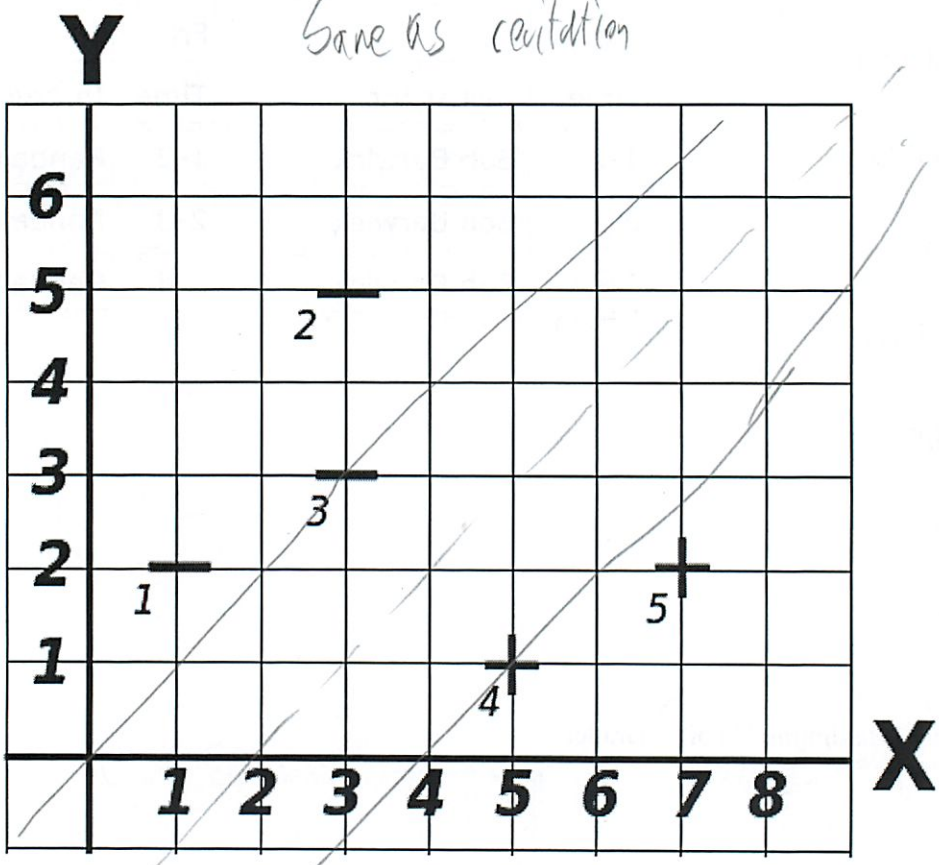
Problem 1: SVMs (50 points)

After reading too much Lord of the Rings, you wake up to find yourself in Middle Earth. You decide the most relevant thing to do is to classify the different races around you.

Part A: Distinguishing Dwarves from Humans (38 points)

You meet 2 dwarves (+) and 3 humans (-) and realize that they have distinguishing features: beard length (x) and height (y). You plot these data points on a grid. Being an expert in SVMs, you decide to start off on an epic journey of classification.

Same as recitation



exactly same as different

A1 (7 points)

Draw the **decision boundary** on the graph above and **clearly label positive and negative gutters, and circle all support vectors.**

What is the width of the road/margin?

$2\sqrt{2}$

A2 (12 points)

Compute $\vec{w}$ and b in the decision boundary $h(\vec{u}) = \vec{w} \cdot \vec{u} + b \geq 0$ for the SVM solution to part A1.

Show your work here.

$\vec{w}$ =

b =

A2 (10 points)

Calculate the weights (alphas) of each data point.
Show your work here.

$\alpha_1 =$

$\alpha_2 =$

$\alpha_3 =$

$\alpha_4 =$

$\alpha_5 =$

A3 (9 points)

What will be the alpha of a new negative point 6 placed at (0, 6)?

What will be the alpha of a new negative point 6 placed at (0,0)?

Supposed we moved point 3 to (4, 2), how will the **magnitude of the alpha 3 change?**

Circle one:

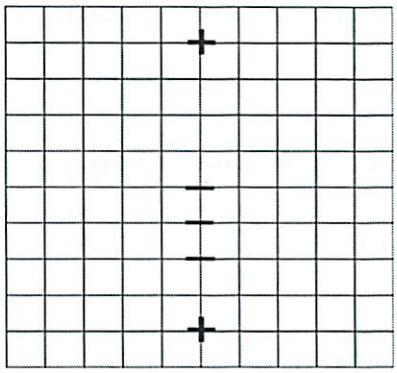
Larger Smaller Same

Part B: Distinguishing Kernels (12 points)

Back in his lab, Gandolf has been hacking on some kernels in preparation for greater classification adventures. For each of the following, indicate **YES** or **NO** whether the kernel can be used to ***perfectly*** classify the test points, and if **YES** ***sketch*** **the decision boundaries and gutters (the street) such a classifier might produce** and ***circle*** **which data points are support vectors.** Note that because of symmetry, more than one answer may be possible for one or more cases.

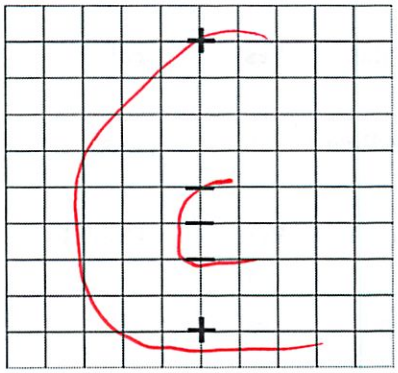
$$K(\vec{u},\vec{v})=\vec{u}\cdot\vec{v}$$

YES **NO** split



$$K(\vec{u},\vec{v})=(\vec{u}\cdot\vec{v}+1)^2$$

YES **NO** polynomic

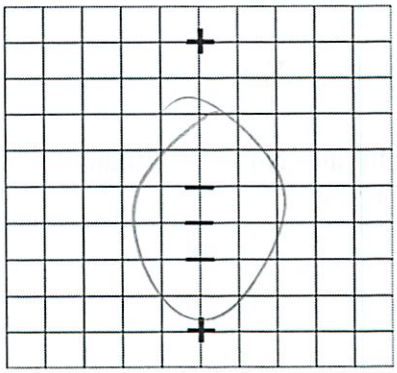


I remember seeing this before...

$$K(\vec{u},\vec{v})=e^{-(|\vec{u}-\vec{v}|)^2/2}$$

YES **NO**

Circle



Problem 2: Boosting (50 points)

After wearing Sauron's ring for several months, Frodo is rapidly losing his sanity. He fears that the ring will interfere with his better judgement and betray him to an enemy. To ensure that he doesn't put his trust into enemy hands, he flees Middle Earth in search of a way to classify his enemies from his friends. In his travels he had heard rumors of the magic of Artificial Intelligence and has decided to hire you to build him a classifier, which will correctly differentiate between his friends and his enemies. Below is all of the information Frodo remembers about the people back in Middle Earth.

ID	Name	Friend	Species	Has Magic	Part of the Fellowship	Has/Had a ring of power	Length of hair (feet)
1	Gandalf	**Yes**	Wizard	Yes	Yes	No	2
2	Sarumon	**No**	Wizard	Yes	No	No	2.5
3	Sauron	**No**	Wizard	Yes	No	Yes	0
4	Legolas	**Yes**	Elf	Yes	Yes	No	2
5	Tree-Beard	**Yes**	Ent	No	No	No	0
6	Sam	**Yes**	Hobbit	No	Yes	No	0.25
7	Elrond	**Yes**	Elf	Yes	No	Yes	2
8	Gollum	**No**	Hobbit	No	No	Yes	1
9	Aragorn	**Yes**	Man	No	Yes	No	0.75
10	Witch-king of Angmar	**No**	Man	Yes	No	Yes	2.5

Part A: Picking Classifiers (10 points)

A1 (6 points)

The data has a high dimensionality and so rather than trying to learn an SVM in a high dimension space you think it would be a smart approach to come up with a series of 1 dimensional stubs that can be used to construct a boosting classifier. Fill in the classifier table below. Each of the different classifiers are given a unique ID and a test returns +1 (friend) if true and -1 (enemy) if false.

Classifier	**Test**	**Misclassified**
A	Species is a Wizard	
B	Species is an Elf	1, 5, 6, 9
C	Species is **not** a Man	
D	Does **not** have magic	1, 4, 7, 8
E	Is **not** part of the Fellowship	
F	Has **never** owned a ring of power	2, 7
G	Hair <= 1ft	
H	Hair <= 2 ft	
I	Friend	2, 3, 8, 10
J	Enemy	

A2 (4 points)

Looking at the results of your current classifiers, you quickly see two more good weak classifiers (make fewer than 4 errors). What are they?

Classifier	**Test**	**Misclassified**
K		
L		

Part B: Build a Strong Classifier (30 points)

B1 (25 points)

You realize that many of your tests are redundant and decide to move forward using only these four classifiers: {**B, D, F, I**}. Run the Boosting algorithm on the dataset with these four classifiers. Fill in the weights, classifiers, errors and alphas for three rounds of boosting. In case of ties, favor classifiers that come first alphabetically.

	Round 1		**Round 2**		**Round 3**	
w1		$h_1 =$		$h_2 =$		$h_3 =$
w2		Err =		Err =		Err =
w3		$\alpha =$		$\alpha =$		$\alpha =$
w4						
w5						
w6						
w7						
w8						
w9						
w10						
Err(B)						
Err(D)						
Err(F)						
Err(I)						

B2 (5 points)

What is the resulting classifier that you obtain after three rounds of Boosting?

H(x) =

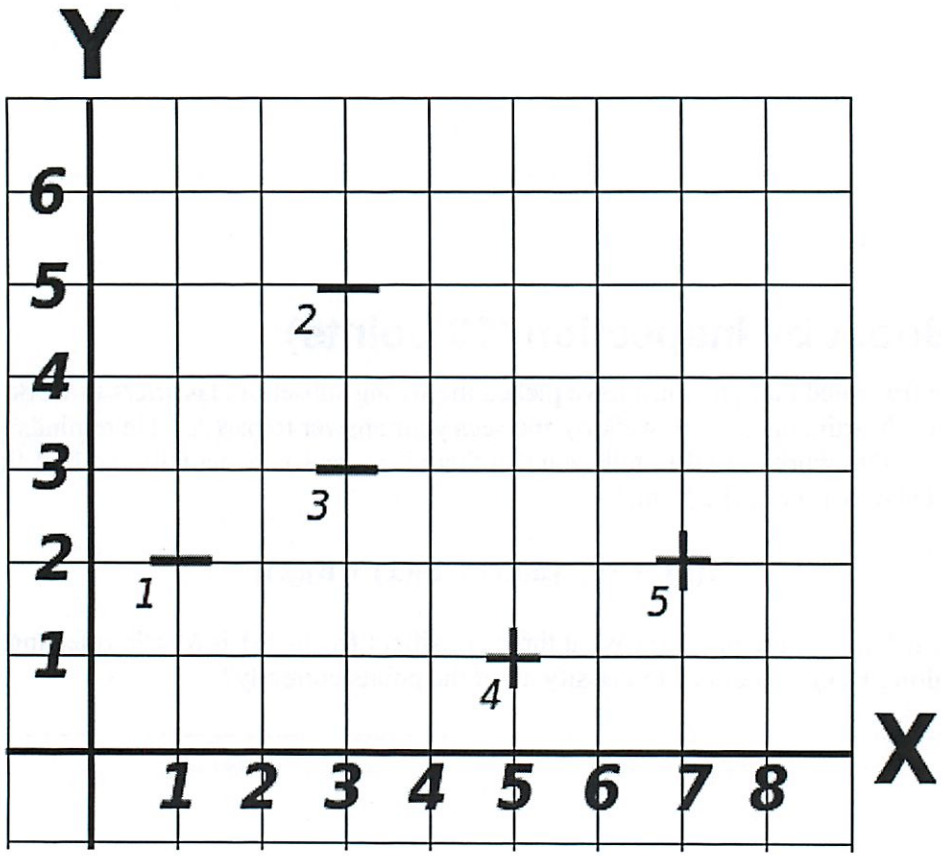
Part C: Boost by Inspection (10 points)

As you become frustrated that you must have picked the wrong subset of classifiers to work with, one of the 6.034 TA's, Martin, happens to walk by and sees your answer to part A1. He reminds you why the boosting algorithm works and then tells you that there is no reason to actually run boosting on this dataset. A boosted classifier of the form:

$$\mathbf{H(x) = Sign[h_1(x) + h_2(x) + h_3(x)]}$$

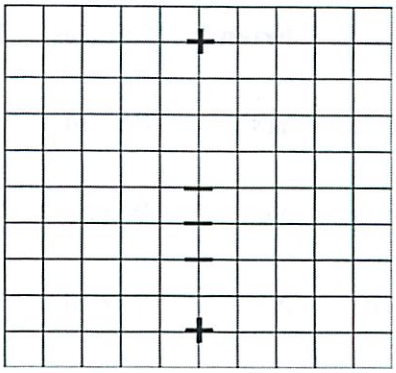
can be found which solves the problem. What three classifiers $\{h_1, h_2, h_3\}$ is Martin referring to, and why is the resulting H(x) guaranteed to classify all of the points correctly?

Tear off sheets. You need not hand these in.



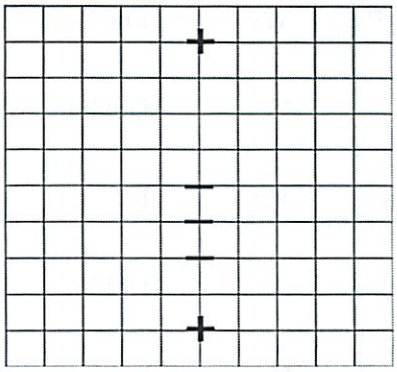
$$K(\vec{u},\vec{v})=\vec{u}\cdot\vec{v}$$

YES **NO**



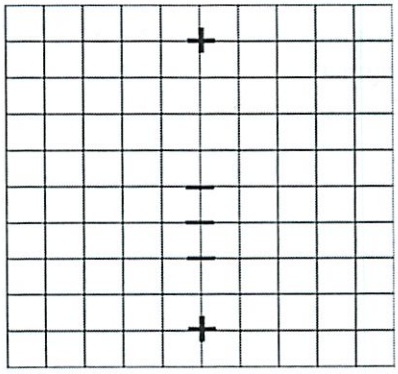
$$K(\vec{u},\vec{v})=(\vec{u}\cdot\vec{v}+1)^2$$

YES **NO**



$$K(\vec{u},\vec{v})=\mathrm{e}^{-(|\vec{u}-\vec{v}|)^2/2}$$

YES **NO**



ID	Name	Friend	Species	Has Magic	Part of the Fellowship	Has/Had a ring of power	Length of hair (feet)
1	Gandalf	**Yes**	Wizard	Yes	Yes	No	2
2	Sarumon	**No**	Wizard	Yes	No	No	2.5
3	Sauron	**No**	Wizard	Yes	No	Yes	0
4	Legolas	**Yes**	Elf	Yes	Yes	No	2
5	Tree-Beard	**Yes**	Ent	No	No	No	0
6	Sam	**Yes**	Hobbit	No	Yes	No	0.25
7	Elrond	**Yes**	Elf	Yes	No	Yes	2
8	Gollum	**No**	Hobbit	No	No	Yes	1
9	Aragorn	**Yes**	Man	No	Yes	No	0.75
10	Witch-king of Angmar	**No**	Man	Yes	No	Yes	2.5

6.034 Quiz 4
1 December 2010

Name	BILBO BAGGINS
email	

Circle your TA and recitation time, so that we can more easily enter your score in our records and return your quiz to you promptly.

TAs
Martin Couturier
Kenny Donahue
Bobby Keys
Gleb Kuznetsov
Kendra Pugh
Mark Seifter
Yuan Shen

Thu	
Time	Instructor
1-2	Bob Berwick
2-3	Bob Berwick
3-4	Bob Berwick

Fri	
Time	Instructor
1-2	Randall Davis
2-3	Randall Davis
3-4	Randall Davis

Problem number	Maximum	Score	Grader
1	50	50	
2	50	50	
Total	100	100	

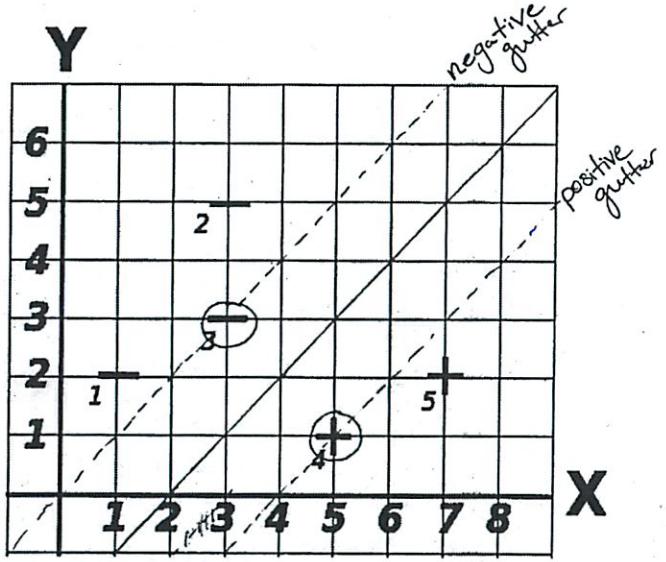
There are 9 pages in this quiz, including this one, but not including blank pages and tear-off sheets. Tear-off sheets are provided at the end with duplicate drawings and data. **As always, open book, open notes, open just about everything, including a calculator, but no computers.**

Problem 1: SVMs (50 points)

After reading too much Lord of the Rings, you wake up to find yourself in Middle Earth. You decide the most relevant thing to do is to classify the different races around you.

Part A: Distinguishing Dwarves from Humans (38 points)

You meet 2 dwarves (+) and 3 humans (-) and realize that they have distinguishing features: beard length (x) and height (y). You plot these data points on a grid. Being an expert in SVMs, you decide to start off on an epic journey of classification.



A1 (7 points)

Draw the **decision boundary** on the graph above and **clearly label positive and negative gutters, and circle all support vectors.**

What is the width of the road/margin?

$2\sqrt{2}$ (must be consistent with your picture)

A2 (12 points)

Compute $\vec{w}$ and b in the decision boundary $h(\vec{u})=\vec{w}\cdot\vec{u}+b\geq 0$ for the SVM solution to part A1.

Show your work here.

$y \leq x - 2$

$x - y - 2 \geq 0$

$cx - cy - 2c \geq 0$

$\vec{w} = \begin{bmatrix} c \\ -c \end{bmatrix}$

$m = \frac{2}{\|\vec{w}\|} = 2\sqrt{2}$

$\frac{2}{\sqrt{c^2+(-c)^2}} = 2\sqrt{2}$

$\frac{4}{2c^2} = 8$

$c = \frac{1}{2}$

$\vec{w} = \begin{bmatrix} 1/2 \\ -1/2 \end{bmatrix}$

$b = -1$

$\frac{1}{2}x - \frac{1}{2}y - 1 \geq 0$

$\vec{w} = \begin{bmatrix} 1/2 \\ -1/2 \end{bmatrix}$

b = -1

A2 (10 points)

Calculate the weights (alphas) of each data point.
Show your work here.

$\sum_i \alpha_i y_i = 0$

$\sum_i \alpha_i y_i \vec{x}_i = \vec{w}$

$\hookrightarrow \alpha_3(-1)\begin{bmatrix} 3 \\ 3 \end{bmatrix} + \alpha_4(1)\begin{bmatrix} 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/2 \\ -1/2 \end{bmatrix}$

$-3\alpha_3 + 5\alpha_4 = 1/2$

$-3\alpha_3 + \alpha_4 = -1/2$ $\rightarrow$ $\alpha_3 = \alpha_4 = 1/4$

α_1 = 0

α_2 = 0

α_3 = 1/4

α_4 = 1/4

α_5 = 0

A3 (9 points)

What will be the alpha of a new negative point 6 placed at (0, 6)?

0

What will be the alpha of a new negative point 6 placed at (0,0)?

0

Supposed we moved point 3 to (4, 2), how will the **magnitude of the alpha 3 change?**

Circle one:

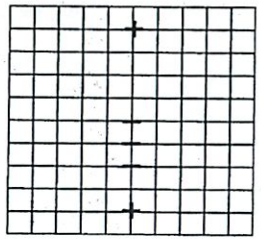
Larger Smaller Same

Part B: Distinguishing Kernels (12 points)

Back in his lab, Gandolf has been hacking on some kernels in preparation for greater classification adventures. For each of the following, indicate **YES** or **NO** whether the kernel can be used to *perfectly* classify the test points, and if **YES** ***sketch*** **the decision boundaries and gutters (the street) such a classifier might produce** and ***circle*** **which data points are support vectors.** Note that because of symmetry, more than one answer may be possible for one or more cases.

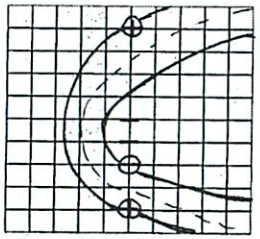
$$K(\vec{u},\vec{v})=\vec{u}\cdot\vec{v}$$

YES (NO)



$$K(\vec{u},\vec{v})=(\vec{u}\cdot\vec{v}+1)^2$$

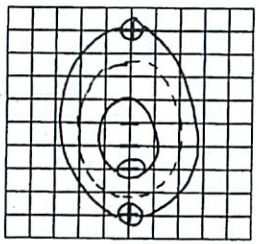
(YES) NO



facing other direction ok

$$K(\vec{u},\vec{v})=e^{-(|\vec{u}-\vec{v}|)^2/2}$$

(YES) NO



Problem 2: Boosting (50 points)

After wearing Sauron's ring for several months, Frodo is rapidly losing his sanity. He fears that the ring will interfere with his better judgement and betray him to an enemy. To ensure that he doesn't put his trust into enemy hands, he flees Middle Earth in search of a way to classify his enemies from his friends. In his travels he had heard rumors of the magic of Artificial Intelligence and has decided to hire you to build him a classifier, which will correctly differentiate between his friends and his enemies. Below is all of the information Frodo remembers about the people back in Middle Earth.

ID	Name	Friend	Species	Has Magic	Part of the Fellowship	Has/Had a ring of power	Length of hair (feet)
1	Gandalf	Yes	Wizard	Yes	Yes	No	2
2	Sarumon	No	Wizard	Yes	No	No	2.5
3	Sauron	No	Wizard	Yes	No	Yes	0
4	Legolas	Yes	Elf	Yes	Yes	No	2
5	Tree-Beard	Yes	Ent	No	No	No	0
6	Sam	Yes	Hobbit	No	Yes	No	0.25
7	Elrond	Yes	Elf	Yes	No	Yes	2
8	Gollum	No	Hobbit	No	No	Yes	1
9	Aragorn	Yes	Man	No	Yes	No	0.75
10	Witch-king of Angmar	No	Man	Yes	No	Yes	2.5

Part A: Picking Classifiers (10 points)

A1 (6 points)

The data has a high dimensionality and so rather than trying to learn an SVM in a high dimension space you think it would be a smart approach to come up with a series of 1 dimensional stubs that can be used to construct a boosting classifier. Fill in the classifier table below. Each of the different classifiers are given a unique ID and a test returns +1 (friend) if true and -1 (enemy) if false.

Classifier	Test	Misclassified
A	Species is a Wizard	2,3,4,5,6,7,9
B	Species is an Elf	1, 5, 6, 9
C	Species is not a Man	2,3,8,9
D	Does not have magic	1, 4, 7, 8
E	Is not part of the Fellowship	1,2,3,4,6,8,9,10
F	Has never owned a ring of power	2, 7
G	Hair <= 1ft	1,3,4,7,8
H	Hair <= 2 ft	3,8
I	Friend	2, 3, 8, 10
J	Enemy	1,4,5,6,7,9

A2 (4 points)

Looking at the results of your current classifiers, you quickly see two more good weak classifiers (make fewer than 4 errors). What are they?

Classifier	Test	Misclassified
K	Species is NOT wizard	1,8,10
L	Is part of fellowship	5,7

Part B: Build a Strong Classifier (30 points)

B1 (25 points)

You realize that many of your tests are redundant and decide to move forward using only these four classifiers: {B, D, F, I}. Run the Boosting algorithm on the dataset with these four classifiers. Fill in the weights, classifiers, errors and alphas for three rounds of boosting. In case of ties, favor classifiers that come first alphabetically.

	Round 1		Round 2		Round 3	
w1	1/10	h_1 = F	1/16	h_2 = B	3/24	h_3 = I
w2	1/10	Err = 2/10	4/16	Err = 4/16	4/24	Err = 7/24
w3	1/10	α = 1/2 ln 4	1/16	α = 1/2 ln 3	1/24	α = 1/2 ln 17/7
w4	1/10		1/16		1/24	
w5	1/10		1/16		3/24	
w6	1/10		1/16		3/24	
w7	1/10		4/16		4/24	
w8	1/10		1/16		1/24	
w9	1/10		1/16		3/24	
w10	1/10		1/16		1/24	
Err(B)	4/10		4/16		12/24	
Err(D)	4/10		7/16		9/24	
Err(F)	2/10		8/16		8/24	
Err(I)	4/10		7/16		7/24	

5pts 10pts 10pts

B2 (5 points)

What is the resulting classifier that you obtain after three rounds of Boosting?

H(x) =

$\text{Sign}\left(\left(\tfrac{1}{2}\ln 4\right)F(x) + \left(\tfrac{1}{2}\ln 3\right)B(x) + \left(\tfrac{1}{2}\ln \tfrac{17}{7}\right)I(x)\right)$

Part C: Boost by Inspection (10 points)

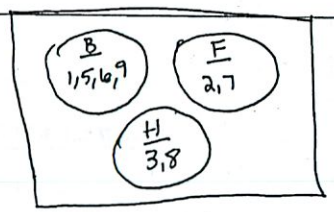
As you become frustrated that you must have picked the wrong subset of classifiers to work with, one of the 6.034 TA's, Martin, happens to walk by and sees your answer to part A1. He reminds you why the boosting algorithm works and then tells you that there is no reason to actually run boosting on this dataset. A boosted classifier of the form:

$$H(x) = \text{Sign}[h_1(x) + h_2(x) + h_3(x)]$$

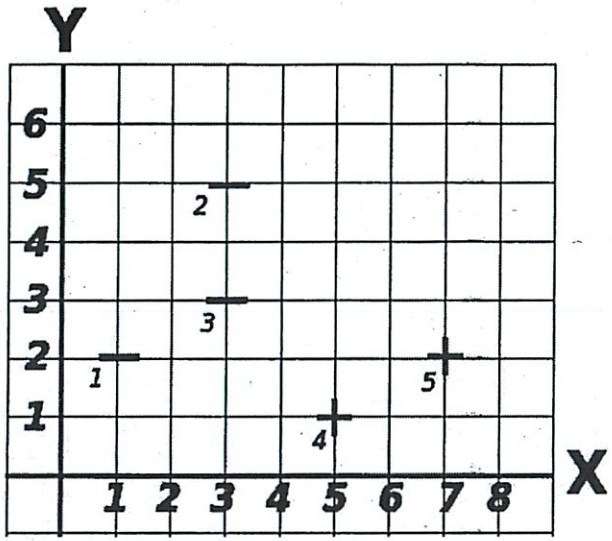
can be found which solves the problem. What three classifiers $\{h_1, h_2, h_3\}$ is Martin referring to, and why is the resulting H(x) guaranteed to classify all of the points correctly?

$\{h_1, h_2, h_3\} = \{B, F, H\}$

The misclassifications of the three classifiers all belong to disjoint sets, implying that a majority vote wins.

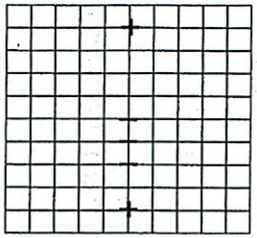


Tear off sheets. You need not hand these in.



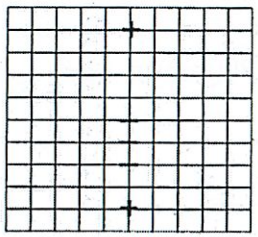
$$K(\vec{u},\vec{v})=\vec{u}\cdot\vec{v}$$

YES **NO**



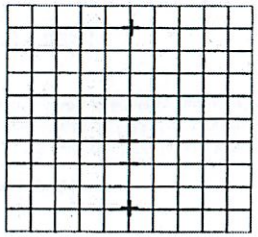
$$K(\vec{u},\vec{v})=(\vec{u}\cdot\vec{v}+1)^2$$

YES **NO**



$$K(\vec{u},\vec{v})=e^{-(|\vec{u}-\vec{v}|)^2/2}$$

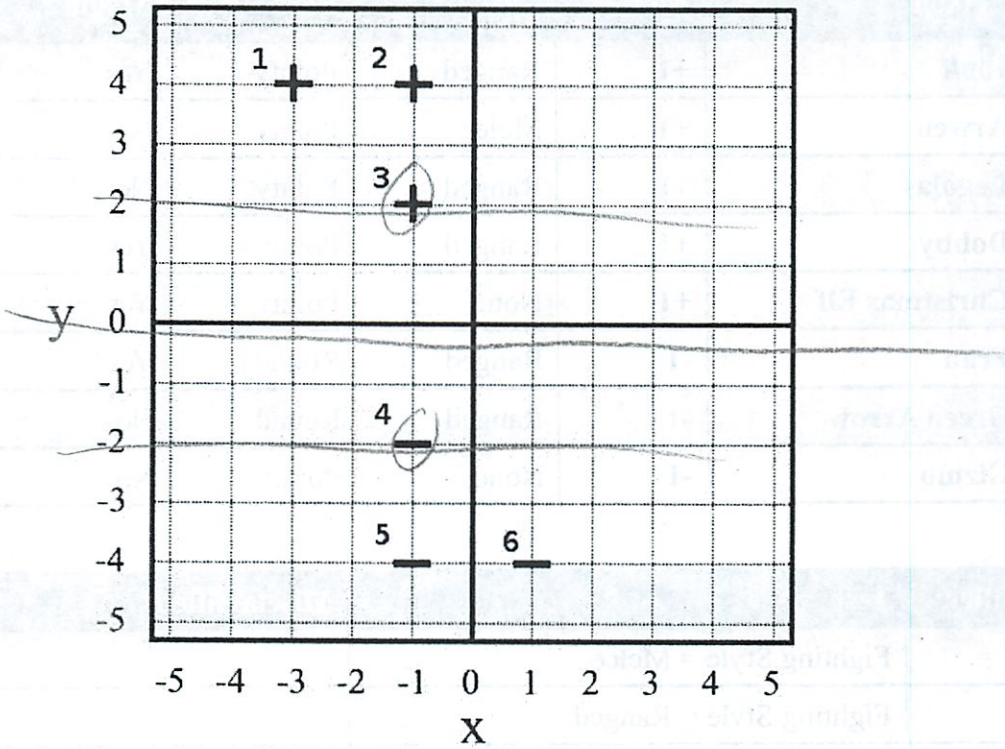
YES **NO**



ID	Name	Friend	Species	Has Magic	Part of the Fellowship	Has/Had a ring of power	Length of hair (feet)
1	Gandalf	Yes	Wizard	Yes	Yes	No	2
2	Sarumon	No	Wizard	Yes	No	No	2.5
3	Sauron	No	Wizard	Yes	No	Yes	0
4	Legolas	Yes	Elf	Yes	Yes	No	2
5	Tree-Beard	Yes	Ent	No	No	No	0
6	Sam	Yes	Hobbit	No	Yes	No	0.25
7	Elrond	Yes	Elf	Yes	No	Yes	2
8	Gollum	No	Hobbit	No	No	Yes	1
9	Aragorn	Yes	Man	No	Yes	No	0.75
10	Witch-king of Angmar	No	Man	Yes	No	Yes	2.5

Tear-off sheet; you need not hand in this page.

Problem 1, Part A



Problem 2

	Person	Elf	Fighting Style	Ears	Magic	Size
1	**Link**	+1	Ranged	Pointy	Yes	Medium
2	**Arwen**	+1	Melee	Pointy	No	Medium
3	**Legolas**	+1	Ranged	Pointy	No	Medium
4	**Dobby**	+1	Ranged	Pointy	Yes	Small
5	**Christmas Elf**	+1	None	Pointy	Yes	Small
6	**Fran**	-1	Ranged	Round	Yes	Medium
7	**Green Arrow**	-1	Ranged	Round	No	Medium
8	**Gizmo**	-1	None	Pointy	No	Small

Classifier	Test	Misclassified
a	Fighting Style = Melee	
b	Fighting Style = Ranged	
c	Fighting Style = None	
d	Ears = Pointy	8
e	Magic = No	
f	Size = Medium	
g	True	

Classifier	Test	Misclassified
i		2, 3, 6
j		5, 6, 7

$\alpha_3 = \alpha_4 + \alpha_7$

$\alpha_4 = \alpha_3 + \alpha_7$

$\alpha_4 = (\alpha_4 + \alpha_7) + \alpha_7$

$0 = 2\alpha_7$

Not right ?

$\alpha_7 = 0$??

$\alpha_3 = \alpha_4$

6.034 Quiz Debrief

12/7

Brought wrong calc
- But prof Watson said I did not need one

Struggled on math

But think pulled both SVM + Boosting out in last 5 min

Forgot the kernel question - did not see it!!!
- I wonder if there were more I missed

Should have studied more
- focused earlier
- Started earlier
- Always say that

So can work out hard math before exam
- But can do these again
- Its a possibility I did get it right

6.034 Quiz 4
December 7, 2011

Name	Michael Plasmeier
Email	the plaz@mit.edu

Circle your **TA** *and* **recitation (for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

TAs	Recitations
Avril Kenney	Thu. 1-2, Bob Berwick
Adam Mustafa	Thu. 2-3, Bob Berwick
Caryn Krakauer	Thu. 3-4, Bob Berwick
Erek Speed	Fri. 1-2, Randall Davis
Gary Planthaber	Fri. 2-3, Randall Davis
Mick Taylor	Fri. 3-4, Randall Davis
Peter Brin	
Tanya Kortz	

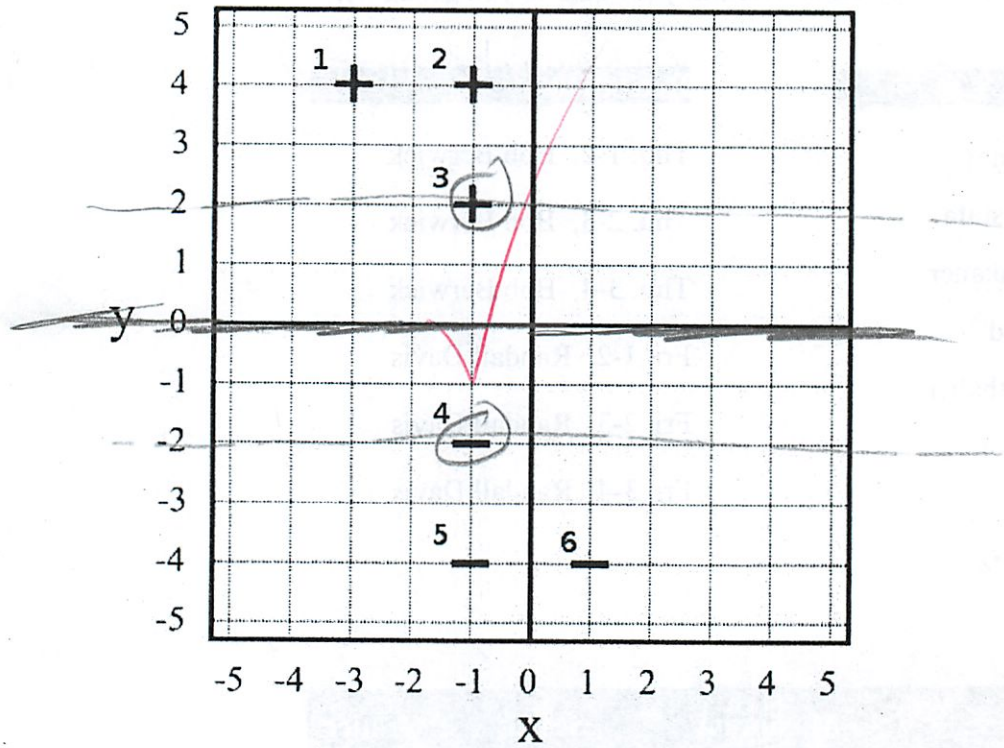
Problem	Maximum	Score	Grader
1	45	30	AM
2	45	35	AM
3	10	4	AFK
Extra Credit	1	1	AFK
Total	101	70	AM

There are a total of 13 pages in this quiz not including one or more tear off sheets that may be provided at the end with duplicate drawings and data. As always, open book, open notes, open just about everything, including a calculator, but no computers.

Problem 1: SVMs (45 points)

Part A: Linear kernels (22 points)

Your good friend, Khan Fusion, tells you about a binary classification technique known as a support vector machine, which appears to be all the rage among computer scientists these days. Unfortunately, Khan is a busy man, and so he could only spend 50 minutes giving you a whirlwind tour to the complex world of SVMs. Eager to see if you're on the right track, you create the following toy classification problem and solve it using an SVM to get a feeling for how it works:



A1 (6 points)

On the picture above, draw with a **heavy solid line** the optimal decision boundary (that is, where the classifier outputs exactly 0) found by an SVM using a linear kernel. Draw with a **heavy dashed line** the edges of the "street" produced by the SVM (that is, where the classifier outputs exactly +1 or -1.) **Circle all support vectors.**

A2(6 points) -2

Give the values that the SVM finds for the following variables; note that **w** is a vector:

$w = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ $\alpha_1 = 0$ $\alpha_3 = 1/4$ $1/8$ $\alpha_5 = 0$

$\alpha_2 = 0$ $\alpha_4 = 1/4$ $1/8$ $\alpha_6 = 0$

$b = 0$

OK

Show your work (for partial credit):

$y = 0$

$y = 0 + 2$

$y = 0 - 2$

$y \geq 0$

$w_1 = 0$ $w_2 = 1$ $b = 0$

$\frac{0}{1}$ ic thing

does not work

Street width = 4

ignore

$\alpha_3 (1) \begin{bmatrix} -1 \\ 2 \end{bmatrix} + \alpha_4 (-1) \begin{bmatrix} -1 \\ -2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

$-\alpha_3 + \alpha_4 = 0$

$\alpha_3 = \alpha_4$

$2\alpha_3 + 2\alpha_4 = 1$

Solve 2 eq, 2 unknown

$2\alpha_4 + 2\alpha_4 = 1$

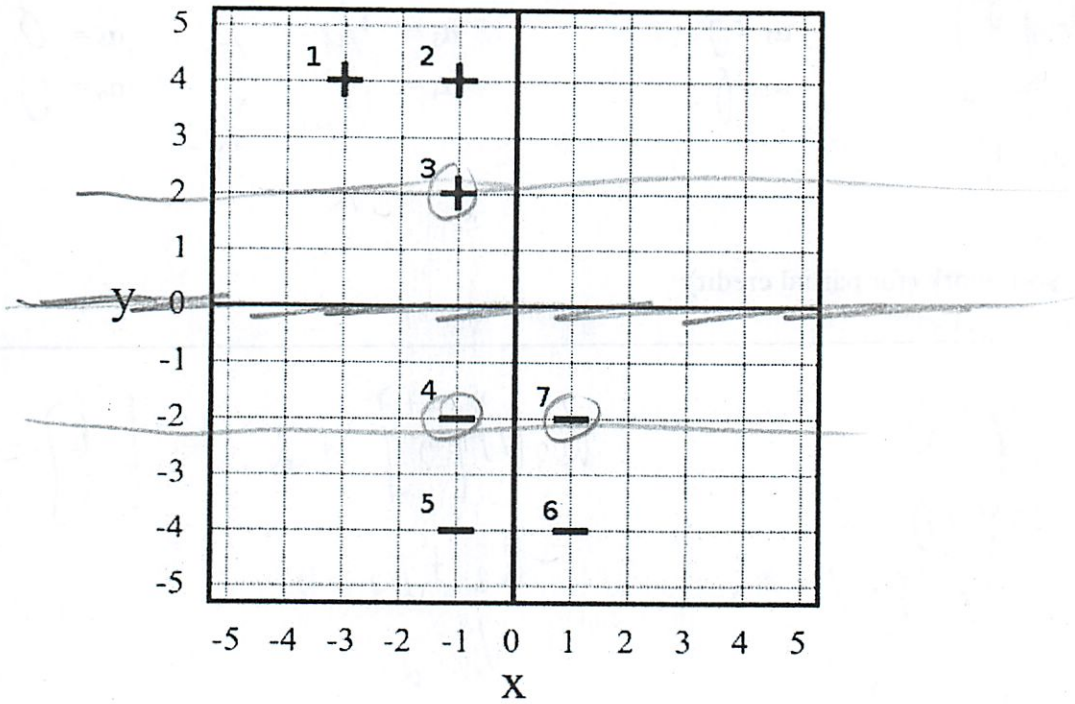
$4\alpha_4 = 1$

$\alpha_4 = \frac{1}{4}$

$\alpha_3 = 1/4$

A3(5 points)

You now add a seventh data point to your graph as follows:



A3a Draw the decision boundary with a **heavy solid line.**

A3b Now what values will the SVM find for each of the following variables?

$\mathbf{w} = \begin{bmatrix} 0 \\ \cancel{1} \end{bmatrix}$ $\alpha_1 = 0$ $\alpha_3 = \cancel{\frac{1}{4}}$ $\alpha_5 = 0$ $\alpha_7 = 0$

$\alpha_2 = 0$ $\alpha_4 = \cancel{\frac{1}{4}}$ $\alpha_6 = 0$

$b = 0$

But consistent

$$\alpha_3(1)\begin{bmatrix} -1 \\ 2 \end{bmatrix} + \alpha_4(-1)\begin{bmatrix} -1 \\ -2 \end{bmatrix} + \alpha_7(-1)\begin{bmatrix} 1 \\ -2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$-\alpha_3 + \alpha_4 - \alpha_7 = 0$

$2\alpha_3 + 2\alpha_4 + \alpha_7 = 1$

? 2 eq, 3 unknowns

also $\alpha_3(1) + \alpha_4(-1) + \alpha_7(-1) = 0$

$\alpha_3 - \alpha_4 - \alpha_7 = 0$

$\alpha_3 = \alpha_4 + \alpha_7$

$\alpha_4 = \alpha_3 + \alpha_7$

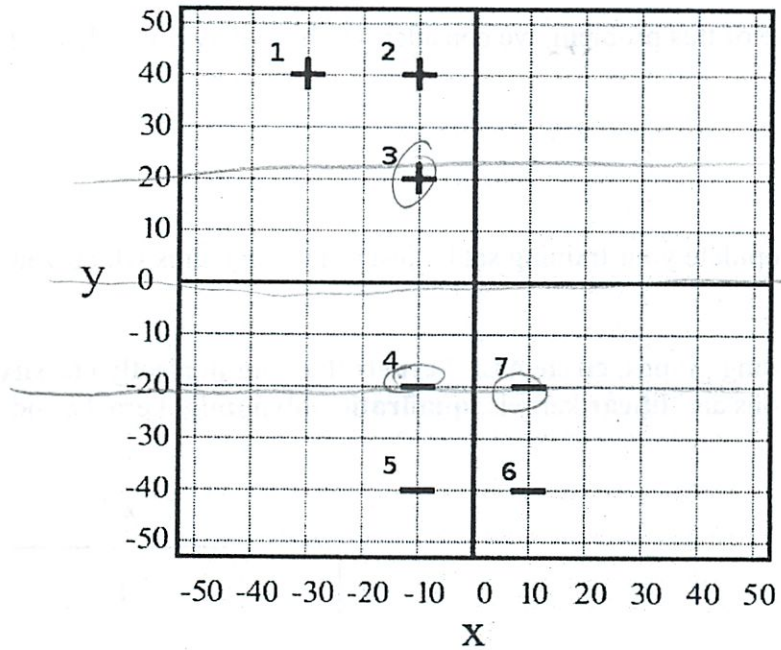
$\alpha_4 = (\alpha_4 + \alpha_7) + \alpha_7$

$0 = 2\alpha_7$

$\alpha_7 = 0$ $\alpha_3 = \alpha_4$

A4(5 points) −4

You want to see how the SVM will react to changing the scale of your problem, so you decide to scale your data by 10 along both dimensions:



What values will the SVM find for each of the following variables?

$\mathbf{w} = \begin{bmatrix} 0 \\ 10 \end{bmatrix}$ $\alpha_1 = 0$ $\alpha_3 = 1/4$ $\alpha_5 = 0$ $\alpha_7 = 0$

$\alpha_2 = 0$ $\alpha_4 = 1/4$ $\alpha_6 = 0$

$b = 0$

Part B: Higher order kernels (23 points)

You're pretty satisfied with your understanding of SVMs at this point, so you decide to use them to tackle a slightly harder problem, certainly worthy of an MIT student's attention: given two input **integers** x and y, can we learn to predict whether their sum, x + y, will be evenly divisible by 2?

(Note: for the purpose of this problem, we consider 0 to be evenly divisible by 2.)

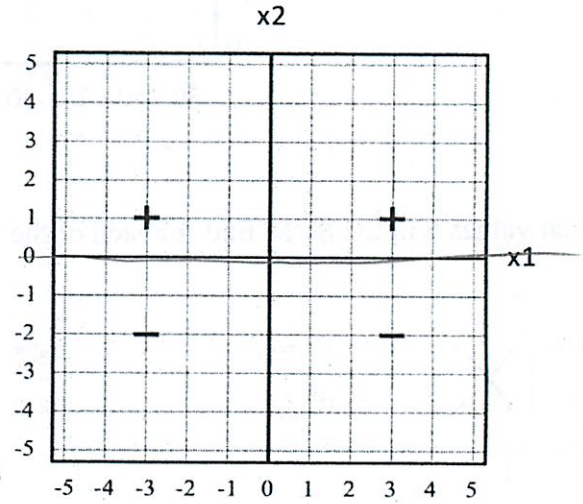
B1 (9 points)

You slowly start to populate your training set by inserting data points which you manually classify.

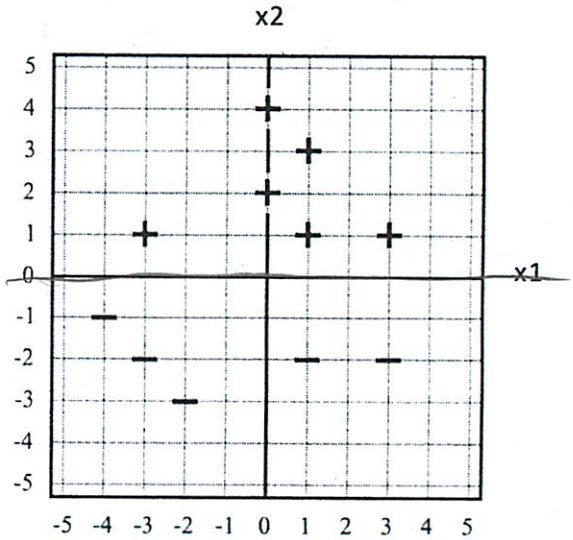
For each of the following graphs, **circle ALL kernels that can perfectly classify the training data.** Your three options are: **linear** kernels, **quadratic polynomial** kernels, and **radial basis function** kernels.

linear **polynomial (quadratic)** **radial**

can do simple functions

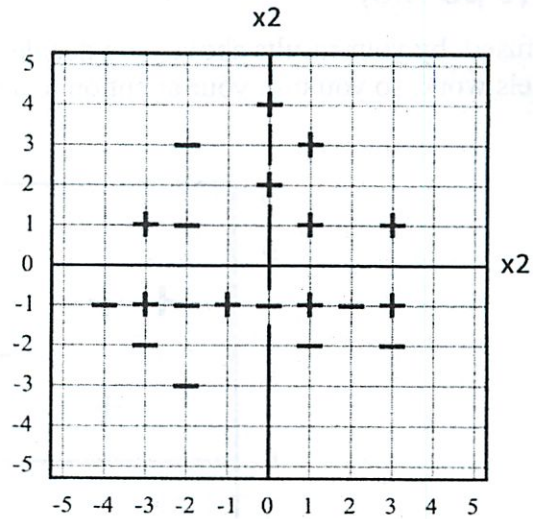


linear **polynomial (quadratic)** **radial**



linear **polynomial (quadratic)** **radial**

multiple radials



B2 (4 points)

-4

You heave a sigh of victory as you finish adding the 121st **distinct** data point to the graph above. You then run your learning algorithm with a radial basis function kernel using suitable parameters. Then, you test your classifier with two random integers between -5 and 5. Will your classifier always be able to classify this data point correctly? **(circle one)**

YES **NO** overfitting to the extreme here

B3 (4 points)

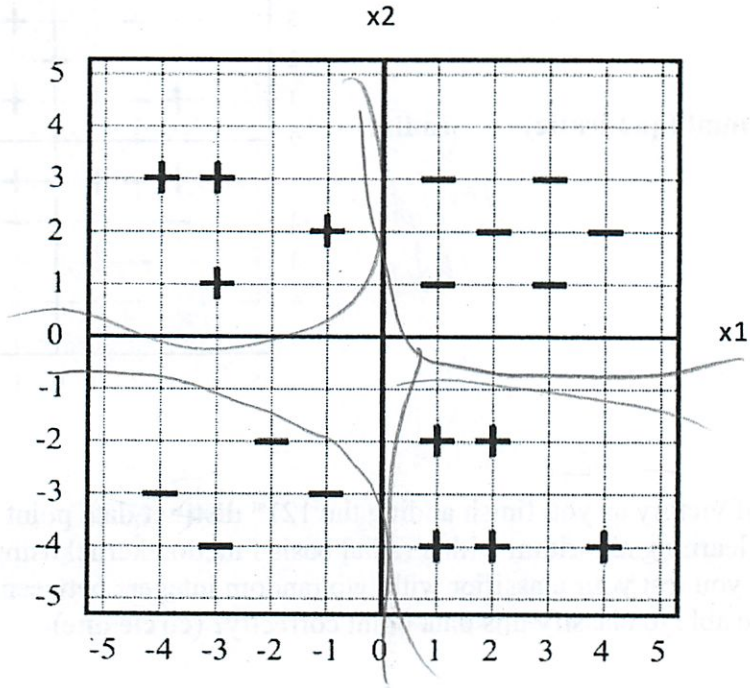
You decide to test your classifier from B2 on random data points outside the range where x and y are between -5 and +5, **without retraining your SVM**. Will your classifier always be able to classify these data points correctly? **(circle one)**

YES **NO** Same as above

too much overfitting

B4 (6 points) −5

Confused by your results above, you decide that you want to get a more intuitive sense of how kernels work, so you turn your attention to a simpler problem:



Give a transformation $\varphi(\mathbf{u})$ (where $\mathbf{u}$ is a vector whose components are u_1 and u_2) that will make these data points linearly separable, and compute the kernel $\mathbf{K}(\mathbf{u}, \mathbf{v})$ associated with the transformation.

$\varphi(\mathbf{u})$:

e^{u}

$\mathbf{K}(\mathbf{u}, \mathbf{v})$:

$\phi(u)\ \phi(v)$

$$\frac{e^{k u \cdot v + b + 1}}{e^{k u \cdot v + b - 1}}$$

Problem 2: Adaboost (45 points)

Your friend Ben has spent the last month playing a new video game instead of attending 6.034. This weekend his girlfriend Brittney is visiting, and they want to make a character for her, but don't know what sort of character it should be. She has given him a description of her ideal character, and Ben has asked you to use your newfound knowledge of Adaboost to build a classifier that determines if she should play an elf or a non-elf.

Below is the table that Ben has compiled of training data for your classifiers. Note that +1 in the Elf column means that the person should be an elf; -1 means the person should not be an elf.

	Person	Elf	Fighting Style	Ears	Magic	Size
1	**Link**	+1	Ranged	Pointy	Yes	Medium
2	**Arwen**	+1	Melee	Pointy	No	Medium
3	**Legolas**	+1	Ranged	Pointy	No	Medium
4	**Dobby**	+1	Ranged	Pointy	Yes	Small
5	**Christmas Elf**	+1	None	Pointy	Yes	Small
6	**Fran**	-1	Ranged	Round	Yes	Medium
7	**Green Arrow**	-1	Ranged	Round	No	Medium
8	**Gizmo**	-1	None	Pointy	No	Small

Part A: Choosing Classifiers (18 points)

A1 (10 points)

-2

Fill in the following chart by indicating which rows, by number, have the wrong Elf value, given the indicated test.

Classifier	Test	Misclassified
a	Fighting Style = Melee	1, 3, 4, 5
b	Fighting Style = Ranged	2, 5, 6, 7
c	Fighting Style = None	1, 2, 3, 4, 8
d	Ears = Pointy	8
e	Magic = No	1, 4, 5, ~~6~~ 7, 8 -2
f	Size = Medium	4, 5, 6, 7
g	True	6, 7, 8

2, 3, 6

5, 6, 7 ← reverse alpha order

A2 (4 points)

You notice that you could add two more good, single test, weak classifiers (fewer than half of the data points are misclassified). Fill in the tests below, given the data points they misclassify.

We have not used **h** in the table to avoid confusion with the weak classifiers that are added up to make the strong classifier, H.

Classifier	Test	Misclassified
i	Magic = No	2, 3, 6
j	Fighting = Ranged || Melee	5, 6, 7

A3 (4 points)

-3

Ben thinks we should use all 9 of these classifiers in boosting, just to be safe. Do you agree? Circle one:

YES **NO**

-strictly worse

Some may be subsets of the others

If you circled NO, list the classifier(s) (by letter) that we will NOT need to use during boosting:

Part B: Running Adaboost (27 points)

B1 (18 points)

−2

Now that you know which weak classifiers you'll use (either all 9 or some subset, depending on your answer to part A3), you're ready to run Adaboost. Fill in the table below for the first three rounds of Adaboost. Break ties by **REVERSE alphabetical order** of the classifier.

	Round 1	Round 2	Round 3
w1	1/8	1/14	1/22
w2	1/8	1/14	1/22
w3	1/8	1/14	1/22
w4	1/8	1/14	1/22
w5	1/8	1/14	1/6
w6	1/8	1/14	1/6
w7	1/8	1/14	1/6
w8	1/8	1/2 7/14	7/22
h	0 8	J 5,6,7	I 2,3,6
ε	1/8	3/14	2/22 + 1/6 = .1818
α	$\frac{1}{2}\ln\left(\frac{1-1/8}{1/8}\right)$	$\frac{1}{2}\ln\left(\frac{1-3/14}{3/14}\right)$	$\frac{1}{2}\ln\left(\frac{.8181}{.1818}\right)$
	$\frac{1}{2}\ln(7)$	$\frac{1}{2}\ln(11/3)$	$\frac{1}{2}\ln(4.5)$

.1818

You can use the space below to show your work:

$\frac{1}{2}\ \frac{1}{1-1/8}$ $\quad \frac{1}{2}\cdot\frac{8}{7} = \frac{8}{14}\cdot\frac{1}{8} = \frac{1}{14}$

$\frac{1}{2}\cdot\frac{1}{1/8} = \frac{8}{1}\cdot\frac{1}{2} = 4\cdot\frac{1}{8} = \frac{1}{2}$

$\frac{1}{2}\cdot\frac{1}{1-3/14}$ $\quad \frac{1}{2}\cdot\frac{14}{11}\cdot\frac{1}{14} = \frac{1}{22}$ $\quad$ 5&7 wrong

$\frac{1}{2}\cdot\frac{1}{3/14}\cdot\frac{1}{14} = \frac{1}{2}\cdot\frac{14}{3}\cdot\frac{1}{14} = \frac{1}{6}\ \frac{2}{12}\ \frac{3}{18}\ \frac{4}{24}$

$\left(\frac{7/8}{1/8}\right.$

$\frac{7}{8}\cdot\frac{8}{1}$

$\frac{11/14}{3/14}$

$\frac{11}{14}\ \frac{14}{3} = \frac{11}{3}$

$\frac{1}{2}\cdot\frac{14}{22}\cdot\frac{1}{14} = \frac{7}{22}$ ←

$\frac{4}{22}\ \frac{2}{11}$

B2 (4 points)

What is the final classifier produced by these three rounds of Adaboost?

H(x) = $\frac{1}{2}\ln(7)D + \frac{1}{2}\ln(\frac{4}{3})J + \frac{1}{2}\ln(4.5)I$ OK

B3 (3 points) −3

Does your classifier correctly classify all of the training data? Circle one:

don't have right calc to check

YES **NO**

If you circled NO, list any training data points that your Adaboost classifier misclassifies:

B4 (2 points)

Below is the description of Brittney's ideal character. According to your classifier, should her character be an elf?

	Elf	Fighting Style	Ears	Magic	Size
Brittney	??	Melee	Pointy	Yes	Medium

Circle one:

ELF **NOT AN ELF**

Problem 3: Representation (10 points)

4/10

Fill in the missing items in table in the most reasonable way using the transition vocabulary:

Appear, Disappear, Change (Δ), Increase (↑), Decrease (↓), not appear, not disappear, not change, not increase and not decrease,

Assume the table is to represent aspects of the meaning found in the following. Ignore the black cells in the table. Assume time increase from left to right.

Patrick and Karen were standing together when Patrick started running. Karen chased him, caught him, stopped him, and hit him with a club.

3 4 5

	start	Karen running.	Chased	stopped	club
	Interval 1	Interval 2	Interval 3	Interval 4	Interval 5
Patrick's speed	Not Δ A	Not Δ	Not Δ	↓ D	Not A ?
Karen's speed	Not A	A	Not Δ		Not A
Distance between K and P			D ↓	D	Not A
Patrick's health	Not Δ	Not Δ	Not Δ	Not Δ	↓

This page intentionally blank.

6.034 Quiz 4
December 7, 2011

Name	Solutions
Email	

Circle your **TA** *and* **recitation (for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

TAs	Recitations
Avril Kenney	Thu. 1-2, Bob Berwick
Adam Mustafa	Thu. 2-3, Bob Berwick
Caryn Krakauer	Thu. 3-4, Bob Berwick
Erek Speed	Fri. 1-2, Randall Davis
Gary Planthaber	Fri. 2-3, Randall Davis
Mick Taylor	Fri. 3-4, Randall Davis
Peter Brin	
Tanya Kortz	

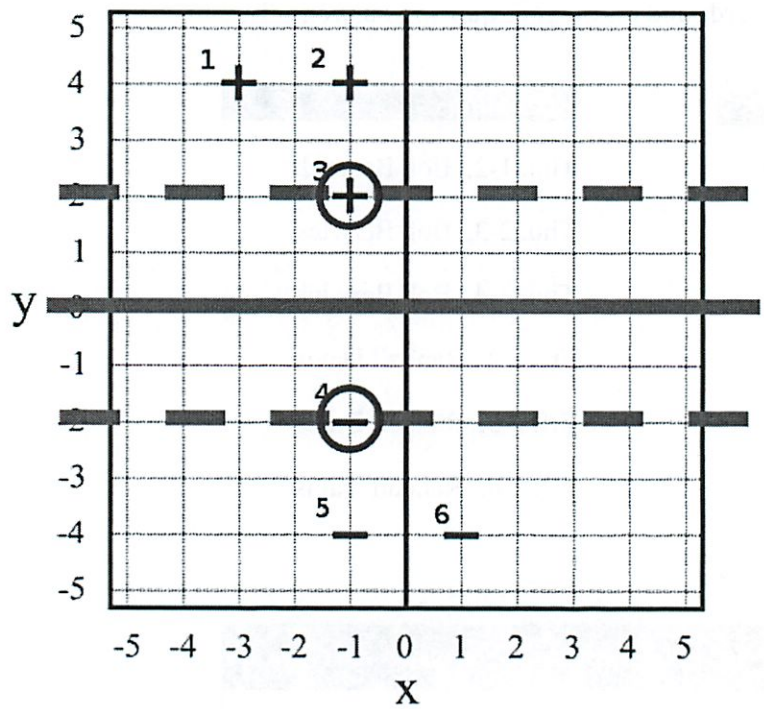
Problem	Maximum	Score	Grader
1	45		
2	45		
3	10		
Extra Credit	1		
Total	101		

There are a total of 13 pages in this quiz not including one or more tear off sheets that may be provided at the end with duplicate drawings and data. As always, open book, open notes, open just about everything, including a calculator, but no computers.

Problem 1: SVMs (45 points)

Part A: Linear kernels (22 points)

Your good friend, Khan Fusion, tells you about a binary classification technique known as a support vector machine, which appears to be all the rage among computer scientists these days. Unfortunately, Khan is a busy man, and so he could only spend 50 minutes giving you a whirlwind tour to the complex world of SVMs. Eager to see if you're on the right track, you create the following toy classification problem and solve it using an SVM to get a feeling for how it works:



A1 (6 points)

On the picture above, draw with a **heavy solid line** the optimal decision boundary (that is, where the classifier outputs exactly 0) found by an SVM using a linear kernel. Draw with a **heavy dashed line** the edges of the "street" produced by the SVM (that is, where the classifier outputs exactly +1 or -1.) **Circle all support vectors.**

A2(6 points)

Give the values that the SVM finds for the following variables:

$\mathbf{w} = \begin{bmatrix} \mathbf{0} \\ \mathbf{1/2} \end{bmatrix}$ $\alpha_1 = 0$ $\alpha_3 = 1/8$ $\alpha_5 = 0$

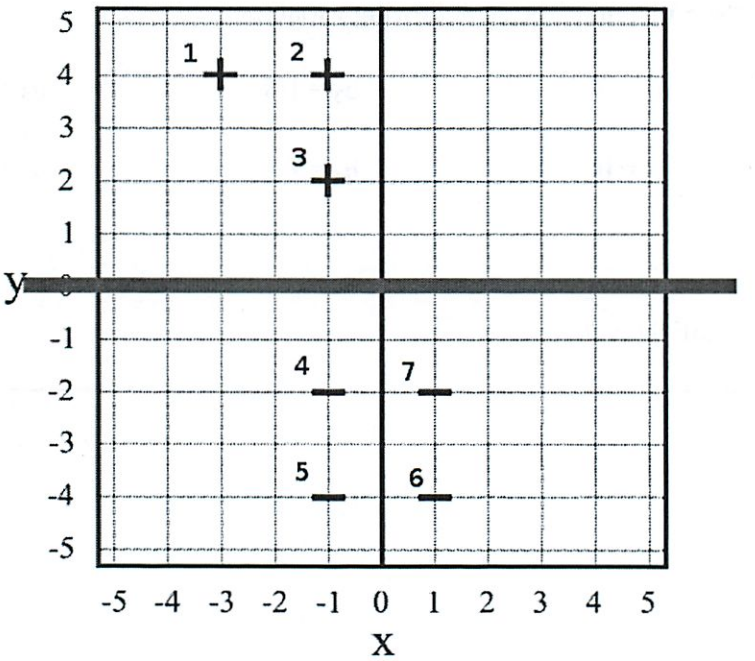
$\alpha_2 = 0$ $\alpha_4 = 1/8$ $\alpha_6 = 0$

$b = 0$

Show your work (for partial credit):

A3(5 points)

You now add a seventh data point to your graph as follows:



A3a Draw the decision boundary with a **heavy solid line.**

A3b What new values will the SVM find for each of the following variables?

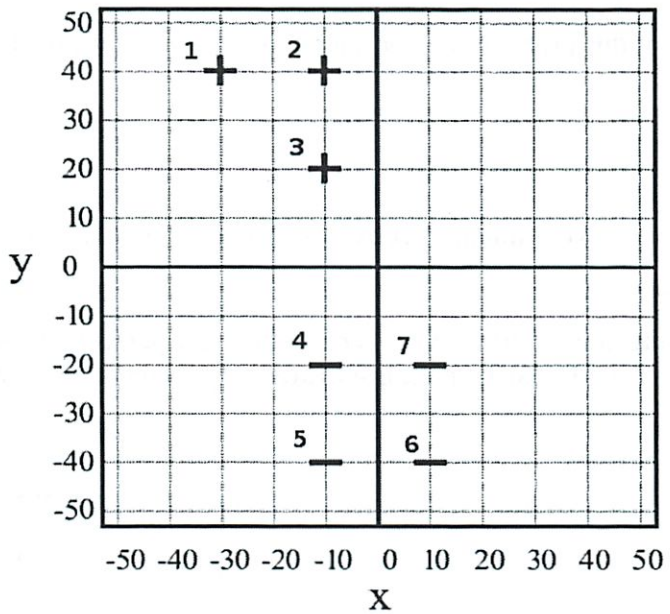
$\mathbf{w} = \begin{bmatrix} \mathbf{0} \\ \mathbf{1/2} \end{bmatrix}$ $\alpha_1 = 0$ $\alpha_3 = 1/8$ $\alpha_5 = 0$ $\alpha_7 = 0$

$\alpha_2 = 0$ $\alpha_4 = 1/8$ $\alpha_6 = 0$

$b = 0$

A4(5 points)

You want to see how the SVM will react to changing the scale of your problem, so you decide to multiply your data by 10 along both dimensions:



What new values will the SVM find for each of the following variables?

$\mathbf{w} = \begin{bmatrix} \mathbf{0} \\ \mathbf{1/20} \end{bmatrix}$ $\alpha_1 = 0$ $\alpha_3 = 1/800$ $\alpha_5 = 0$ $\alpha_7 = 0$

$\alpha_2 = 0$ $\alpha_4 = 1/800$ $\alpha_6 = 0$

$b = 0$

Part B: Higher order kernels (23 points)

You're pretty satisfied with your understanding of SVMs at this point, so you decide to use them to tackle a slightly harder problem, certainly worthy of an MIT student's attention: given two input **integers** x and y, can we learn to predict whether their sum, x + y, will be evenly divisible by 2?

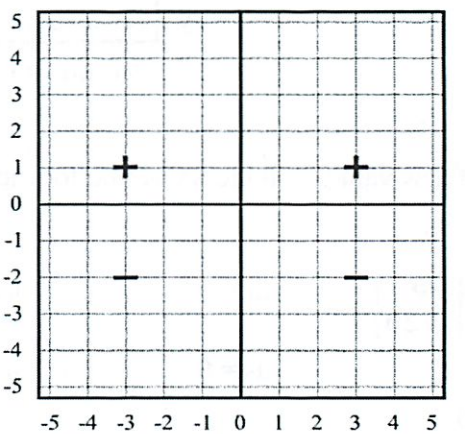
(Note: for the purpose of this problem, we consider 0 to be evenly divisible by 2.)

B1 (9 points)

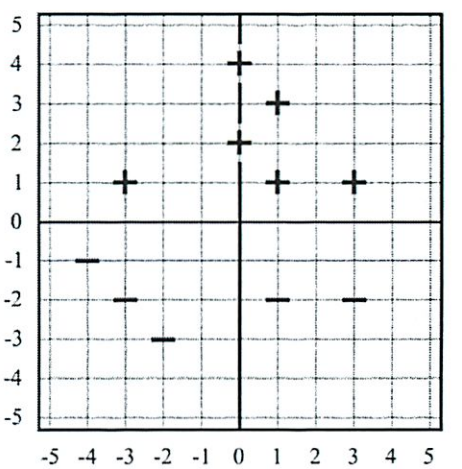
You slowly start to populate your training set by inserting data points which you manually classify.

For each of the following graphs, **circle ALL kernels that can perfectly classify the training data.** Your three options are: **linear** kernels, **quadratic polynomial** kernels, and **radial basis function** kernels.

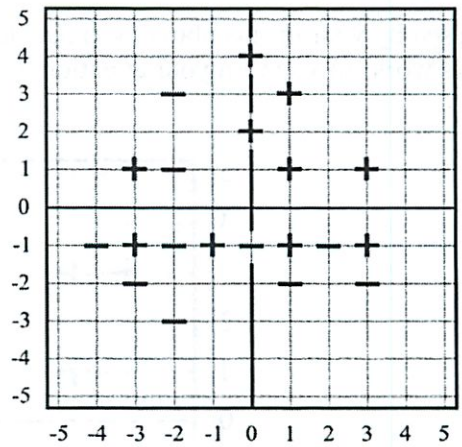
linear **polynomial (quadratic)** **radial**



linear **polynomial (quadratic)** **radial**



linear **polynomial (quadratic)** **radial**



B2 (4 points)

You heave a sigh of victory as you finish adding the 121st **distinct** data point to the graph above. You then run your learning algorithm with a radial basis function kernel using suitable parameters and test your classifier with two random integers between -5 and 5. Will your classifier always be able to classify this data point correctly? (**circle one**)

YES **NO**

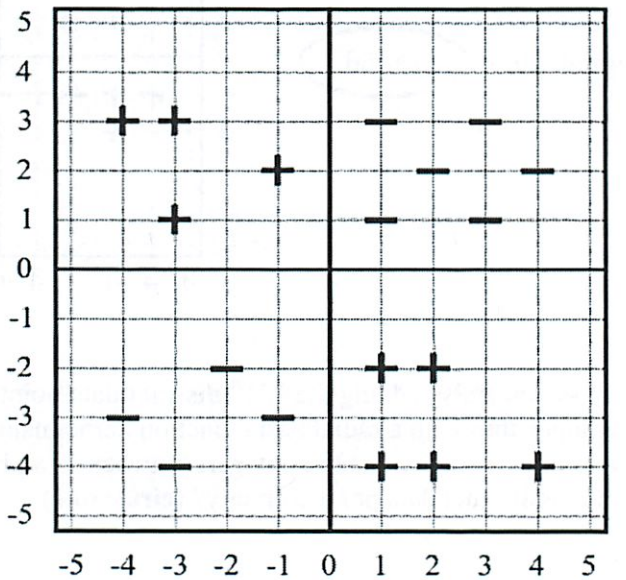
B3 (4 points)

You decide to test your classifier from B2 on random data points outside the range where x and y are between -5 and +5, **without retraining your SVM**. Will your classifier always be able to classify these data points correctly? (**circle one**)

YES **NO**

B4 (6 points)

Confused by your results above, you decide that you want to get a more intuitive sense of how kernels work, so you turn your attention to a simpler problem:



Give a transformation φ(**u**) (where **u** is a vector whose components are u_1 and u_2) that will make these data points linearly separable, and compute the kernel **K(u, v)** associated with the transformation.

φ(**u**): $u_1 u_2$

K(u, v): $\varphi(\boldsymbol{u}) \circ \varphi(\boldsymbol{v}) = u_1 u_2 v_1 v_2$

Note: A variety of solutions were accepted, but the above is one of the simplest. Another is $\varphi(\boldsymbol{u}) = |\boldsymbol{u}_2 - \boldsymbol{u}_1|$.

Problem 2: Adaboost (45 points)

Your friend Ben has spent the last month playing a new video game instead of attending 6.034. This weekend his girlfriend Brittney is visiting, and they want to make a character for her, but don't know what sort of character it should be. She has given him a description of her ideal character, and Ben has asked you to use your newfound knowledge of Adaboost to build a classifier and decide if she should play an elf or a non-elf.

Below is the table that Ben has compiled of training data for your classifiers. Note that +1 in the Elf column means that the person should be an elf; -1 means the person should not be an elf.

	Person	Elf	Fighting Style	Ears	Magic	Size
1	**Link**	+1	Ranged	Pointy	Yes	Medium
2	**Arwen**	+1	Melee	Pointy	No	Medium
3	**Legolas**	+1	Ranged	Pointy	No	Medium
4	**Dobby**	+1	Ranged	Pointy	Yes	Small
5	**Christmas Elf**	+1	None	Pointy	Yes	Small
6	**Fran**	-1	Ranged	Round	Yes	Medium
7	**Green Arrow**	-1	Ranged	Round	No	Medium
8	**Gizmo**	-1	None	Pointy	No	Small

Part A: Choosing Classifiers (18 points)

A1 (10 points)

Fill in the following chart of misclassified data points for the given weak classifiers.

Classifier	Test	Misclassified
a	Fighting Style = Melee	1, 3, 4, 5
b	Fighting Style = Ranged	2, 5, 6, 7
c	Fighting Style = None	1, 2, 3, 4, 8
d	Ears = Pointy	8
e	Magic = No	1, 4, 5, 7, 8
f	Size = Medium	4, 5, 6, 7
g	True	6, 7, 8

A2 (4 points)

You notice that you could add two more good, single test, weak classifiers (fewer than half of the data points are misclassified). Fill in the tests below, given the data points they misclassify.

You may test either for properties or their negations (that is, !large would be a valid test).

We have not used **h** in the table to avoid confusion with the weak classifiers that are added up to make the strong classifier, H.

Classifier	Test	Misclassified
h	Magic = Yes	2, 3, 6
i	Fighting Style ≠ None	5, 6, 7

A3 (4 points)

Ben thinks we should use all 9 of these classifiers in boosting, just to be safe. Do you agree? Circle one:

YES **NO**

If you circled NO, list the classifier(s) (by letter) that we will NOT need to use during boosting:

b, c, e, f, g

Part B: Running Adaboost (27 points)

B1 (18 points)

Now that you know which weak classifiers you'll use (either all 9 or some subset, depending on your answer to part A3), you're ready to run Adaboost. Fill in the table below for the first three rounds of Adaboost. Break ties by **REVERSE alphabetical order** of the classifier.

	Round 1	Round 2	Round 3
w1	1/8	1/14	3/66 = 1/22
w2	1/8	1/14	3/66 = 1/22
w3	1/8	1/14	3/66 = 1/22
w4	1/8	1/14	3/66 = 1/22
w5	1/8	1/14	11/66 = 1/6
w6	1/8	1/14	11/66 = 1/6
w7	1/8	1/14	11/66 = 1/6
w8	1/8	7/14 = 1/2	21/66 = 7/22
h	d	j	i
ε	1/8	3/14	17/66
α	$\frac{1}{2}\ln 7$	$\frac{1}{2}\ln\frac{11}{3}$	$\frac{1}{2}\ln\frac{49}{17}$

You can use the space below to show your work:

B2 (4 points)

What is the final classifier produced by these three rounds of Adaboost?

$$\mathrm{H(x)} = sign\left[\frac{1}{2}\ln 7 * d(x) + \frac{1}{2}\ln\frac{11}{3} * j(x) + \frac{1}{2}\ln\frac{49}{17} * i(x)\right]$$

B3 (3 points)

Does your classifier correctly classify all of the training data? Circle one:

YES **NO**

If you circled NO, list any training data points that your Adaboost classifier misclassifies:

#6: Fran

B4 (2 points)

Below is the description of Brittney's ideal character. According to your classifier, should her character be an elf?

	Elf	Fighting Style	Ears	Magic	Size
Brittney	??	Melee	Pointy	Yes	Medium

Circle one:

ELF **NOT AN ELF**

Problem 3: Representation (10 points)

Fill in the missing items in table in the most reasonable way using the transition vocabulary:

Appear, Disappear, Change (Δ), Increase (↑), Decrease (↓), not appear, not disappear, not change, not increase, and not decrease,

Assume the table is to represent aspects of the meaning found in the following. Ignore the black cells in the table. Assume time increases from left to right.

Patrick and Karen were standing together when Patrick started running. Karen chased him, caught him, stopped him, and hit him with a club.

	Interval 1	Interval 2	Interval 3	Interval 4	Interval 5
Patrick's Speed	**A or ↑**	Not Δ	Not Δ	**D**	Not A
Karen's Speed	Not A	A	Not Δ		**Not A or Not Δ ***
Distance between K and P			↓	D	Not A
Patrick's health	Not Δ	Not Δ	Not Δ	Not Δ	**↓ or Δ or D**

* We also accepted D with an explanation.