Machine Learning

Computer Sciences 760

Spring 2017

www.cs.wisc.edu/~dpage/cs760/

CS 760: Machine Learning

- Professor: David Page

  email: page@biostat.wisc.edu, put “760” in subject for any email

  office hours: 1pm-2pm TR in 1153 WID, starting tomorrow

  other times by appointment in 3174 WID

- TAs:
  - Heemanshu Suri, hsuri@wisc.edu
  - Kirthanaa Raghuraman, kraghuraman@wisc.edu

Monday, Wednesday *and* Friday?

- We'll have 30 lectures in all, just like a standard TR class
- Most weeks we'll just meet Mon and Wed
- Some weeks we'll meet on Friday
- This arrangement facilitates making up for days I'm out of town
- First three weeks we will meet MWF
- I will give 2 weeks' advance notice for other Fridays that the class meets

Class capacity

- Used to be limited to 30
- Demand has grown well over 200, hence now twice/year
- I’ve allowed 110 to register (room capacity)

Course emphases

- a variety of learning settings: supervised learning, unsupervised learning, reinforcement learning, active learning, etc.
- a broad toolbox of machine-learning methods: decision trees, nearest neighbor, Bayesian networks, SVMs, etc.
- some underlying theory: bias-variance tradeoff, PAC learning, mistake-bound theory, etc.
- experimental methodology for evaluating learning systems: cross validation, ROC and PR curves, hypothesis testing, etc.

Two major goals

1. Understand what a learning system should do

2. Understand how (and how well) existing systems work

Course requirements

- 5 homework assignments: 40%
- midterm exam (late in semester): 35%
- project (groups of 4-5): 25%

Homework 1

- Due in two weeks (2/1)
- Download Weka (it's free; you might already have access; if not, Google it)
- Create a data set of your choosing
  - At least 20 examples (data points)
  - At least 10 features (variables)
- Run decision trees, nearest neighbor, others if you want and submit a PDF report (1 page) at course moodle

Expected background

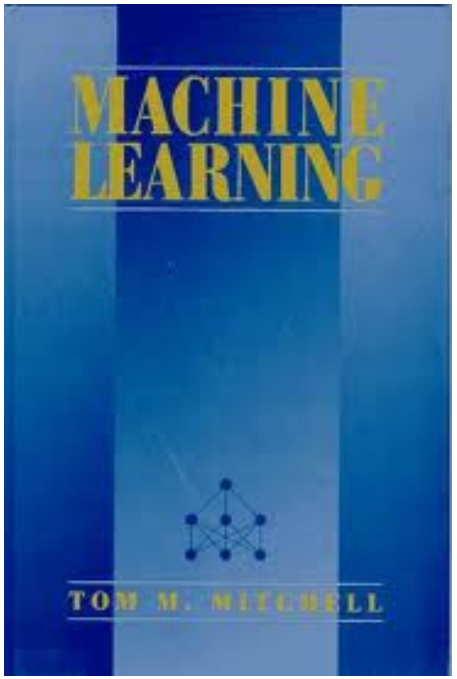
- CS 540 (Intro to Artificial Intelligence) or equivalent
  - search
  - first-order logic
  - unification
  - deduction
- reasonable programming skills
- basics of probability: but we'll review
- linear algebra
  - vectors and matrices
- calculus
  - partial derivatives

Programming languages

- for the programming assignments, you can use
  - C
  - C++
  - Java
  - Perl
  - Python
  - R
- programs must be callable from the command line and *must run on the CS lab machines (this is where they will be tested during grading!)*

Course readings

- *Machine Learning*. T. Mitchell. McGraw Hill, 1997.



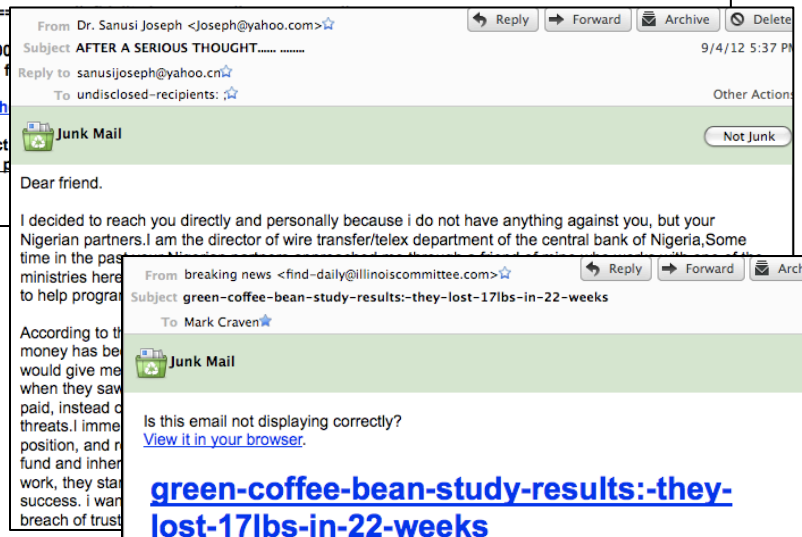
- additional on-line articles, surveys, and chapters

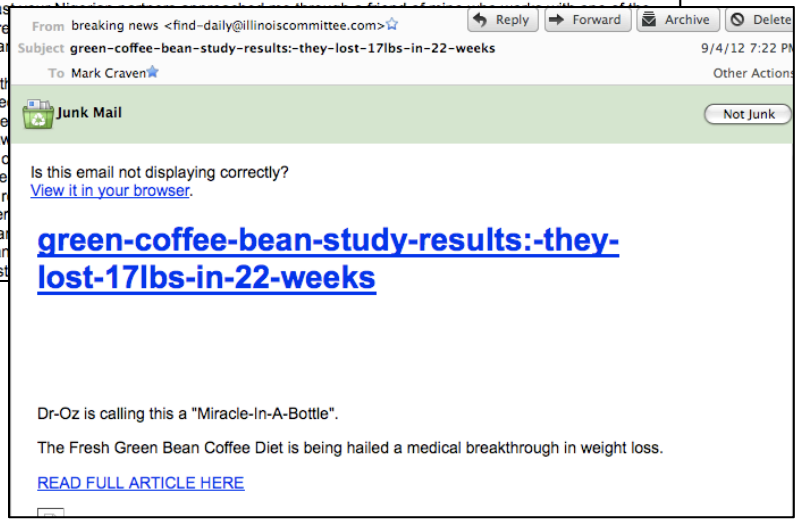
What is machine learning?

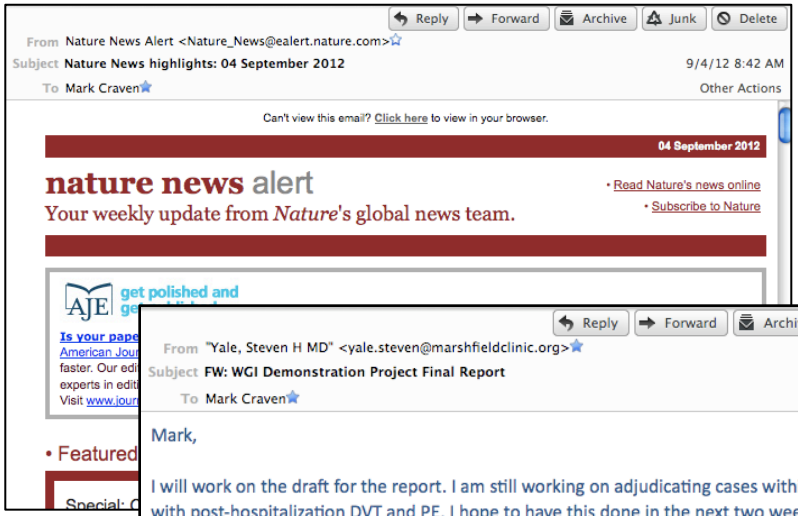
- the study of algorithms that
  improve their performance P
  at some task T
  with experience E

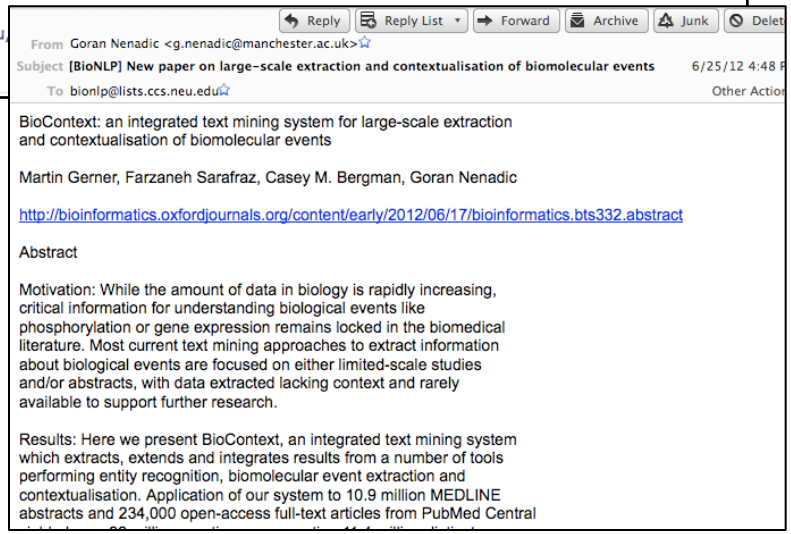
- to have a well defined learning task,
  we must specify: $< T, P, E >$

ML example: spam filtering







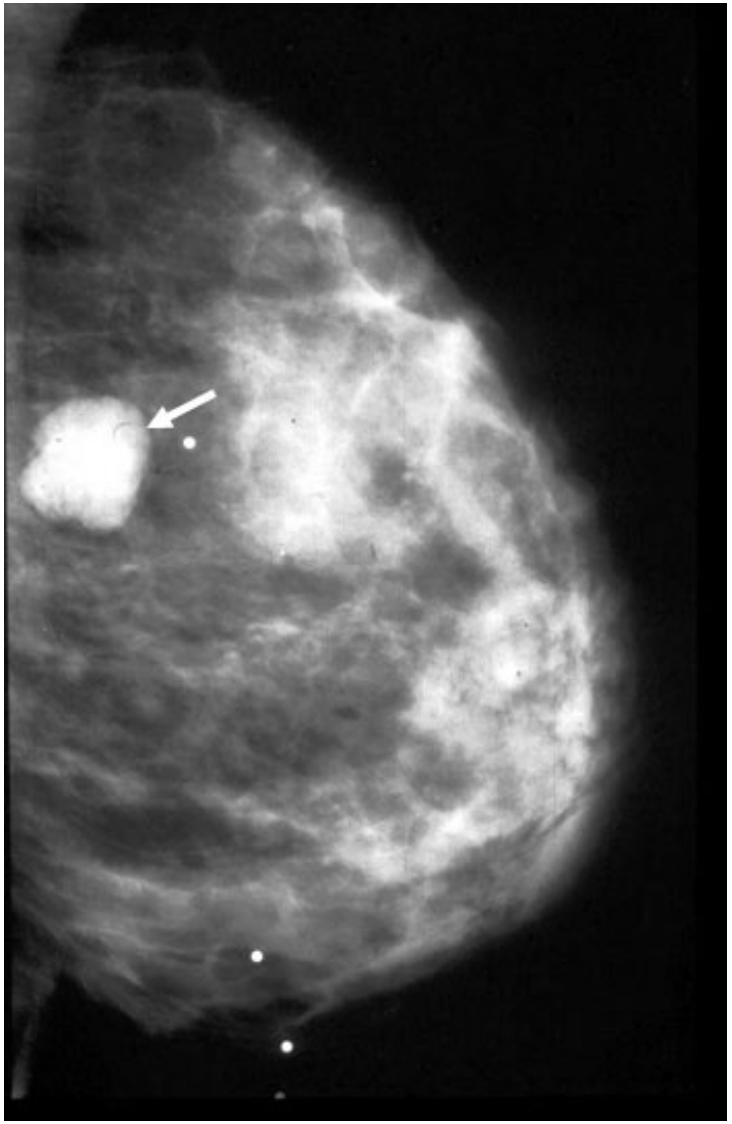


ML example: spam filtering

- T : given new mail message, classify as spam vs. other
- P : minimize misclassification costs
- E : previously classified (filed) messages

ML example: mammography

[Burnside et al., *Radiology* 2009]



ML example: mammography

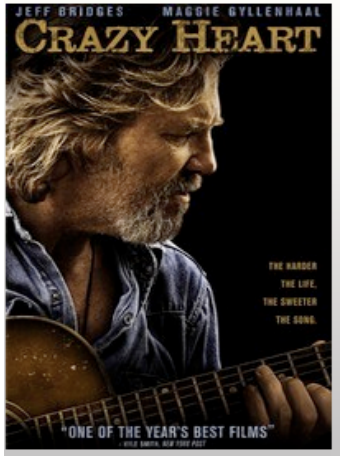
- T : given new mammogram, classify as benign vs. malignant
- P : minimize misclassification costs
- E : previously encountered patient histories (mammograms + subsequent outcomes)

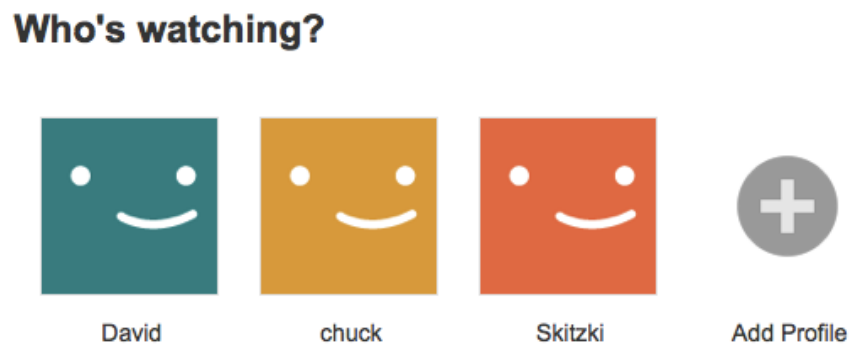
ML example: predictive text input

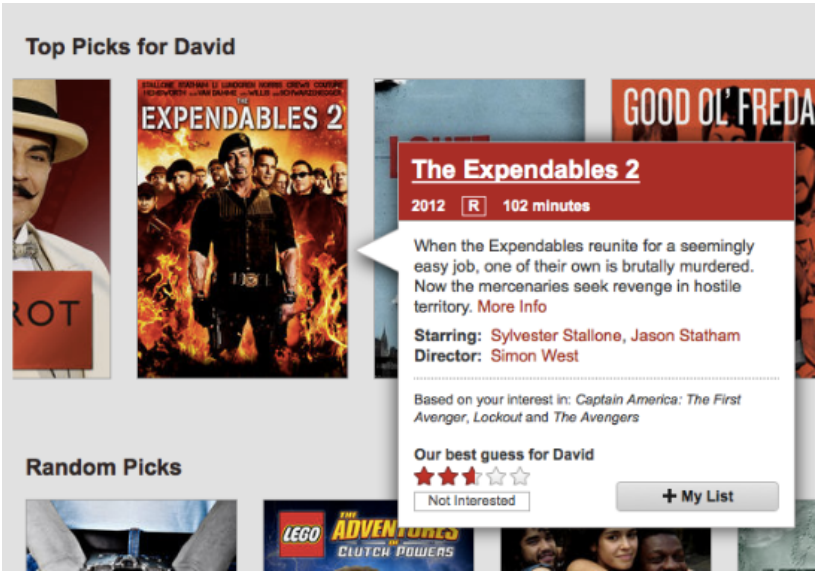
ML example: predictive text input

- T : given (partially) typed word, predict the word the user intended to type
- P : minimize misclassifications
- E : words previously typed by the user
  (+ lexicon of common words + knowledge of keyboard layout)

ML example: Netflix Prize







ML example: Netflix

- T : given a user/movie pair, predict the user's rating (1-5 stars) of the movie
- P : minimize difference between predicted and actual rating
- E : histories of previously rated movies (user, movie, rating triples)

Goals for this part of lecture

- define the supervised and unsupervised learning tasks
- consider how to represent instances as fixed-length feature vectors
- understand the concepts
  - instance (example)
  - feature (attribute)
  - feature space
  - feature types
  - supervised learning
  - classification (concept learning)
  - regression
  - i.i.d. assumption
  - generalization

Goals for the lecture (continued)

- understand the concepts
  - unsupervised learning
  - clustering
  - anomaly detection
  - dimensionality reduction

Can I eat this mushroom?

I don’t know what type it is – I’ve never seen it before. Is it edible or poisonous?

Can I eat this mushroom?

suppose we're given examples of edible and poisonous mushrooms
(we'll refer to these as *training examples* or *training instances*)

edible



poisonous




can we learn a model that can be used to classify other mushrooms?

Representing instances using feature vectors

- we need some way to represent each instance
- one common way to do this: use a fixed-length vector to represent *features* (a.k.a. *attributes*) of each instance
- also represent *class label* of each instance

	cap-shape	cap-surface	cap-color	bruises	odor		class
$\boldsymbol{x}_1 = \langle$	bell,	fibrous,	gray,	false,	foul,	$\ldots\rangle$	$y_1 =$ edible
$\boldsymbol{x}_2 = \langle$	convex,	scaly,	purple,	false,	musty,	$\ldots\rangle$	$y_2 =$ poisonous
$\boldsymbol{x}_3 = \langle$	bell,	smooth,	red,	true,	musty,	$\ldots\rangle$	$y_3 =$ edible
			$\vdots$				

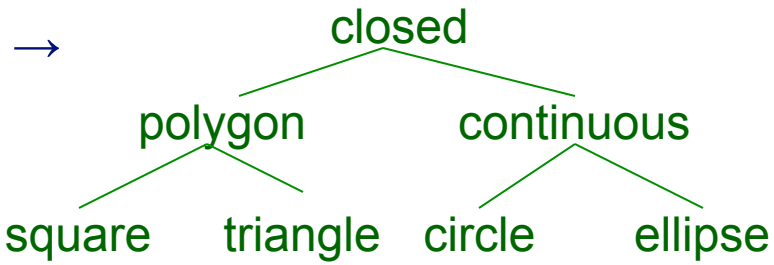
Standard feature types

- *nominal* (including Boolean)
  - no ordering among possible values
    e.g. *color* ϵ {*red*, *blue*, *green*} (vs. *color* = *1000* Hertz)

- *linear* (or *ordinal*)
  - possible values of the feature are totally ordered
    e.g. *size* ϵ *{small, medium, large}* ← discrete
    weight ϵ [0…500] ← continuous

- *hierarchical*
  - possible values are partially *ordered* in an ISA hierarchy

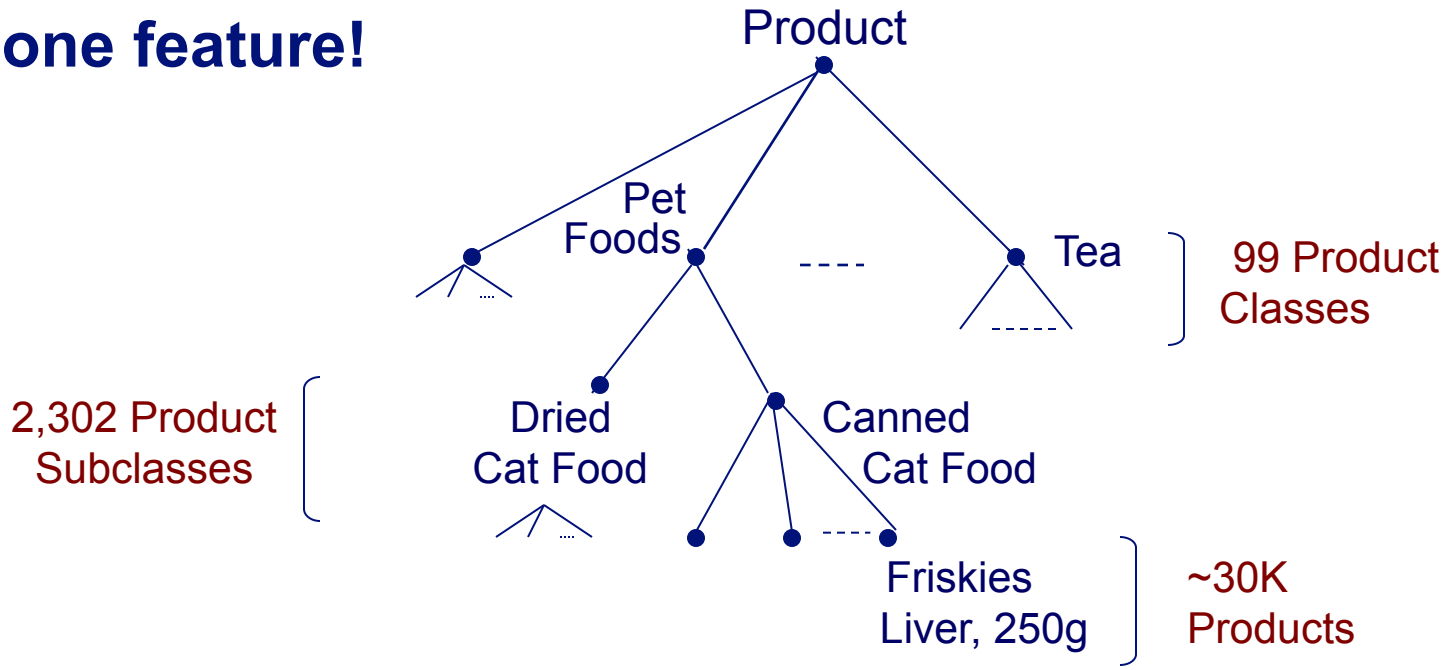
e.g. *shape* →



Feature hierarchy example

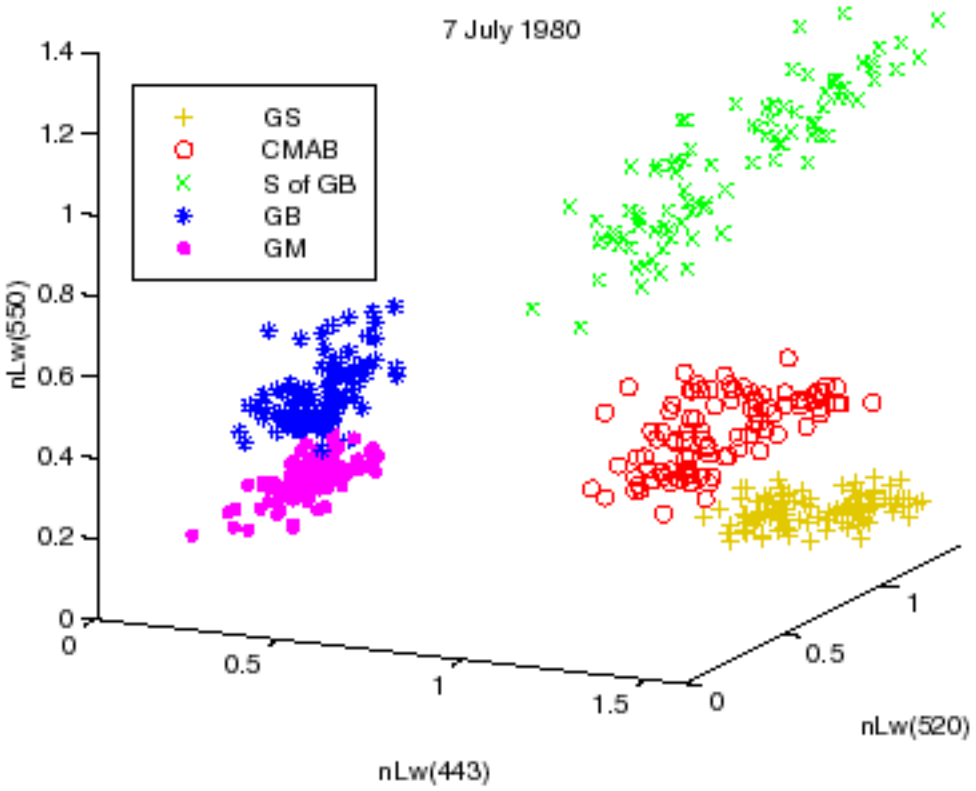
Lawrence et al., *Data Mining and Knowledge Discovery* 5(1-2), 2001

Structure of one feature!



Feature space

we can think of each instance as representing a point in a d-dimensional feature space where d is the number of features



example: optical properties of oceans in three spectral bands
[Traykovski and Sosik, *Ocean Optics XIV Conference Proceedings*, 1998]

Another view of the feature-vector representation: a single database table

	feature 1	feature 2	. . .	feature d	class
instance 1	0.0	small		red	true
instance 2	9.3	medium		red	false
instance 3	8.2	small		blue	false
. . .					
instance n	5.7	medium		green	true

Representation Caveat

- Feature vector format has proved very "workable"
- But much real world data doesn't arrive in neatly aligned feature vectors
  - Sequences: events in time, genomes, books
  - Graphs: social networks, logistics, comms
  - Relational databases: patient's health data distributed over many tables

ML example: Stock Forecasting

ML example: Stock Forecasting

- T : given a stock, predict the value tomorrow/next week/next month
- P : minimize difference between predicted and actual value
- E : histories of this stock, other stocks

- Alternatives in specification:
  - T: given NYSE, choose an investment strategy
  - P: maximize profit
  - E: might also include background information about companies

ML example: Personalized Medicine

Demographics

ID	Year of Birth	Gender
P1	3.10.1946	M

Diagnoses

ID	Date	Diagnosis	Sign/Symptom
P1	6.2.2011	Atrial fibrillation	Discomfort

The Electronic Health Record (EHR)

Demographics

ID	Year of Birth	Gender
P1	1946.03.10	M

Diagnoses

ID	Date	Diagnosis	Sign/Symptom
P1	7.3.2011	Atrial fibrillation	Dizziness, Nausea

The Electronic Health Record (EHR)

Demographics

ID	Year of Birth	Gender
P1	1946.03.10	M

Diagnoses

ID	Date	Diagnosis	Sign/Symptom
P1	2.29.2012	Stroke	Schizophasia

The Electronic Health Record (EHR)

Demographics

ID	Year of Birth	Gender
P1	1946.03.10	M

Diagnoses

ID	Date	Diagnosis	Sign/Symptom
P1	6.2.2011	Atrial fibrillation	Discomfort
P1	7.3.2011	Atrial fibrillation	Dizziness, Nausea
P1	2.29.2012	Stroke	Schizophasia

Electronic Health Record (EHR)

Patient ID	Gender	Birthdate
P1	M	3/22/1963

Demographics

Patient ID	Date	Physician	Symptoms	Diagnosis
P1	1/1/2001	Smith	palpitations	hypoglycemic
P1	2/1/2001	Jones	fever, aches	influenza

Diagnoses

Patient ID	Date	Lab Test	Result
P1	1/1/2001	blood glucose	42
P1	1/9/2001	blood glucose	45

Lab Results

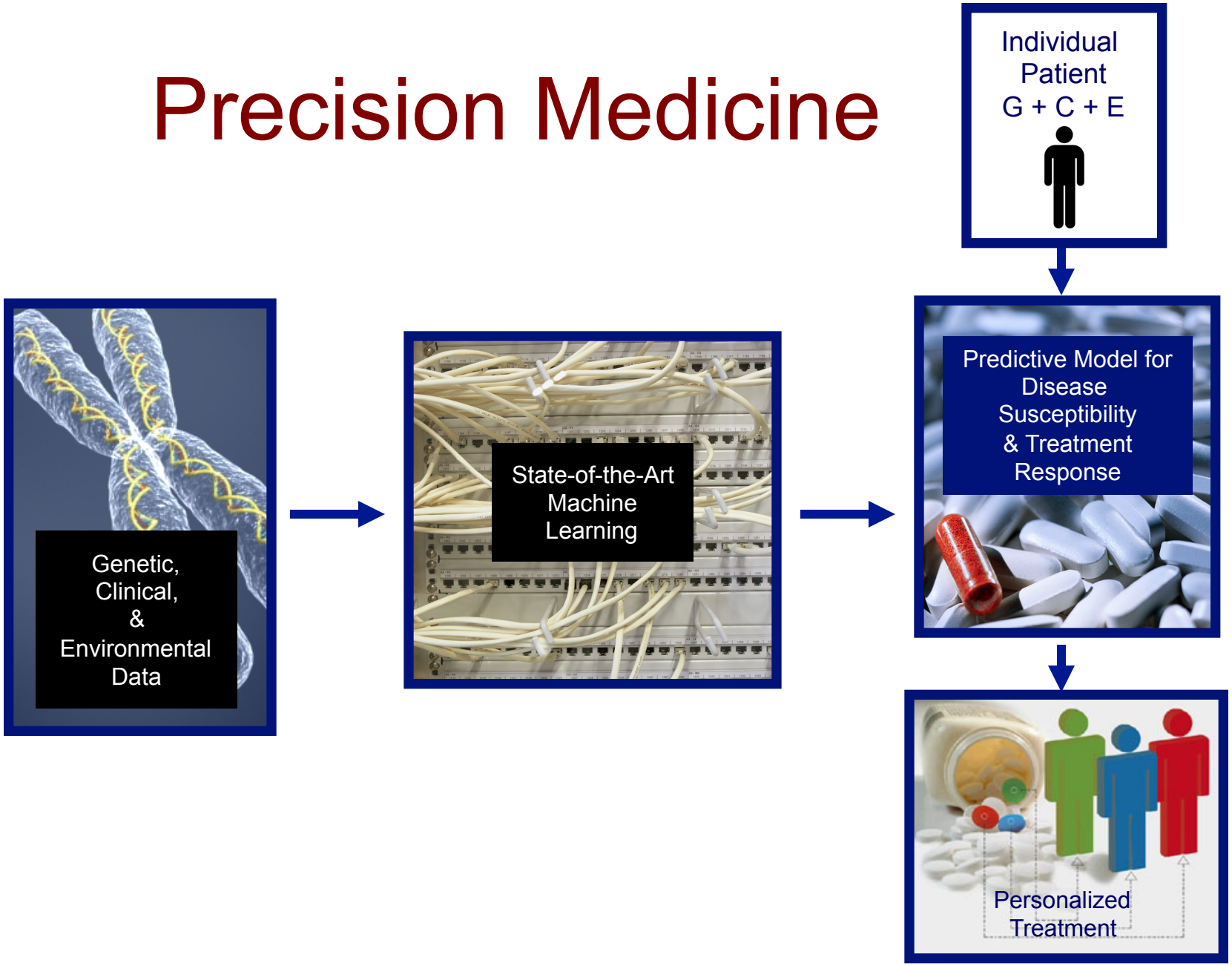
Patient ID	Date	Observation	Result
P1	1/1/2001	Height	5'11
P2	1/9/2001	BMI	34.5

Vitals

Medications

Patient ID	Date Prescribed	Date Filled	Physician	Medication	Dose	Duration
P1	5/17/1998	5/18/1998	Jones	Prilosec	10mg	3 months

Precision Medicine



ML example: Precision Medicine

- T : given a patient and disease diagnosis, choose best treatment
- P : cure disease
- E : treatment and outcomes for other patients with same disease
  (+ electronic health records (EHRs) + genome sequences)

- Alternatives in specification:
  - T: given a patient, choose lifestyle and treatment plan
  - P: maximize patient health as measured by survey questions
  - E: might also include answers to questionnaire about lifestyle

Back to Feature Vectors

	feature 1	feature 2	. . .	feature d	class
instance 1	0.0	small		red	true
instance 2	9.3	medium		red	false
instance 3	8.2	small		blue	false
. . .					
instance n	5.7	medium		green	true

The supervised learning task

problem setting

- set of possible instances: X
- unknown *target function*: $f : X \rightarrow Y$
- set of *models* (a.k.a. *hypotheses*): $H = \{h \mid h : X \rightarrow Y\}$

given

- training set of instances of unknown target function f

$(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2) \ \ldots \ (\boldsymbol{x}_n, y_n)$

output

- model $h \in H$ that best approximates target function

The supervised learning task

- when y is discrete, we term this a *classification* task (or *concept learning*)
- when y is continuous, it is a *regression* task
- later in the semester, we will consider tasks in which each y is more structured object (e.g. a *sequence* of discrete labels)

i.i.d. instances

- we often assume that training instances are *independent and identically distributed* (i.i.d.) – sampled independently from the same unknown distribution

- later in the course we'll consider cases where this assumption does not hold
  - cases where sets of instances have dependencies
    - instances sampled from the same medical image
    - instances from time series
    - etc.
  - cases where the learner can select which instances are labeled for training
    - *active learning*
  - the target function changes over time (*concept drift*)

Generalization

- The primary objective in supervised learning is to find a model that *generalizes* – one that accurately predicts y for previously unseen $\boldsymbol{x}$

Can I eat this mushroom that **was not** in my training set?

Model representations

throughout the semester, we will consider a broad range of representations for learned models, including

- decision trees
- neural networks
- support vector machines
- Bayesian networks
- logic clauses
- ensembles of the above
- etc.

Mushroom features (from the UCI Machine Learning Repository)

sunken is one possible value of the *cap-shape* feature

cap-shape: bell=b,conical=c,convex=x,flat=f, knobbed=k,sunken=s
cap-surface: fibrous=f,grooves=g,scaly=y,smooth=s
cap-color: brown=n,buff=b,cinnamon=c,gray=g,green=r, pink=p,purple=u,red=e,white=w,yellow=y
bruises?: bruises=t,no=f
odor: almond=a,anise=l,creosote=c,fishy=y,foul=f, musty=m,none=n,pungent=p,spicy=s
gill-attachment: attached=a,descending=d,free=f,notched=n
gill-spacing: close=c,crowded=w,distant=d
gill-size: broad=b,narrow=n
gill-color: black=k,brown=n,buff=b,chocolate=h,gray=g, green=r,orange=o,pink=p,purple=u,red=e, white=w,yellow=y
stalk-shape: enlarging=e,tapering=t
stalk-root: bulbous=b,club=c,cup=u,equal=e, rhizomorphs=z,rooted=r,missing=?
stalk-surface-above-ring: fibrous=f,scaly=y,silky=k,smooth=s
stalk-surface-below-ring: fibrous=f,scaly=y,silky=k,smooth=s
stalk-color-above-ring: brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y
stalk-color-below-ring: brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y
veil-type: partial=p,universal=u
veil-color: brown=n,orange=o,white=w,yellow=y
ring-number: none=n,one=o,two=t
ring-type: cobwebby=c,evanescent=e,flaring=f,large=l, none=n,pendant=p,sheathing=s,zone=z
spore-print-color: black=k,brown=n,buff=b,chocolate=h,green=r, orange=o,purple=u,white=w,yellow=y
population: abundant=a,clustered=c,numerous=n, scattered=s,several=v,solitary=y
habitat: grasses=g,leaves=l,meadows=m,paths=p, urban=u,waste=w,woods=d

A learned decision tree

```
odor = a: e (400.0)
odor = c: p (192.0)
odor = f: p (2160.0)
odor = l: e (400.0)
odor = m: p (36.0)
odor = n
|   spore-print-color = b: e (48.0)
|   spore-print-color = h: e (48.0)
|   spore-print-color = k: e (1296.0)
|   spore-print-color = n: e (1344.0)
|   spore-print-color = o: e (48.0)
|   spore-print-color = r: p (72.0)
|   spore-print-color = u: e (0.0)
|   spore-print-color = w
|   |   gill-size = b: e (528.0)
|   |   gill-size = n
|   |   |   gill-spacing = c: p (32.0)
|   |   |   gill-spacing = d: e (0.0)
|   |   |   gill-spacing = w
|   |   |   |   population = a: e (0.0)
|   |   |   |   population = c: p (16.0)
|   |   |   |   population = n: e (0.0)
|   |   |   |   population = s: e (0.0)
|   |   |   |   population = v: e (48.0)
|   |   |   |   population = y: e (0.0)
|   spore-print-color = y: e (48.0)
odor = p: p (256.0)
odor = s: p (576.0)
odor = y: p (576.0)
```

if odor=almond, predict edible

if odor=none $\wedge$
spore-print-color=white $\wedge$
gill-size=narrow $\wedge$
gill-spacing=crowded,
predict poisonous

Classification with a learned decision tree

once we have a learned model, we can use it to classify previously unseen instances

$\boldsymbol{x} = \langle$bell, fibrous, brown, false, foul, …$\rangle$

```
odor = a: e (400.0)
odor = c: p (192.0)
odor = f: p (2160.0)
odor = l: e (400.0)
odor = m: p (36.0)
odor = n
|   spore-print-color = b: e (48.0)
|   spore-print-color = h: e (48.0)
|   spore-print-color = k: e (1296.0)
|   spore-print-color = n: e (1344.0)
|   spore-print-color = o: e (48.0)
|   spore-print-color = r: p (72.0)
|   spore-print-color = u: e (0.0)
|   spore-print-color = w
|   |   gill-size = b: e (528.0)
|   |   gill-size = n
|   |   |   gill-spacing = c: p (32.0)
|   |   |   gill-spacing = d: e (0.0)
|   |   |   gill-spacing = w
|   |   |   |   population = a: e (0.0)
|   |   |   |   population = c: p (16.0)
|   |   |   |   population = n: e (0.0)
|   |   |   |   population = s: e (0.0)
|   |   |   |   population = v: e (48.0)
|   |   |   |   population = y: e (0.0)
|   spore-print-color = y: e (48.0)
odor = p: p (256.0)
odor = s: p (576.0)
odor = y: p (576.0)
```

$y =$ edible or poisonous?

Unsupervised learning

in unsupervised learning, we're given a set of instances, without y's

$$\boldsymbol{x}_1, \boldsymbol{x}_2 \ \ldots \ \boldsymbol{x}_n$$

goal: discover interesting regularities that characterize the instances

common unsupervised learning tasks

- *clustering*
- *anomaly detection*
- *dimensionality reduction*

Clustering

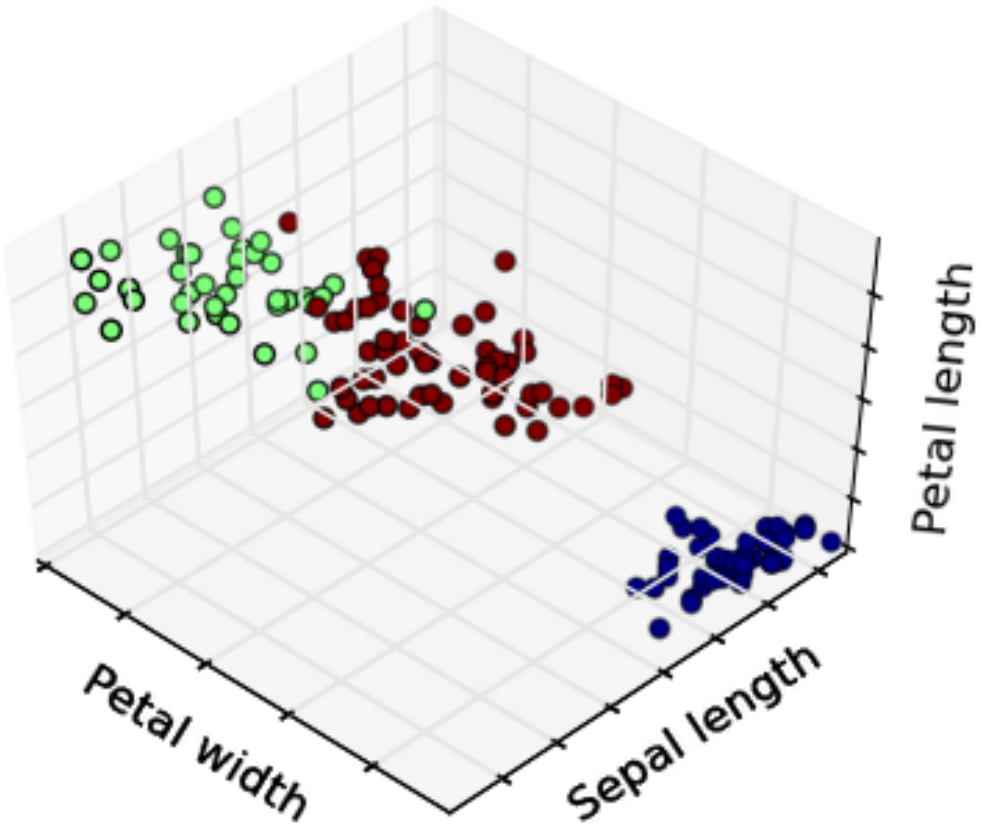
given

- training set of instances $\boldsymbol{x}_1, \boldsymbol{x}_2 \ldots \boldsymbol{x}_n$

output

- model $h \in H$ that divides the training set into clusters such that there is intra-cluster similarity and inter-cluster dissimilarity

Clustering example



Clustering irises using three different features (the colors represent clusters identified by the algorithm, not y's provided as input)

Anomaly detection

learning task

given

- training set of instances $\boldsymbol{x}_1, \boldsymbol{x}_2 \ldots \boldsymbol{x}_n$

output

- model $h \in H$ that represents "normal" $\boldsymbol{x}$

performance task

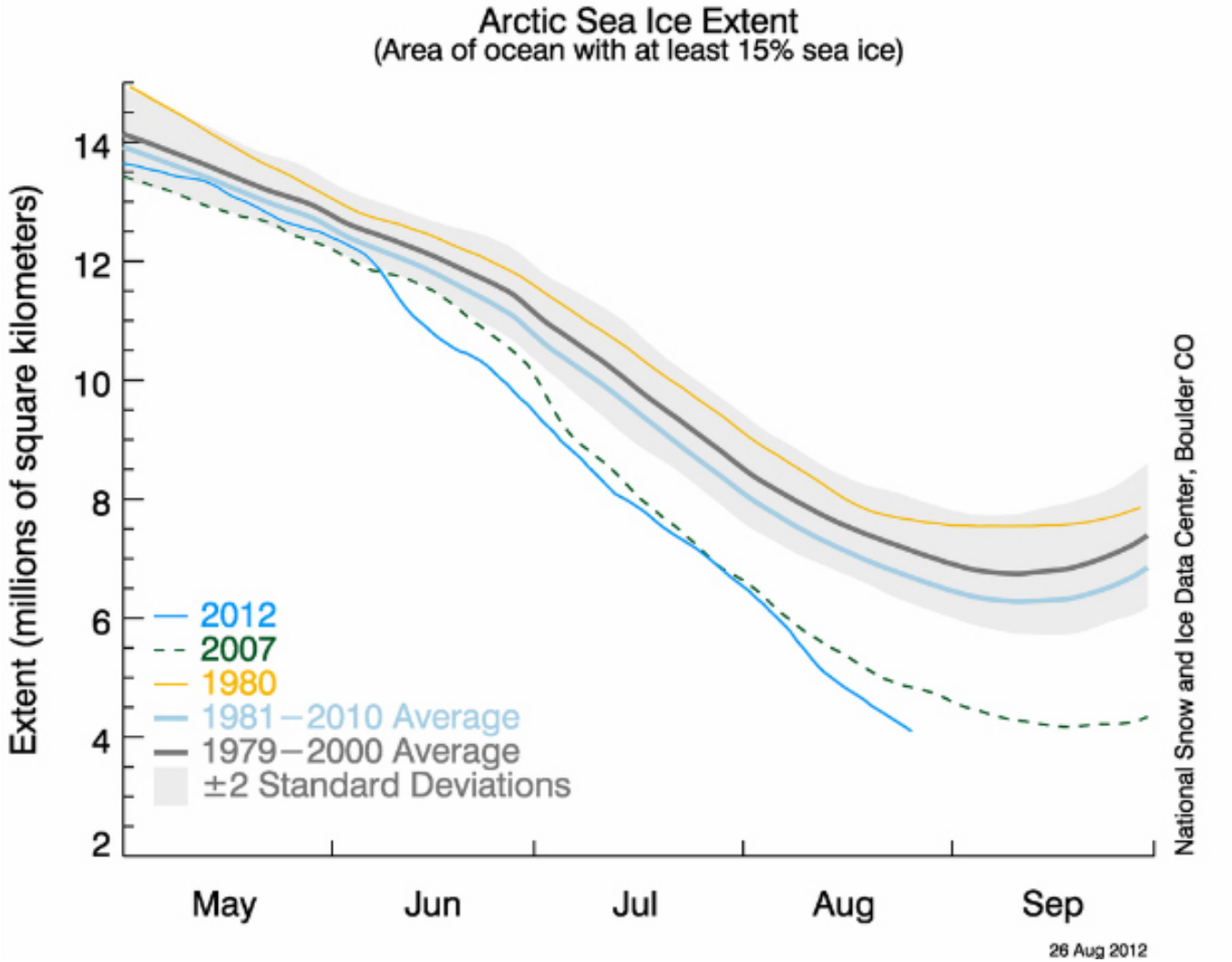
given

- a previously unseen $\boldsymbol{x}$

determine

- if $\boldsymbol{x}$ looks normal or anomalous

Anomaly detection example



Does the data for 2012 look anomalous?

Dimensionality reduction

given

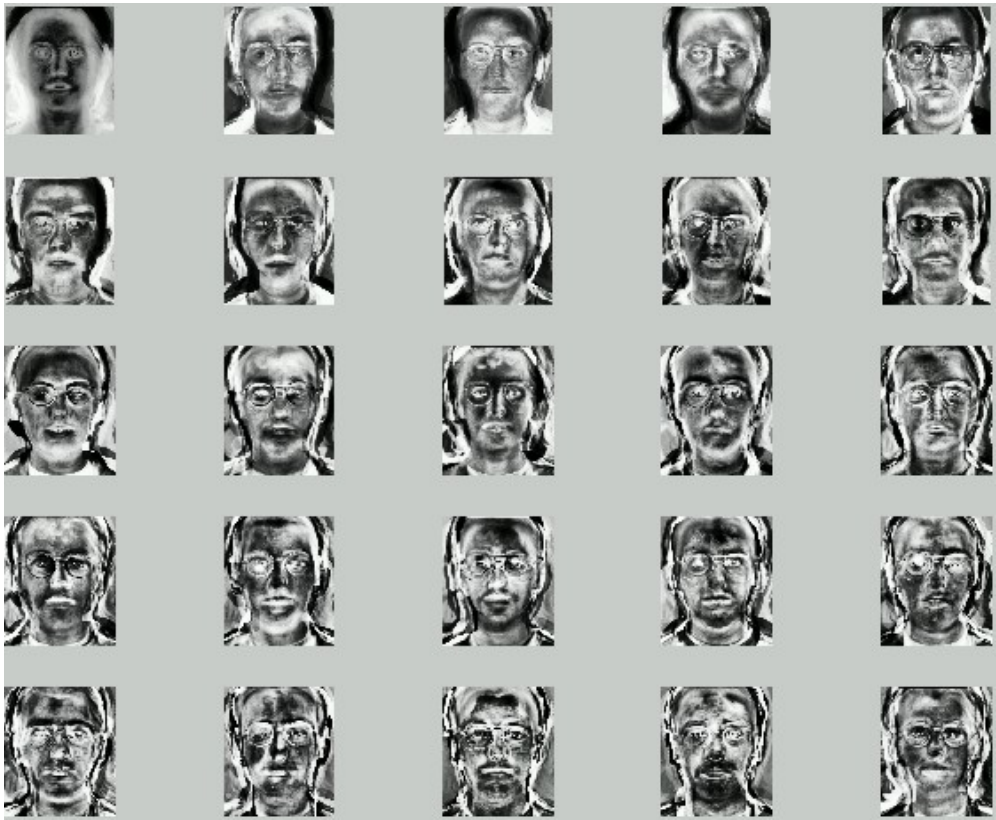
- training set of instances $\boldsymbol{x}_1, \boldsymbol{x}_2 \ldots \boldsymbol{x}_n$

output

- model $h \in H$ that represents each $\boldsymbol{x}$ with a lower-dimension feature vector while still preserving key properties of the data

Dimensionality reduction example

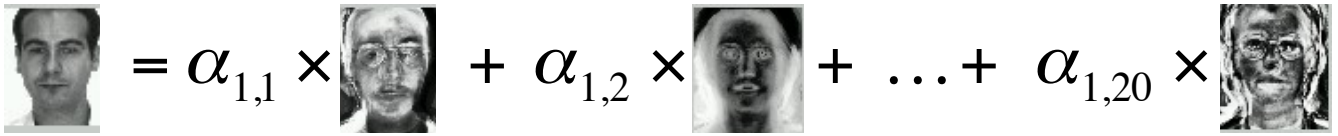
We can represent a face using all of the pixels in a given image



More effective method (for many tasks): represent each face as a linear combination of *eigenfaces*

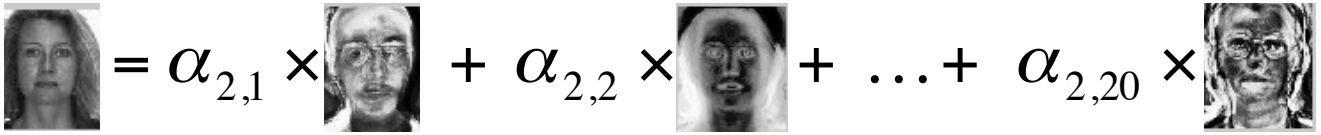
Dimensionality reduction example

represent each face as a linear combination of *eigenfaces*



$$= \alpha_{1,1} \times + \alpha_{1,2} \times + \ldots + \alpha_{1,20} \times$$

$$\boldsymbol{x}_1 = \langle \alpha_{1,1}, \alpha_{1,2}, \ldots, \alpha_{1,20} \rangle$$



$$= \alpha_{2,1} \times + \alpha_{2,2} \times + \ldots + \alpha_{2,20} \times$$

$$\boldsymbol{x}_2 = \langle \alpha_{2,1}, \alpha_{2,2}, \ldots, \alpha_{2,20} \rangle$$

# of features is now 20 instead of # of pixels in images

Other learning tasks

later in the semester we'll cover other learning tasks that are not strictly supervised or unsupervised

- *reinforcement learning*
- *semi-supervised learning*
- *etc.*