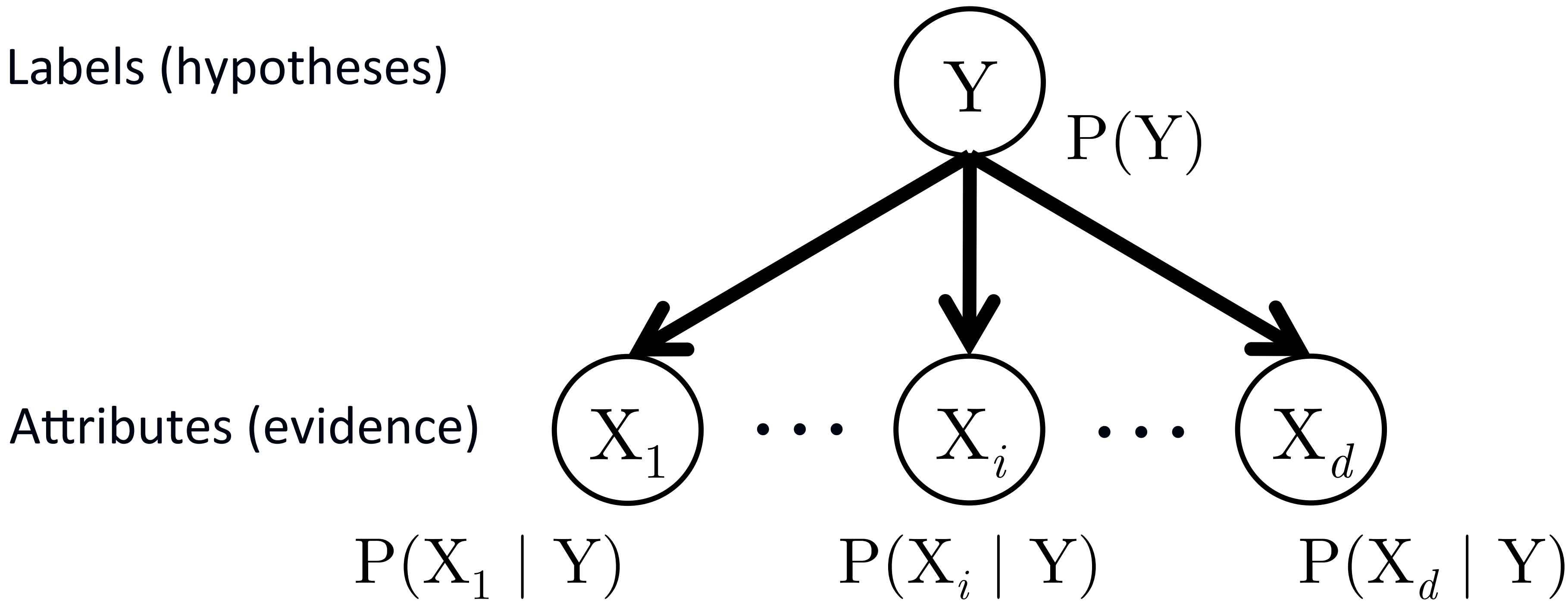
ML Applications: Text Classification

Naïve Bayes Review

The Naïve Bayes Graphical Model



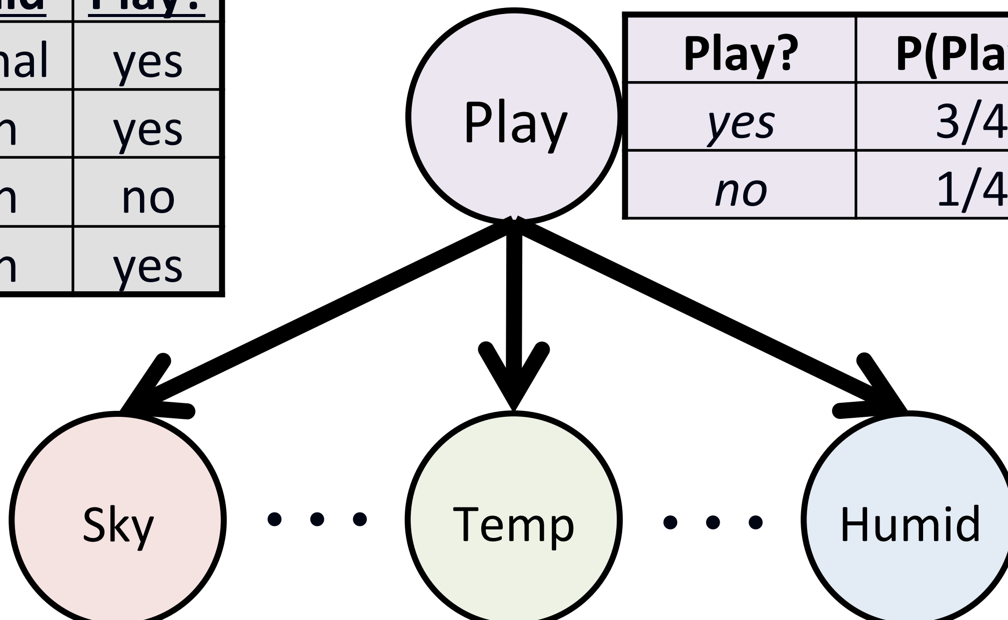
- CPTs are estimated via counting
- Laplace smoothing eliminates zero counts:

$$P(X_j = v \mid Y = y_k) = \frac{1 + \mathrm{c}_v}{K + \sum_{v' \in \mathrm{values}(X_j)} \mathrm{c}_{v'}}$$

Example NB Graphical Model

Data:

Sky	Temp	Humid	Play?
sunny	warm	normal	yes
sunny	warm	high	yes
rainy	cold	high	no
sunny	warm	high	yes



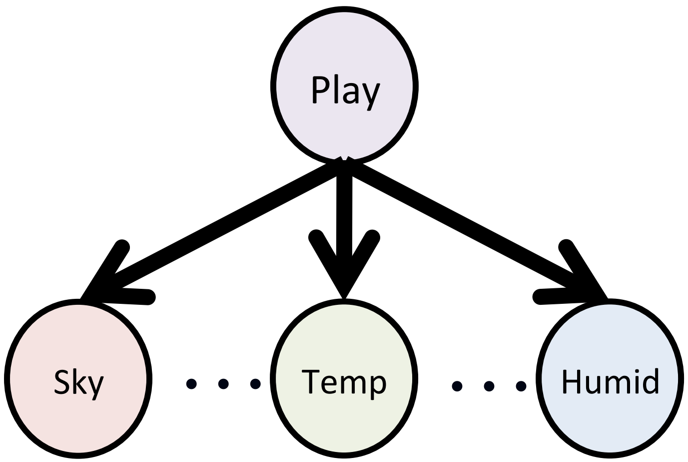
Play?	P(Play)
yes	3/4
no	1/4

Sky	Play?	P(Sky | Play)
sunny	*yes*	4/5
rainy	*yes*	1/5
sunny	*no*	1/3
rainy	*no*	2/3

Temp	Play?	P(Temp | Play)
warm	*yes*	4/5
cold	*yes*	1/5
warm	*no*	1/3
cold	*no*	2/3

Humid	Play?	P(Humid | Play)
high	*yes*	3/5
norm	*yes*	2/5
high	*no*	2/3
norm	*no*	1/3

Example Using NB for Classification



Play?	P(Play)
yes	3/4
no	1/4

Temp	Play?	P(Temp | Play)
warm	*yes*	4/5
cold	*yes*	1/5
warm	*no*	1/3
cold	*no*	2/3

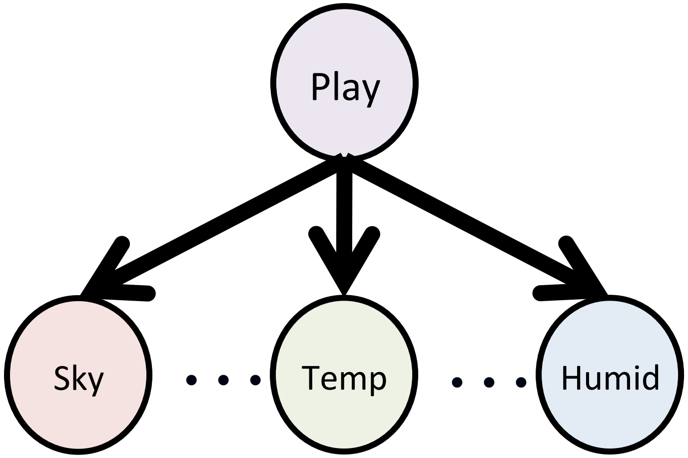
Sky	Play?	P(Sky | Play)
sunny	*yes*	4/5
rainy	*yes*	1/5
sunny	*no*	1/3
rainy	*no*	2/3

Humid	Play?	P(Humid | Play)
high	*yes*	3/5
norm	*yes*	2/5
high	*no*	2/3
norm	*no*	1/3

$$h(\mathbf{x}) = \arg\max_{y_k} \; \log P(Y = y_k) + \sum_{j=1}^{d} \log P(X_j = x_j \mid Y = y_k)$$

Goal: Predict label for $\mathbf{x}$ = (rainy, warm, normal)

Example Using NB for Classification



Play?	P(Play)
yes	3/4
no	1/4

Temp	Play?	P(Temp | Play)
warm	*yes*	4/5
cold	*yes*	1/5
warm	*no*	1/3
cold	*no*	2/3

Sky	Play?	P(Sky | Play)
sunny	*yes*	4/5
rainy	*yes*	1/5
sunny	*no*	1/3
rainy	*no*	2/3

Humid	Play?	P(Humid | Play)
high	*yes*	3/5
norm	*yes*	2/5
high	*no*	2/3
norm	*no*	1/3

Predict label for:

$\mathbf{x}$ = (rainy, warm, normal)

$$P(\text{play} \mid \mathbf{x}) \propto \log P(\text{play}) + \log P(\text{rainy} \mid \text{play}) + \log P(\text{warm} \mid \text{play}) + \log P(\text{normal} \mid \text{play})$$

$$\propto \log 3/4 + \log 1/5 + \log 4/5 + \log 2/5 = -1.319$$

predict PLAY

$$P(\neg\text{play} \mid \mathbf{x}) \propto \log P(\neg\text{play}) + \log P(\text{rainy} \mid \neg\text{play}) + \log P(\text{warm} \mid \neg\text{play}) + \log P(\text{normal} \mid \neg\text{play})$$

$$\propto \log 1/4 + \log 2/3 + \log 1/3 + \log 1/3 = -1.732$$

Document Classification

Document Classification

PROBLEM SETTING

Given:

- Representation of a document
- Set of classes 1,...,K

(AI) (Programming) (HCI)

Classes:

(AI)		(Programming)		(HCI)	
ML	Planning	Semantics	Garb.Coll.	Multimedia	GUI

Training Data:

ML	Planning	Semantics	Garb.Coll.	Multimedia	GUI
learning	planning	programming	garbage	...	...
intelligence	temporal	semantics	collection		
algorithm	reasoning	language	memory		
reinforcement	plan	proof...	optimization		
network...	language...		region...		

Based on slide by Chris Manning

Document Classification

Test Data:

"planning language proof intelligence"

PROBLEM SETTING

Given:

- Representation of a document
- Set of classes 1,...,K

Determine:

- Class to which document d belongs

(AI) (Programming) (HCI)

Classes:

ML | Planning | Semantics | Garb.Coll. | Multimedia | GUI

Training Data:

- **ML:** learning <u>intelligence</u> algorithm reinforcement network...
- **Planning:** <u>planning</u> temporal reasoning plan <u>language</u>...
- **Semantics:** programming semantics <u>language</u> <u>proof</u>...
- **Garb.Coll.:** garbage collection memory optimization region...
- **Multimedia:** ...
- **GUI:** ...

Based on slide by Chris Manning

Text Classification: Examples

- Classify news stories as *World, US, Business, SciTech, Sports, etc.*
- Add terms to Medline abstracts (e.g. "Conscious Sedation" [E03.250])
- Classify business names by industry
- Classify student essays as *A/B/C/D/F*
- Classify email as *Spam/Other*
- Classify email to tech staff as *Mac/Windows/ ...*
- Classify pdf files as *ResearchPaper/Other*
- Determine authorship of documents
- Classify movie reviews as *Favorable/Unfavorable/Neutral*
- Classify technical papers as *Interesting/Uninteresting*
- Classify jokes as *Funny/NotFunny*
- Classify websites of companies by Standard Industrial Classification (SIC) code

Based on slide by William Cohen

Text Classification: Examples

- Best-studied benchmark: *Reuters-21578* newswire stories
  - 9603 train, 3299 test documents, 80-100 words each, 93 classes

ARGENTINE 1986/87 GRAIN/OILSEED REGISTRATIONS

BUENOS AIRES, Feb 26

Argentine grain board figures show crop registrations of grains, oilseeds and their products to February 11, in thousands of tonnes, showing those for future shipments month, 1986/87 total and 1985/86 total to February 12, 1986, in brackets:

- Bread wheat prev 1,655.8, Feb 872.0, March 164.6, total 2,692.4 (4,161.0).
- Maize Mar 48.0, total 48.0 (nil).
- Sorghum nil (nil)
- Oilseed export registrations were:
- Sunflowerseed total 15.0 (7.9)
- Soybean May 20.0, total 20.0 (nil)

The board also detailed export registrations for subproducts, as follows....

→ Categories: grain, wheat (of 93 binary choices)

Slide by William Cohen

Document Retrieval

Google

UPenn Machine Learning

Web News Images Videos Shopping More Search tools

About 244,000 results (0.25 seconds)

CIS520 Machine Learning - fling.seas.upenn.edu
https://alliance.sea... University of Pennsylvania School of Engineering an...
Aug 26, 2014 - Welcome to CIS520: **Machine Learning**. Lectures: Wu & Chen Auditorium, MW 10:30–12:00, F 9:30am-11:00pm (Please bring Turningpoint ...

PRiML.upenn: Penn Research in Machine Learning - Home ...
priml.upenn.edu/
Monday, April 1st at 1:30pm, Nina Balcan is giving a talk "Learning Valuation ... place best paper award at the 6th Annual **Machine Learning** Symposium at the ...

CIS520 Machine Learning | Lectures / Lectures BrowseTitle
https://alliance.sea... University of Pennsylvania School of Engineering an...
40+ items - Date, Subject, Reading. On your own, learn linear algebra, Self ...

Date	Subject	Reading.
F/Aug 29	Probability Review slides	Bishop 1.1-1.4.
W/Sep 3	Nearest Neighbor	Bishop 2.5.

Machine learning courses at Penn
www.cis.upenn.edu/~ungar/ml-courses.html University of Pennsylvania
CIS520 - **Machine Learning**. Ben Taskar or Lyle Ungar This is the course to take on macahine learning. Not easy. CIS521 - Artificial Intelligence Standard intro to ...

CIS 419/519 Introduction to Machine Learning - Fall 2014
www.cis.upenn.edu/~cis519/fall2014/ University of Pennsylvania
... on the day listed. The readings will come from **Machine Learning** (Flach), Learning from Data (LfD), the reading packet (Handout), or online sources.

CIS 520 - Machine Learning - Fall 2010 - SEAS
https://www.seas.... University of Pennsylvania School of Engineering an...

Spam Filtering

From: "" <takworlld@hotmail.com>
Subject: real estate is the only way... gem oalvgkay

Anyone can buy real estate with no money down

Stop paying rent TODAY !

There is no need to spend hundreds or even thousands for similar courses

I am 22 years old and I have already purchased 6 properties using the methods outlined in this truly INCREDIBLE ebook.

Change your life NOW !

=======================================
Click Below to order:
http://www.wholesaledaily.com/sales/nmd.htm
=======================================

Bag of Words Representation

What is the ~~best~~ representation for documents?

simplest, yet useful

Idea: Treat each document as a sequence of words

- Assume that word positions are generated *independently*

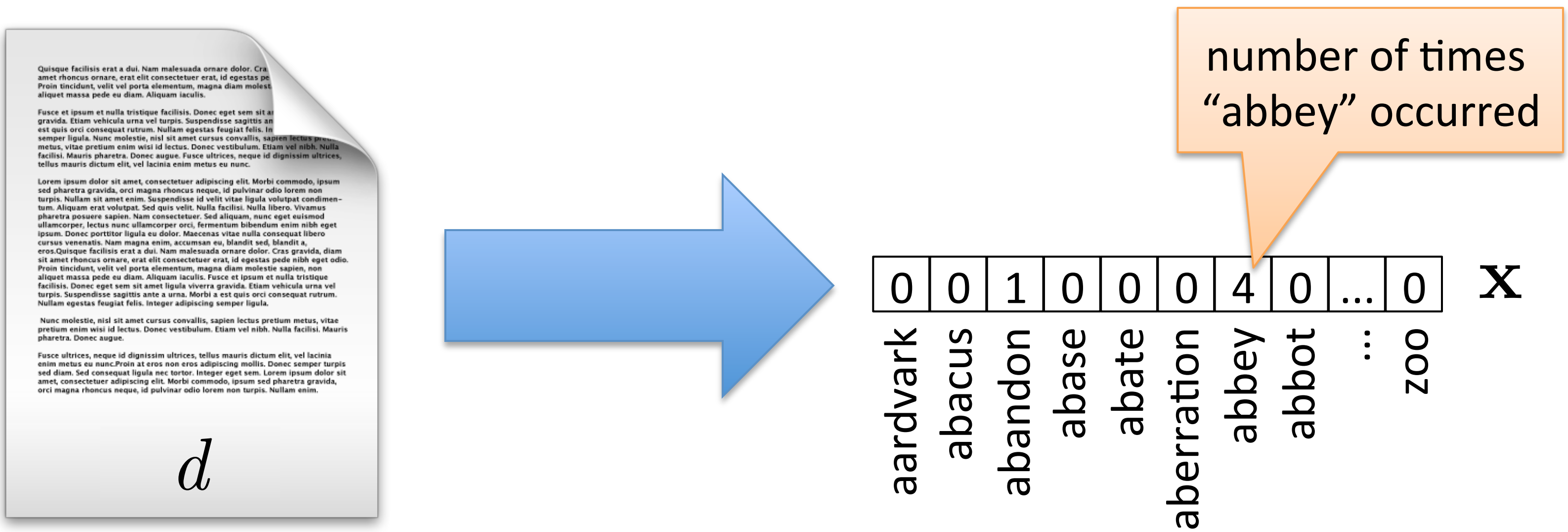
<u>Dictionary</u>: set of all possible words

- Compute over set of documents
- Use Webster's dictionary, etc.

Bag of Words Representation

Represent document d as a vector of word counts $\mathbf{x}$

- x_j represents the count of word j in the document
  - $\mathbf{x}$ is sparse (few non-zero entries)



Another View of Naïve Bayes For Document Classification

- Let the model parameters for class c be given by:

$$\boldsymbol{\theta}_c = \{\theta_{c1}, \theta_{c2}, \ldots, \theta_{c|D|}\}$$

size of dictionary D

  - $\theta_{cj} = \mathrm{P}($word j occurs in a document from $c)$
  - Also have that $\sum_j \theta_{cj} = 1$

- The likelihood of a document d characterized by $\mathbf{x}$ is

$$P(d \mid \boldsymbol{\theta}_c) = \frac{(\sum_j x_j)!}{\prod_j x_j!} \prod_j (\theta_{cj})^{x_j}$$

  - This is just the multinomial distribution, a generalization of the binomial distribution $\binom{n}{k} p^k (1-p)^{n-k}$

Another View of Naïve Bayes For Document Classification

- The likelihood of a document d characterized by $\mathbf{x}$ is

$$P(d \mid \boldsymbol{\theta}_c) = \frac{(\sum_j x_j)!}{\prod_j x_j!} \prod_j (\theta_{cj})^{x_j}$$

- Use Bayes rule:

introduce class priors

$$\log P(\boldsymbol{\theta}_c \mid d) \propto \log \left(P(\boldsymbol{\theta}_c) \prod_{j=1}^{|D|} (\theta_{cj})^{x_j} \right) = \log P(\boldsymbol{\theta}_c) + \sum_{j=1}^{|D|} x_j \log \theta_{cj}$$

Therefore,

$$h(d) = \arg\max_c \left(\log P(\boldsymbol{\theta}_c) + \sum_{j=1}^{|D|} x_j \log \theta_{cj} \right)$$

This is just a linear decision function!

Document Classification with Naïve Bayes

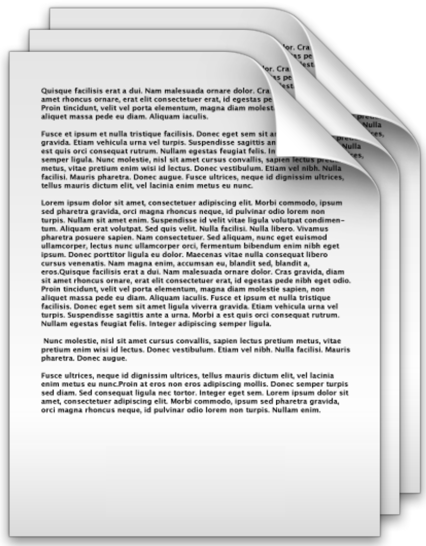
1. Compute dictionary D over training set (if not given)
2. Represent training documents as bags of words over D
3. Estimate class priors via counting
4. Estimate conditional probabilities as $\hat{\theta}_{cj} = \dfrac{N_{cj} + 1}{N_c + |D|}$
   - N_{cj} is number of times word j occurs in documents from class c
   - N_c is total number of words in all documents from class c

- Naïve Bayes model for new documents (represented in D) is:

$$h(d) = \arg\max_c \left(\log P(c) + \sum_j x_j \hat{w}_{cj} \right)$$

$$\text{where } \hat{w}_{cj} = \log \hat{\theta}_{cj}$$

What are Some Issues with the Bag of Words Representation?



- Documents have different lengths
- Some words aren't meaningful to represent the content of a document
  - e.g., "the", "a", etc.
- Rare words may be more meaningful than common words

Need a better representation for the documents...

Eliminate Stop Words

Common, "less-meaningful" words are called <u>stop words</u>

- Delete stop words before doing any document processing

Example stop words:

a	because	does	haven't	i	more	our	some	they'll	we'll	why
about	been	doesn't	having	i'd	most	ours	such	they're	we're	why's
above	before	doing	he	i'll	mustn't	ourselves	than	they've	we've	with
after	being	don't	he'd	i'm	my	out	that	this	were	won't
again	below	down	he'll	i've	myself	over	that's	those	weren't	would
against	between	during	he's	if	no	own	the	through	what	wouldn't
all	both	each	her	in	nor	same	their	to	what's	you
am	but	few	here	into	not	shan't	theirs	too	when	you'd
an	by	for	here's	is	of	she	them	under	when's	you'll
and	can't	from	hers	isn't	off	she'd	themselves	until	where	you're
any	cannot	further	herself	it	on	she'll	then	up	where's	you've
are	could	had	him	it's	once	she's	there	very	which	your
aren't	couldn't	hadn't	himself	its	only	should	there's	was	while	yours
as	did	has	his	itself	or	shouldn't	these	wasn't	who	yourself
at	didn't	hasn't	how	let's	other	so	they	we	who's	yourselves
be	do	have	how's	me	ought		they'd	we'd	whom	

Term Frequency

Term frequency $tf_{t,d}$ is some measure of importance of term t to document d

Boolean: $tf_{t,d} = 1$ if t occurs in d, 0 otherwise

Raw Counts: $tf_{t,d} = c_{t,d}$

– $c_{t,d}$ is the number of times t occurs in d

Log-Scaled Counts: $tf_{t,d} = \begin{cases} 1 + \log c_{t,d} & \text{if } c_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$

– Reduces relative impact of frequent terms

Normalized Counts: $tf_{t,d} = c_{t,d} / |d|$

– Normalize raw counts by length of document $|d|$

Inverse Document Frequency

Idea: rare terms are more important than common terms

Example: if all training documents for a class contain

- the (relatively) common word "water", and
- the (relatively) rare word "hippopotamus",
- the term "hippopotamus" is likely more important

Inverse Document Frequency

$$idf_{t,X} = \log\left(\frac{|X|}{|X_t| + 1}\right)$$

- X is the total set of documents
- X_t is the subset of documents containing term t

TF-IDF Transform

- To compensate for issues with raw word counts, use TF-IDF transform on the features with naïve Bayes

$$tfidf_{t,d,X} = tf_{t,d} \times idf_{t,X}$$

  - Represent document as a vector $\mathbf{x}$ of TF-IDF features
  - x_j represents the TF-IDF of word j in the document

Recommendations: (From [Rennie, et al. ICML'03])

  - Use raw counts or log-scaled counts for $tf_{t,d}$
  - Normalize each TF-IDF vector $\mathbf{x}$ to have unit length by $\mathbf{x} = \mathbf{x} \ / \ ||\mathbf{x}||_2$ and use these unit vectors in naïve Bayes

You must use the same TF-IDF transform for new documents!

(For more details, see http://people.csail.mit.edu/jrennie/papers/icml03-nb.pdf)

Using SVMs for Document Classification

Words → Counts → Weight Matrix

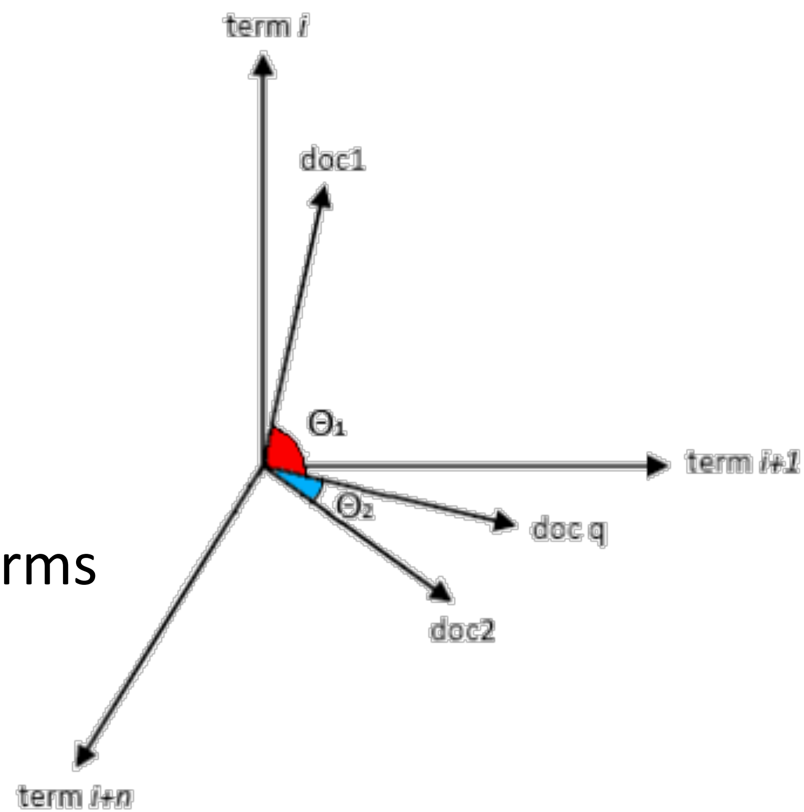
	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

Each document is now represented by a real-valued vector of $|D|$ TF-IDF weights

Based on slide by P. Nayak and P. Raghavan

Documents as Vectors

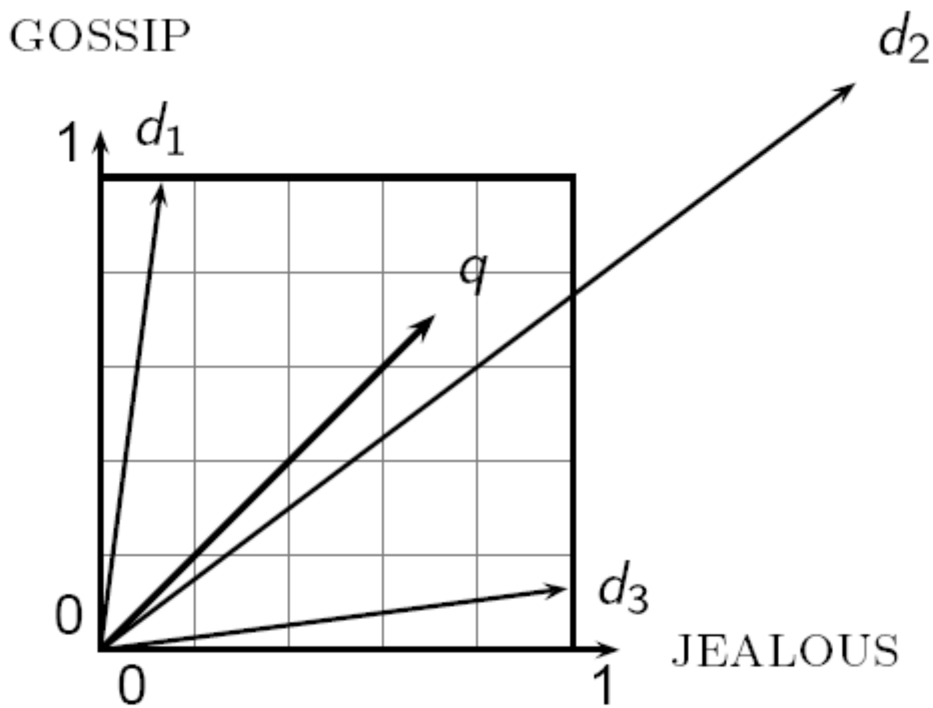
- So we have a $|D|$-dimensional vector space
  - Terms are axes of the space
  - Documents are points or vectors in this space



- Very high-dimensional:
  - Over 1M words in english
  - More if we allow non-word terms
- Very sparse vectors
- **Idea:** Measure similarity of documents via proximity in the vector space

Why Euclidean Distance is a Bad Idea

- Because Euclidean distance is large for vectors of different lengths



$||\mathbf{q} - \mathbf{d}_2||_2$ is large, even though the distribution of terms in the query $\mathbf{q}$ and the distribution of terms in the document $\mathbf{d}_2$ are very similar

Use Angle Instead of Distance

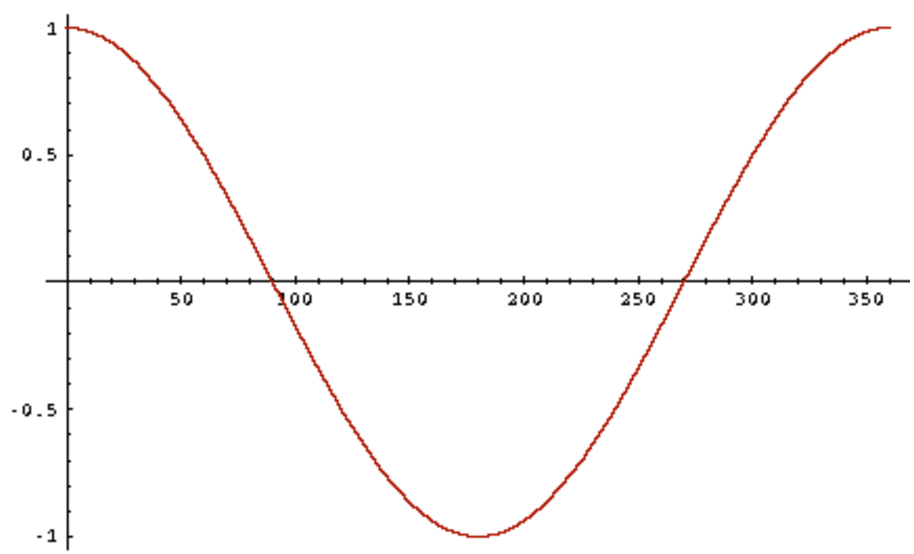
Thought experiment:

- Take a document d and append it to itself, creating a new document d'
- Semantically, d and d' have the same content
- But, the Euclidean distance between the two documents can be quite large
- However, note that the angle between the two documents is 0, corresponding to maximal similarity

Key Idea: Measure similarity based on angle of vector

From Angles to Cosines

- The following two notions are equivalent:
  - Measure similarity between documents d_i and d_j via decreasing order of the angle between $\mathbf{x}_i$ and $\mathbf{x}_j$
  - Measure similarity in increasing order of cosine($\mathbf{x}_i$, $\mathbf{x}_j$)
- Cosine is a monotonically decreasing function for the interval [0^o, 180^o]



Based on slide by P. Nayak and P. Raghavan

Length Normalization

- A vector can be (length-) normalized by dividing each of its components by its length (the L_2 norm)

$$\mathbf{x} = \mathbf{x} \;/\; ||\mathbf{x}||_2$$

- Dividing a vector by its L_2 norm makes it a unit (length) vector (on surface of unit hypersphere)

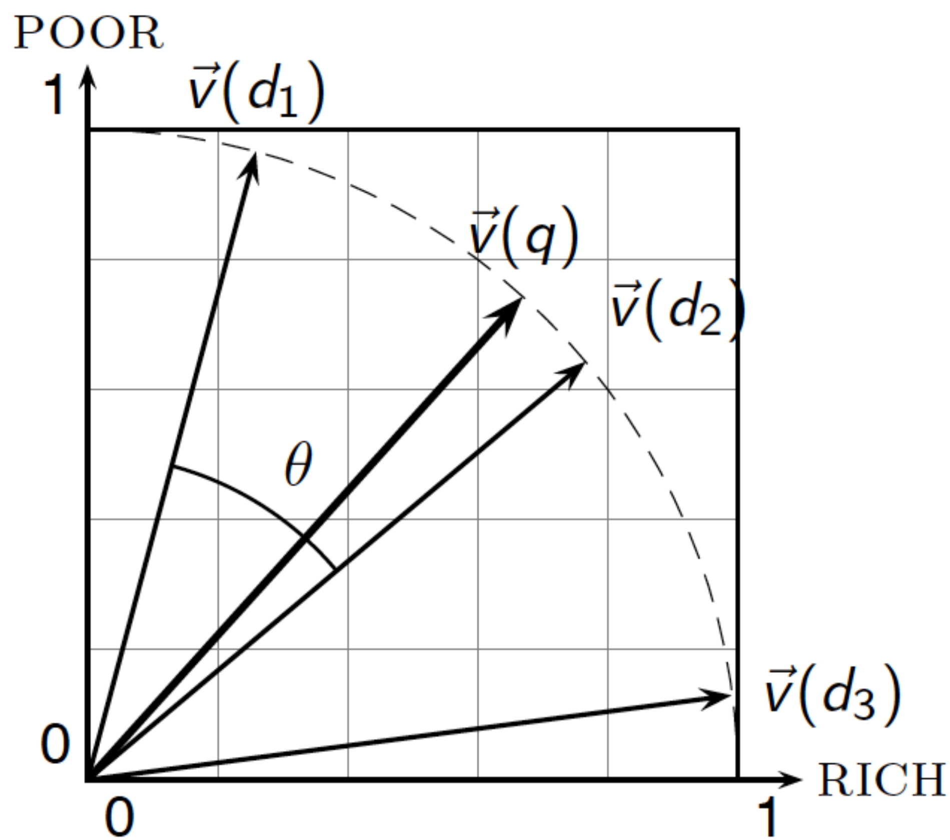
- Effect on the two documents d and d‘ (d appended to itself) from earlier slide: they have identical vectors after length-normalization
  - Long and short documents now have comparable weights

Cosine Similarity

$\mathbf{x}_i$ and $\mathbf{x}_j$ are TF-IDF weight vectors

$$\cos(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{|\mathbf{x}_i||\mathbf{x}_j|} = \underbrace{\frac{\mathbf{x}_i}{|\mathbf{x}_i|}}_{} \cdot \underbrace{\frac{\mathbf{x}_j}{|\mathbf{x}_j|}}_{}$$

unit-length vectors



$\cos(\mathbf{x}_i, \mathbf{x}_j)$ is the cosine similarity of $\mathbf{x}_i$ and $\mathbf{x}_j$

- Equivalently, the cosine of the angle between $\mathbf{x}_i$ and $\mathbf{x}_j$
- For unit vectors, cosine similarity is simply the dot product

Example: Cosine Similarity Amongst Three Documents

Term Frequencies (counts)

term	SaS	PaP	WH
affection	115	58	20
jealous	10	7	11
gossip	2	0	6
wuthering	0	0	38

SaS: *Sense and Sensibility*

PaP: *Pride and Prejudice*

WH: *Wuthering Heights*?

Note: To simplify this example, we don't do IDF weighting

Example: Cosine Similarity Amongst Three Documents

Log-Scaled Counts

term	SaS	PaP	WH
affection	3.06	2.76	2.30
jealous	2.00	1.85	2.04
gossip	1.30	0	1.78
wuthering	0	0	2.58

After Length Normalization

term	SaS	PaP	WH
affection	0.789	0.832	0.524
jealous	0.515	0.555	0.465
gossip	0.335	0	0.405
wuthering	0	0	0.588

$\cos(\text{SaS},\text{PaP}) \approx 0.789 \times 0.832 + 0.515 \times 0.555 + 0.335 \times 0.0 + 0.0 \times 0.0$
≈ 0.94

$\cos(\text{SaS},\text{WH}) \approx 0.79$

$\cos(\text{PaP},\text{WH}) \approx 0.69$

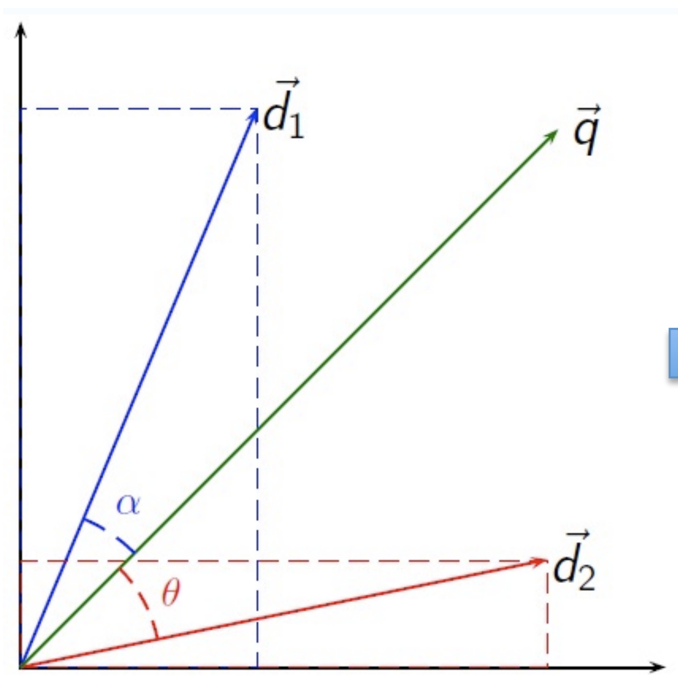
Why is cos(SaS,PaP) > cos(SaS,WH)?

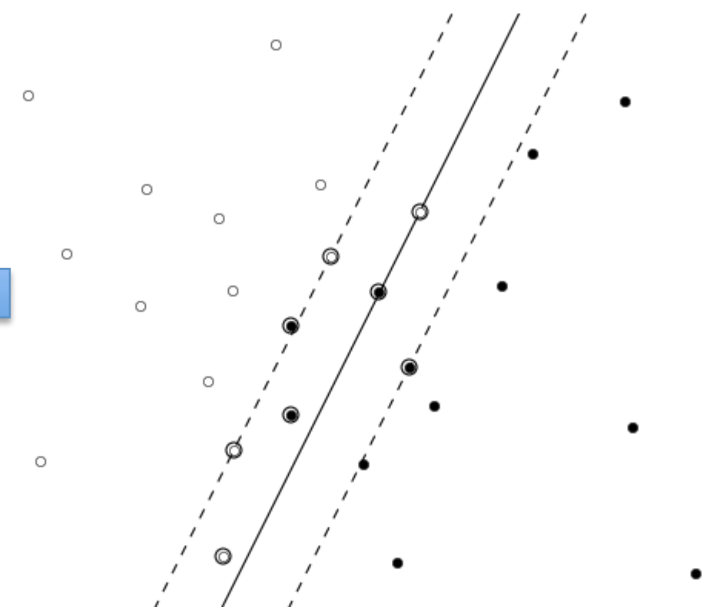
Based on slide by P. Nayak and P. Raghavan

SVMs for Text Classification

- Use the cosine similarity kernel on TF-IDF features

$$K(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i^\intercal \mathbf{x}_j}{\|\mathbf{x}_i\| \; \|\mathbf{x}_j\|}$$





Advanced Evaluation Metrics

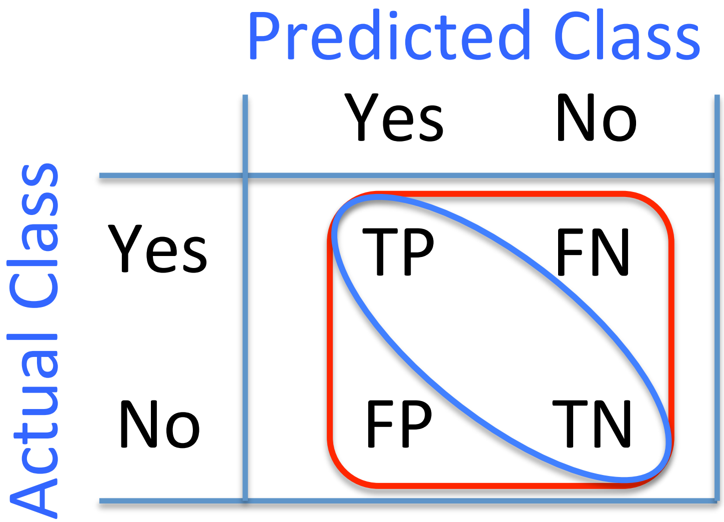
Confusion Matrix

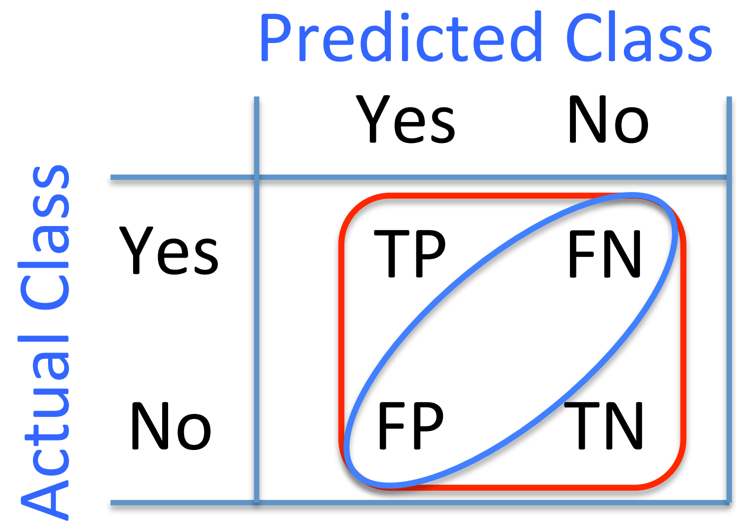
Given a dataset of P positive instances and N negative instances:

		Predicted Class	
		Yes	No
Actual Class	Yes	TP	FN
	No	FP	TN

Accuracy & Error

Given a dataset of P positive instances and N negative instances:





$$\text{accuracy} = \frac{TP + TN}{P + N}$$

$$\text{error} = 1 - \frac{TP + TN}{P + N} = \frac{FP + FN}{P + N}$$

Why Not Just Use Accuracy?

- How to build a 99.9999% accurate search engine on a low budget….

- Users doing information retrieval *want to find something* and have a certain tolerance for junk

Precision & Recall

Precision

- the fraction of positive predictions that are correct
- P(is pos|predicted pos)

$$\text{precision} = \frac{TP}{TP + FP}$$

Predicted Class

Actual Class		Yes	No
	Yes	TP	FN
	No	FP	TN

Recall

- fraction of positive instances that are identified
- P(predicted pos|is pos)

$$\text{recall} = \frac{TP}{TP + FN}$$

Predicted Class

Actual Class		Yes	No
	Yes	TP	FN
	No	FP	TN

Precision & Recall

Precision

- the fraction of positive predictions that are correct
- P(is pos|predicted pos)

$$\text{precision} = \frac{TP}{TP + FP}$$

Recall

- fraction of positive instances that are identified
- P(predicted pos|is pos)

$$\text{recall} = \frac{TP}{TP + FN}$$

- You can get high recall (but low precision) by only predicting positive
- Recall is a non-decreasing function of the # positive predictions
- Typically, precision decreases as either the number of positive predictions or recall increases
- Precision & recall are widely used in information retrieval

F-Measure

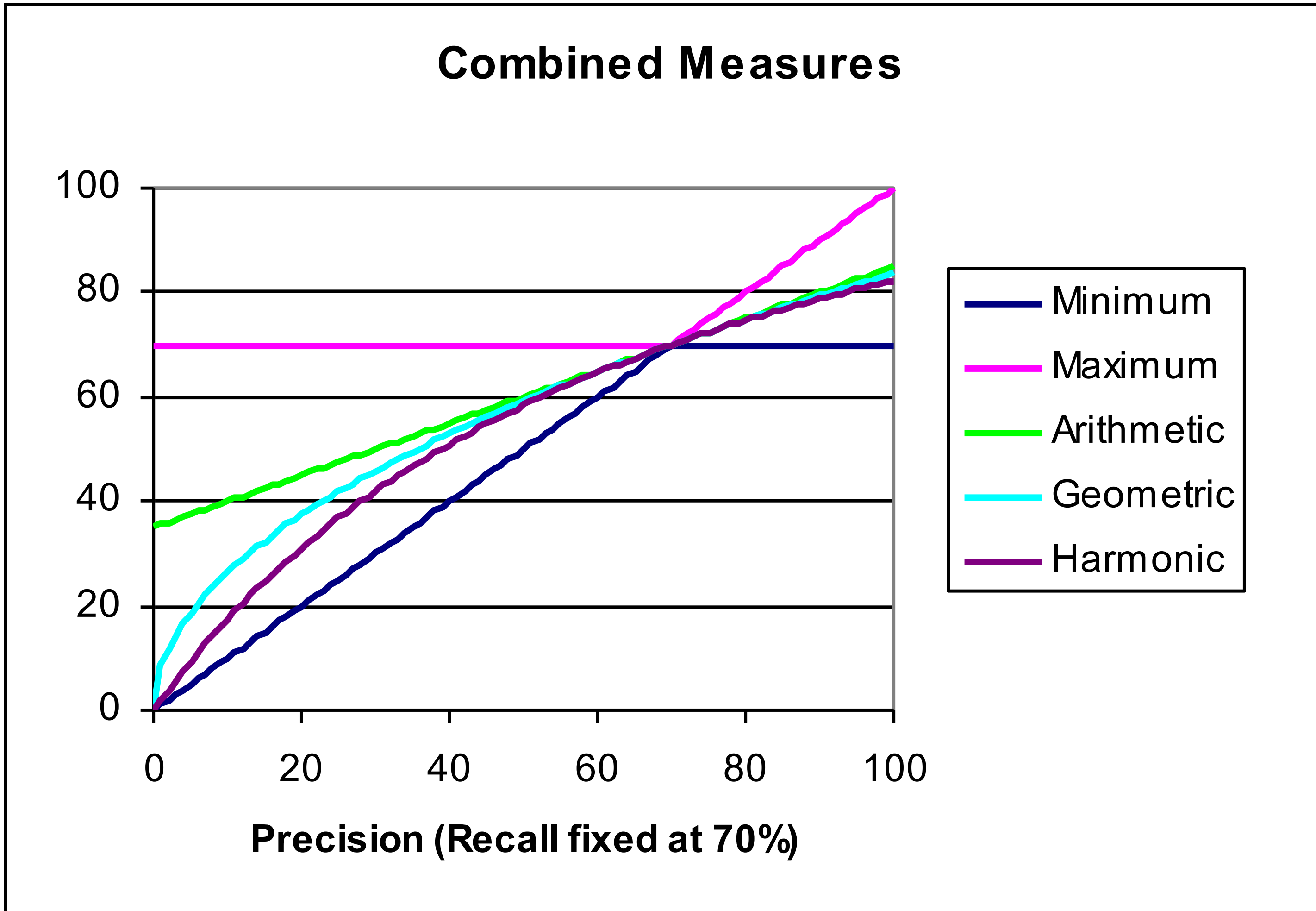
- Combined measure of precision/recall tradeoff

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

  - This is the harmonic mean of precision and recall
  - In the F_1 measure, precision and recall are weighted evenly
  - Can also have biased weightings that emphasize either precision or recall more (F_2 = 2 × recall; $F_{0.5}$ = 2 × precision)

- Limitations:
  - F-measure can exaggerate performance if balance between precision and recall is incorrect for application
    - Don't typically know balance ahead of time

F_1 and Other Averages



A Word of Caution

- Consider binary classifiers A, B, C:

		A	.	B	.	C	.
		1	0	1	0	1	0
Predictions	1	0.9	0.1	0.8	0	0.78	0
	0	0	0	0.1	0.1	0.12	0.1

- Clearly A is useless, since it always predicts 1
- B is slightly better than C
  - less probability mass wasted on the off-diagonals
- But, here are the performance metrics:

Metric	A	B	C
Accuracy	0.9	0.9	0.88
Precision	0.9	1.0	1.0
Recall	1.0	0.888	0.8667
F-score	0.947	0.941	0.9286

Slide by Kevin Murphy

Receiver Operating Characteristic (ROC)

ROC curves assess predictive behavior independent of error costs or class distributions

- Originated from signal detection theory
- Common in medical diagnosis, now used for ML

Plots TP rate vs FP Rate

$$\text{TP rate} = \text{TP}/\text{P}$$

$$\text{FP rate} = \text{FP}/\text{N}$$

Predicted Class

Actual Class	Yes	No
Yes	TP	FN
No	FP	TN

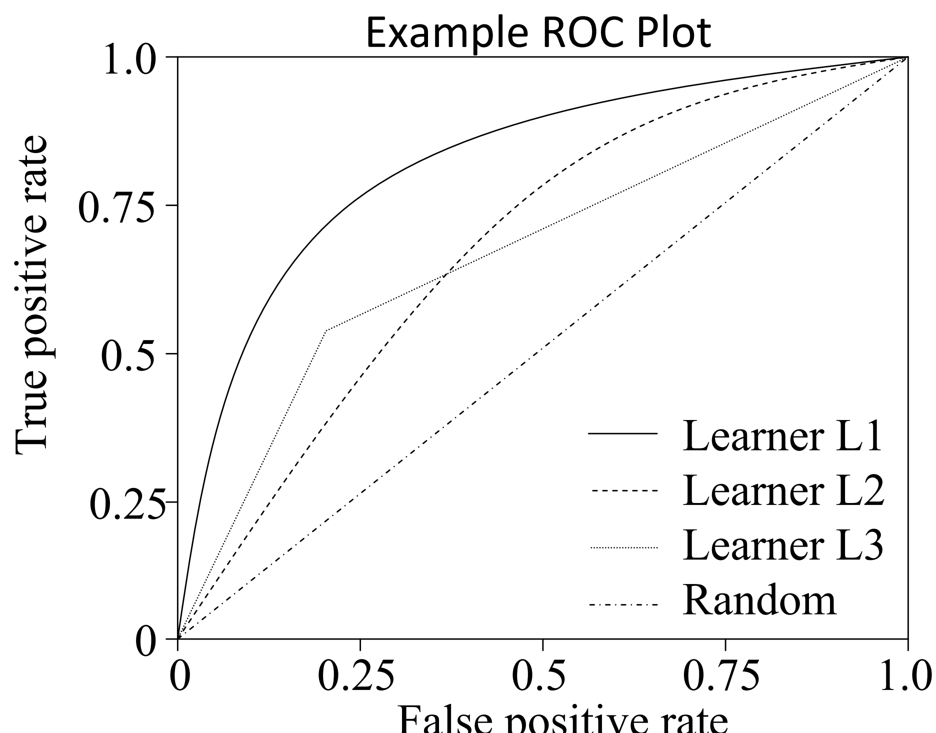
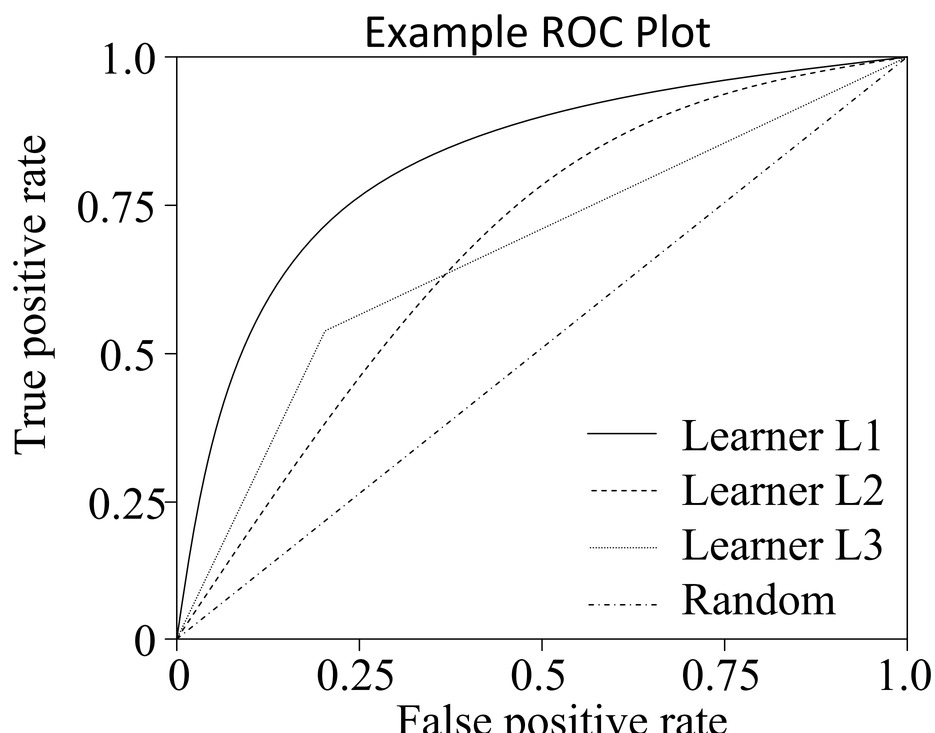


Figure and some text provided by Larry Holder

Performance Depends on Threshold

Predict positive if $\mathrm{P}(y = 1 \mid \mathbf{x}) > \theta$, otherwise negative

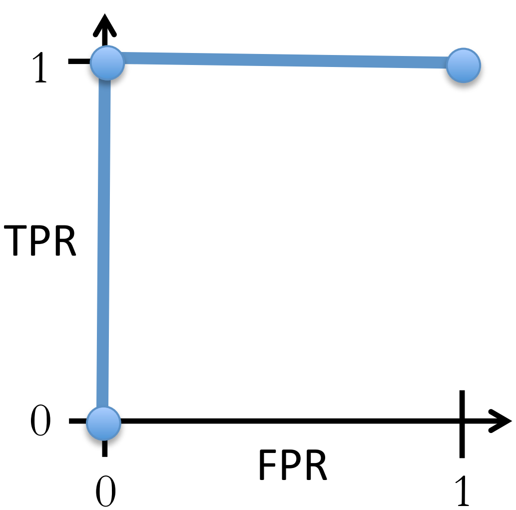
- Number of TPs and FPs depend on threshold θ
- As we vary θ, we get different (TPR, FPR) points



Based on slide by Kevin Murphy, Figure by Larry Holder

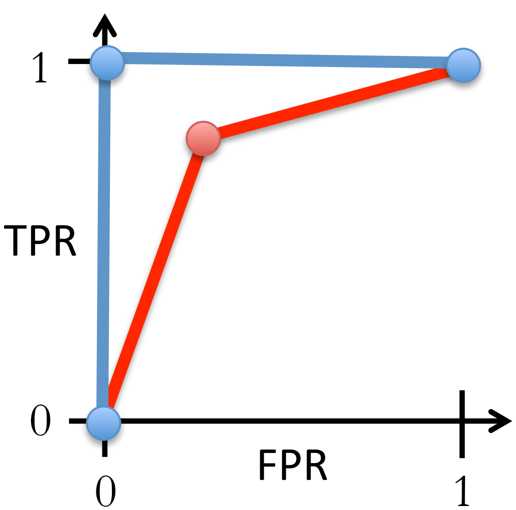
ROC Example

i	y_i	$p(y_i = 1 \mid \mathbf{x}_i)$	$h(\mathbf{x_i} \mid \theta = 0)$	$h(\mathbf{x_i} \mid \theta = 0.5)$	$h(\mathbf{x_i} \mid \theta = 1)$
1	1	0.9	1	1	0
2	1	0.8	1	1	0
3	1	0.7	1	1	0
4	1	0.6	1	1	0
5	1	0.5	1	1	0
6	0	0.4	1	0	0
7	0	0.3	1	0	0
8	0	0.2	1	0	0
9	0	0.1	1	0	0
			$TPR = 5/5 = 1$ $FPR = 4/4 = 1$	$TPR = 5/5 = 1$ $FPR = 0/4 = 0$	$TPR = 0/5 = 0$ $FPR = 0/4 = 0$



ROC Example

i	y_i	$p(y_i = 1 \mid \mathbf{x}_i)$	$h(\mathbf{x_i} \mid \theta = 0)$	$h(\mathbf{x_i} \mid \theta = 0.5)$	$h(\mathbf{x_i} \mid \theta = 1)$
1	1	0.9	1	1	0
2	1	0.8	1	1	0
3	1	0.7	1	1	0
4	1	0.6	1	1	0
5	1	**0.2**	1	**0**	0
6	0	**0.6**	1	**1**	0
7	0	0.3	1	0	0
8	0	0.2	1	0	0
9	0	0.1	1	0	0
			$TPR = 5/5 = 1$ $FPR = 4/4 = 1$	$TPR = 4/5 = 0.8$ $FPR = 1/4 = 0.25$	$TPR = 0/5 = 0$ $FPR = 0/4 = 0$



Receiver Operating Characteristic (ROC)

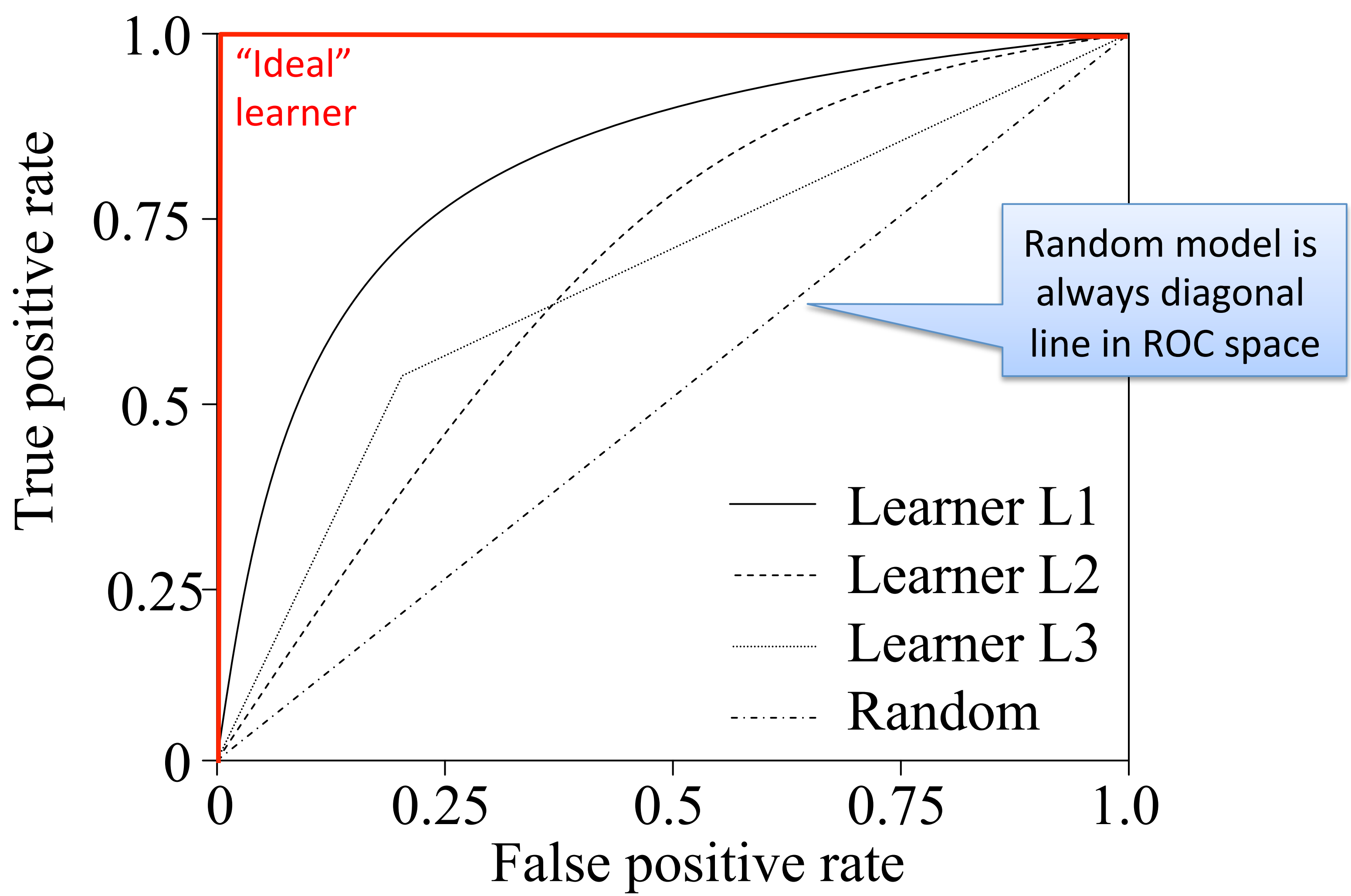


Figure by Larry Holder

Receiver Operating Characteristic (ROC)

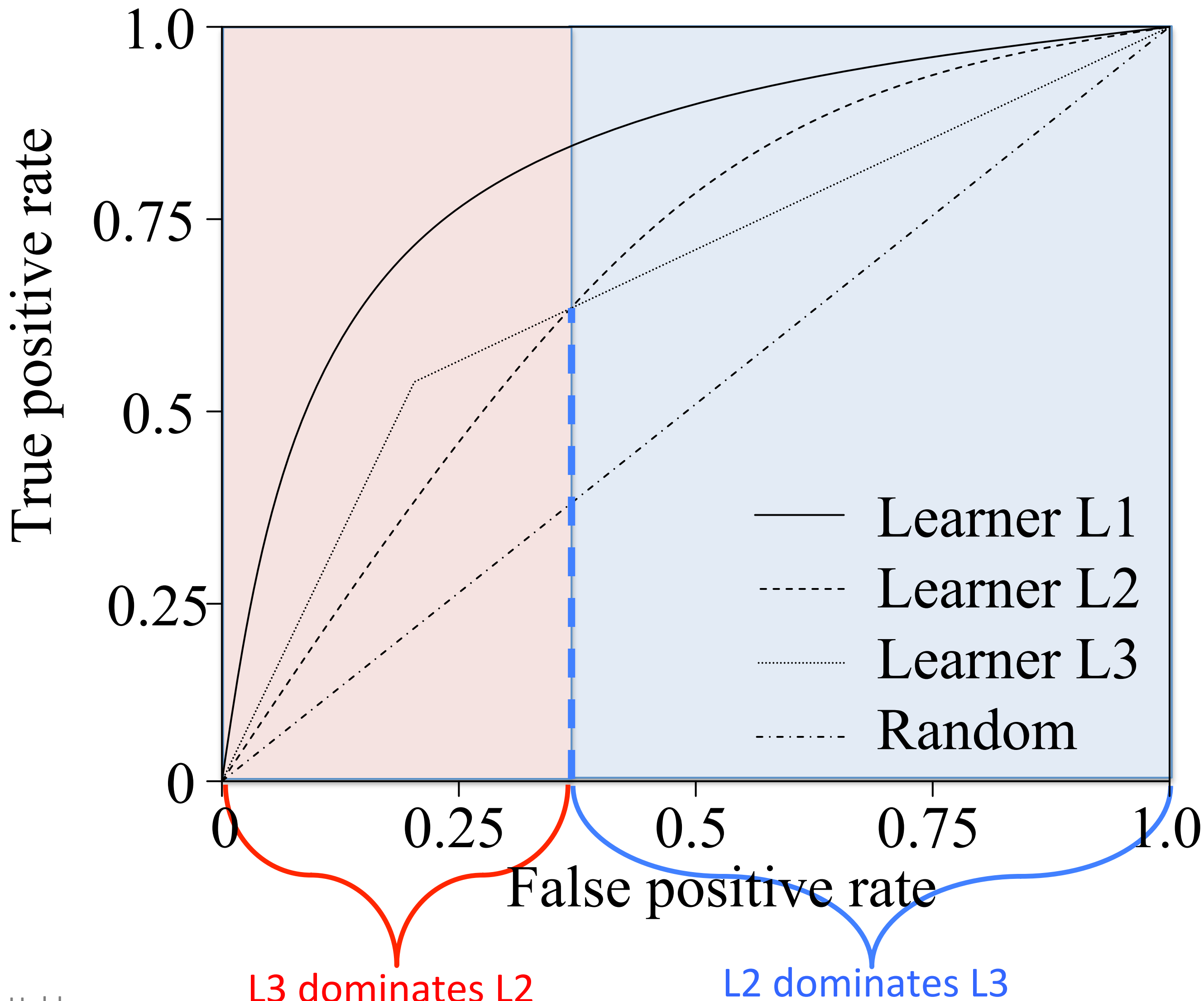


Figure by Larry Holder

Receiver Operating Characteristic (ROC)

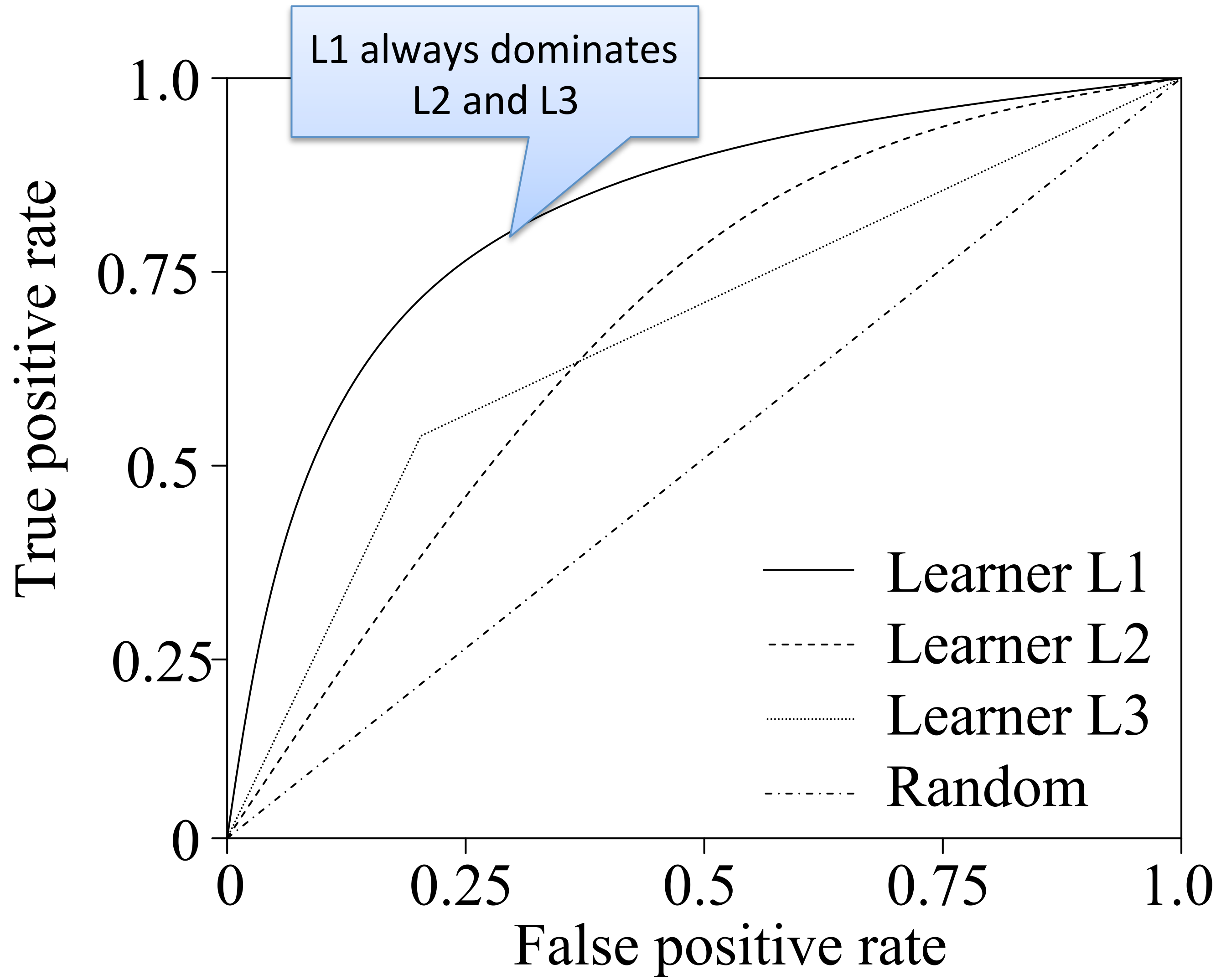
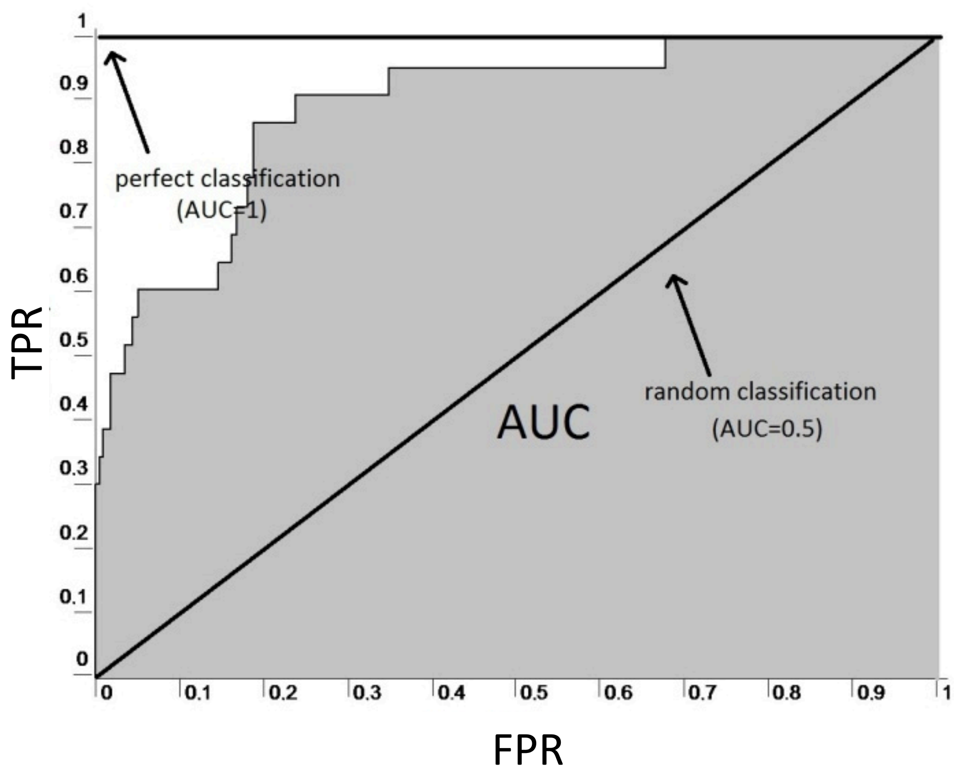
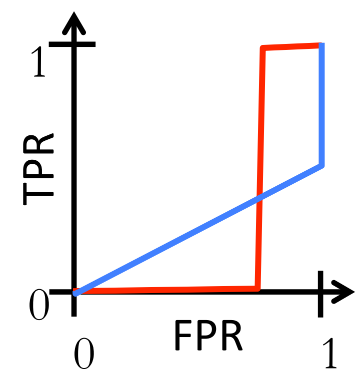


Figure by Larry Holder

Area Under the ROC Curve

- Can take area under the ROC curve to summarize performance as a single number
  - Be cautious when you see only AUC reported without a ROC curve; AUC can hide performance issues





Same AUC, very different performance