CS 760: Machine Learning
ML Overview

Fred Sala

University of Wisconsin-Madison

DATE, 2021

Announcements

- **HW 1 due Thursday**:
    - Self-test, should feel mostly easy
- Class roadmap:

Tuesday Sept. 14	ML Overview
Thursday Sept. 16	Supervised Learning I
Tuesday Sept. 21	Supervised Learning II
Thursday Sept. 23	Evaluation
Tuesday Sept. 28	Regression I

Mostly SL

Outline

- **Review from last time**
  - Supervised vs. unsupervised learning
- **Supervised learning concepts**
  - Features, models, training, other terminology
- **Unsupervised learning concepts**
  - Clustering, anomaly detection, dimensionality reduction

Outline

- **Review from last time**
  - Supervised vs. unsupervised learning
- **Supervised learning concepts**
  - Features, models, training, other terminology
- **Unsupervised learning concepts**
  - Clustering, anomaly detection, dimensionality reduction

Review: ML Overview: Definition

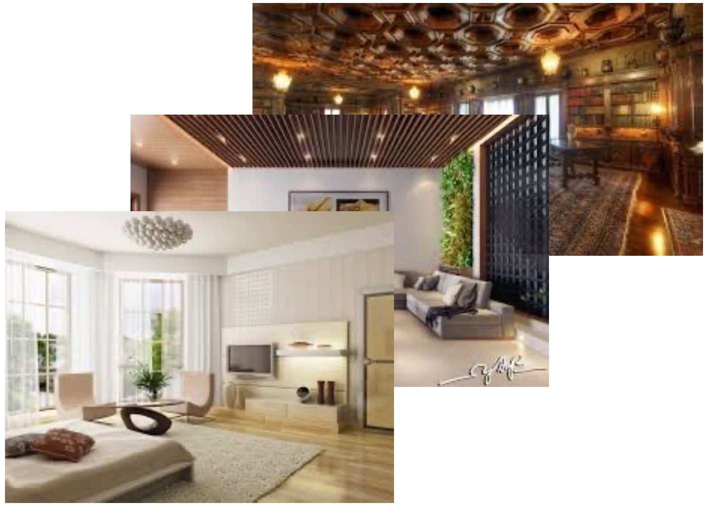
What is machine learning?

“A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T** as measured by **P**, improves with experience **E**.” *Machine Learning*, Tom Mitchell, 1997

ML Overview: Flavors

Supervised Learning

- Learning from examples, as above
- **Workflow**:
  - Collect a set of examples {data, labels}: **training set**
  - “**Train**” a model to match these examples
  - ”**Test**” it on new data
- **Image classification**:



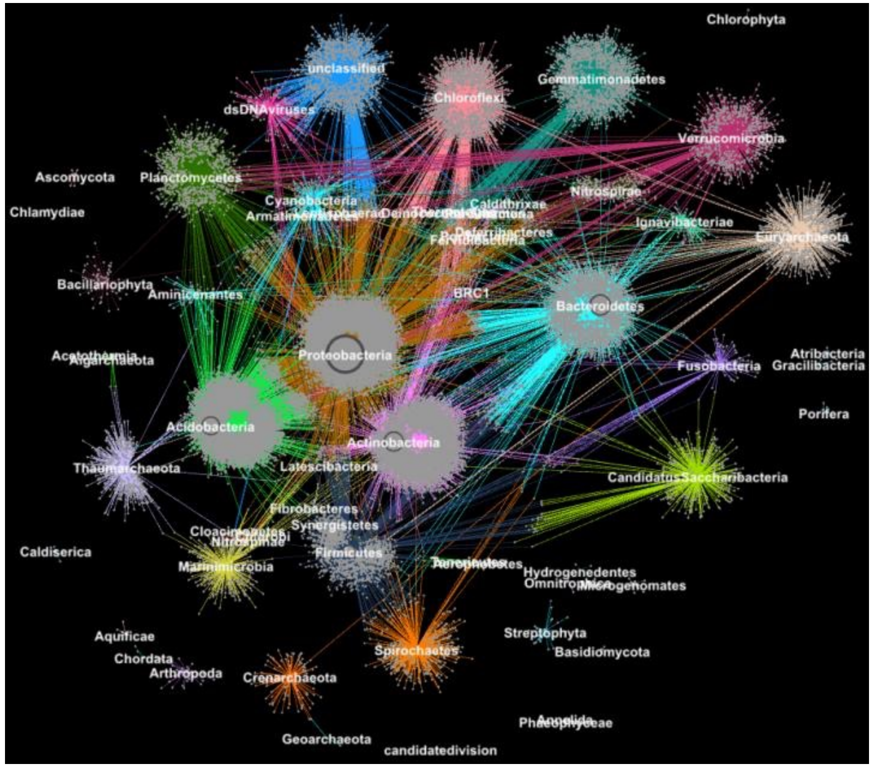
indoor

outdoor

ML Overview: Flavors

Unsupervised Learning

- Data, but no labels. No input/output.
- Goal: get “something”: structure, hidden information, more
- **Workflow**:
  - Collect a set {data}
  - Perform some algorithm on it



- **Clustering**: reveal hidden structure

ML Overview: Flavors

Reinforcement Learning

- Agent interacting with the world; gets rewards for actions
- Goal: learn to perform some activity
- **Workflow**:
  - Create an environment, reward, agent
  - **Train**: modify policy to maximize rewards
  - **Deploy** in new environment

- **Controlling aircraft**: learn to fly

Outline

- **Review from last time**
  - Supervised vs. unsupervised learning
- **Supervised learning concepts**
  - Features, models, training, other terminology
- **Unsupervised learning concepts**
  - Clustering, anomaly detection, dimensionality reduction

Supervised Learning

- Can I eat this?
- Safe or poisonous?
  - **Never seen it before**
- How to decide?

Supervised Learning: Training Instances

- I know about other mushrooms:

safe



poisonous




- Training set of **examples/instances/labeled data**

Supervised Learning: Formal Setup

Problem setting

- Set of possible instances $\mathcal{X}$
- Unknown *target function* $f : \mathcal{X} \rightarrow \mathcal{Y}$
- Set of *models* (a.k.a. *hypotheses*): $\mathcal{H} = \{h | h : \mathcal{X} \rightarrow \mathcal{Y}\}$

Get

- Training set of instances for unknown target function,

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$$

safe

poisonous

safe

Supervised Learning: Formal Setup

Problem setting

- Set of possible instances $\mathcal{X}$
- Unknown *target function* $f : \mathcal{X} \to \mathcal{Y}$
- Set of *models* (a.k.a. *hypotheses*) $\mathcal{H} = \{h | h : \mathcal{X} \to \mathcal{Y}\}$

Get

- Training set of instances for unknown target function *f*,

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$$

Goal: model *h* that best approximates *f*

Supervised Learning: Objects

Three types of sets

- Input space, output space, hypothesis class

$$\mathcal{X}, \mathcal{Y}, \mathcal{H}$$

- **Examples**:
  - Input space: feature vectors $\mathcal{X} \subseteq \mathbb{R}^d$

  - Output space:
    - **Binary** $\mathcal{Y} = \{-1, +1\}$ safe poisonous
    - **Continuous** $\mathcal{Y} \subseteq \mathbb{R}$ 13.23°

Input Space: Feature Vectors

- Need a way to represent instances:

cap-shape, cap-surface, cap-color, bruises, odor

$$\mathbf{x}^{(1)} = \langle \text{bell}, \quad \text{fibrous}, \; \text{gray}, \quad \text{false}, \; \text{foul}, \ldots \rangle$$

safe

- For each instance, store features as a vector.
    - What kinds of features can we have?

Input Space: Feature Types

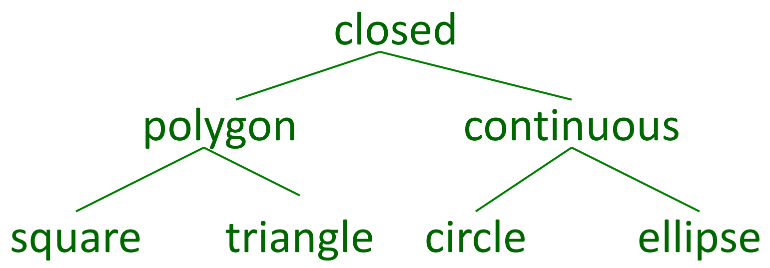
- *nominal* (including Boolean)
  - no ordering among values (e.g. *color* ∈ {*red*, *blue*, *green*}   (vs. *color = 1000* Hertz))

- *ordinal*
  - values of the feature are totally ordered (e.g. *size* ∈ {*small, medium, large*})

- *numeric (*continuous*)*

  weight ∈ [0…500]

- *hierarchical*
  - possible values are partially *ordered* in a hierarchy, e.g. *shape*



Input Space: Features Example

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

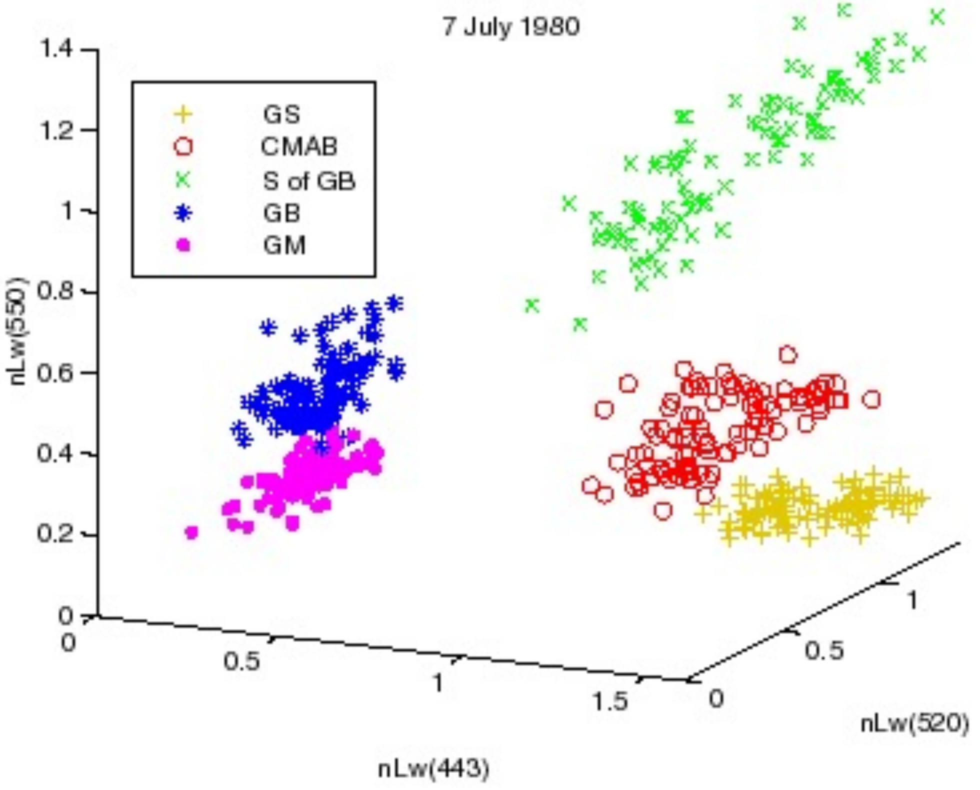
Mushroom features (UCI Repository)

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

Input Space: Feature Spaces

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

- **Example**: optical properties of oceans in three spectral bands



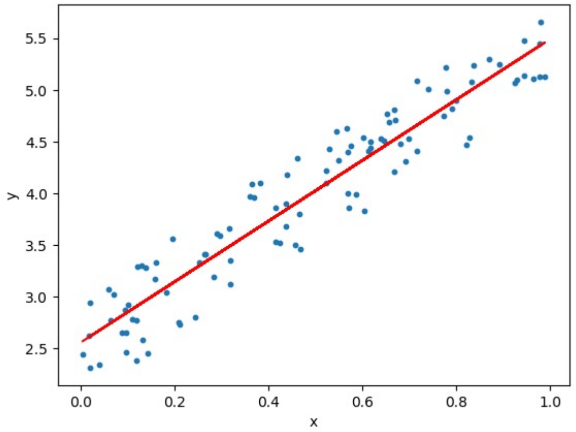
[Traykovski and Sosik, *Ocean Optics XIV Conference Proceedings*, 1998]

Output space: Classification vs. Regression

Choices of $\mathcal{Y}$ have special names:

- Discrete: "**classification**". The elements of $\mathcal{Y}$ are **classes**
  - Note: doesn't have to be binary

- Continuous: "**regression**"
  - Example: linear regression



- There are other types…

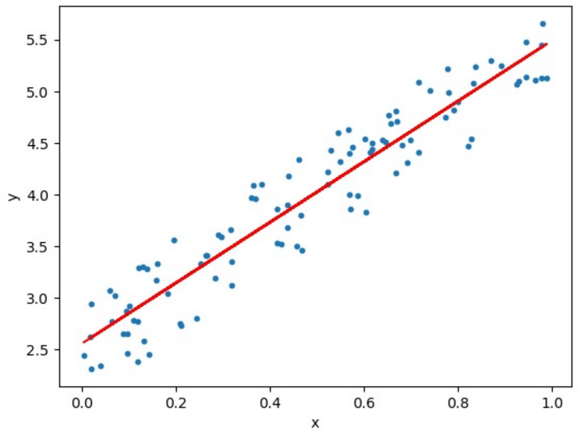
Hypothesis class

We talked about $\mathcal{X}, \mathcal{Y}$ what about $\mathcal{H}$?

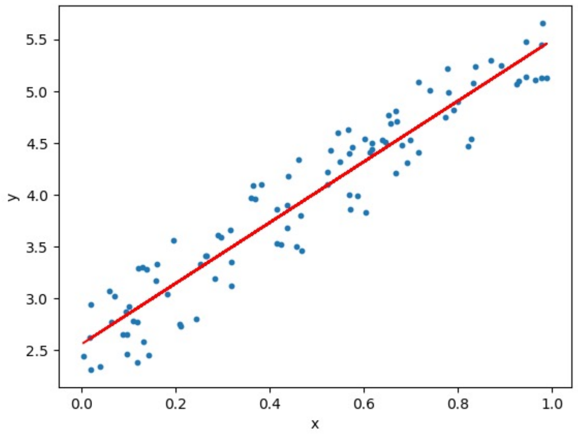
- Recall: hypothesis class / model space.
  - Theoretically, could be all maps from $\mathcal{X}$ to $\mathcal{Y}$
  - Doesn't work! Many reasons why.

- Pick specific class of models. Ex: linear models:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d$$



Hypothesis class: Linear Functions



- **Example** class of models: linear models

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d$$

Parameters (weights)

Features

- How many linear functions are there?
  - Can any function be fit by a linear model?

Hypothesis class: Other Examples

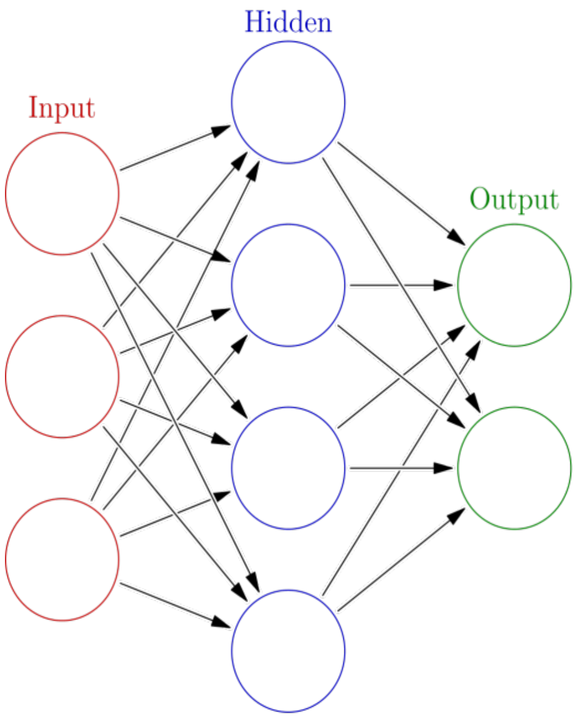
Example classes of models: neural networks

$$f^{(k)}(x) = \sigma(W_k^T f^{(k-1)}(x)))$$

Feedforward network

- Each layer:
  - linear transformation
  - Non-linearity

  - What are the parameters here?



Wikipedia

Back to Formal Setup

Problem setting

- Set of possible instances $\mathcal{X}$ ✓
- Unknown *target function* $f : \mathcal{X} \to \mathcal{Y}$ ✓
- Set of *models* (a.k.a. *hypotheses*) $\mathcal{H} = \{h | h : \mathcal{X} \to \mathcal{Y}\}$ ✓

Get

- Training set of instances for unknown target function *f*,

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$$ ✓

Goal: model *h* that best approximates *f*

Supervised Learning: Training

Goal: model *h* that best approximates *f*

- One way: empirical risk minimization (ERM)

$$\hat{f} = \arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(h(x^{(i)}), y^{(i)}))$$

Hypothesis Class

Loss function (how far are we)?

Model prediction

Batch vs. Online Learning

- **Batch learning**: get all your instances at once

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$$

- **Online learning**: get them sequentially
  - Train a model on initial group, then update

$$\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\} \qquad \{(x^{(m+1)}, y^{(m+1)})\}$$

Supervised Learning: Predicting

Now that we have our learned model, we can use it for predictions.

$$\mathbf{x} = \langle \text{bell, fibrous, brown, false, foul, ...} \rangle$$

```
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)
```

safe **or** **poisonous**

Interlude: Polynomials

Another class of models: polynomials:

$$h_\theta(x) = \theta_d x^d + \theta_{d-1} x^{d-1} + \ldots + \theta_1 x + \theta_0$$

- How to fit a polynomial?

Lagrange basis

$$L(x) = \sum_{i=0}^{n} y_n \ell_i(x)$$

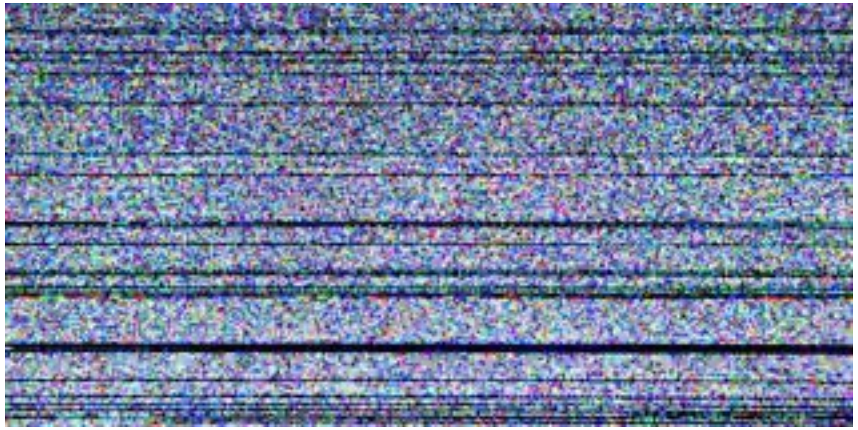
$$\ell_i(x) = \prod_{0 \le m \le n, m \ne i} \frac{x - x_m}{x_i - x_m}$$

Interlude: Polynomials

- Lagrange interpolation produces a **perfect fit**, e.g.,

$$L(x_i) = y_i \quad \forall i \in \{1, \ldots, n\}$$

- So, are we done?
  - More advantages: no training required. Just write down the *L*
  - **Q**: what degree are the x_i?
    - How sensitive to noise?
    - How will they **extrapolate**?



Generalization

Fitting data isn't the only task, we want to **generalize**

- Apply learned model to unseen data:
  - For $(x, y) \sim \mathcal{D}$,

$$\mathbb{E}_{\mathcal{D}}[\ell(\hat{f}(x), y)]$$

- Can study theoretically or empirically
  - For theory: need assumptions, ie, training instances are iid
  - Not always true!

Outline

- **Review from last time**
  - Supervised vs. unsupervised learning
- **Supervised learning concepts**
  - Features, models, training, other terminology
- **Unsupervised learning concepts**
  - Clustering, anomaly detection, dimensionality reduction

Unsupervised Learning: Setup

- Given instances $\{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$

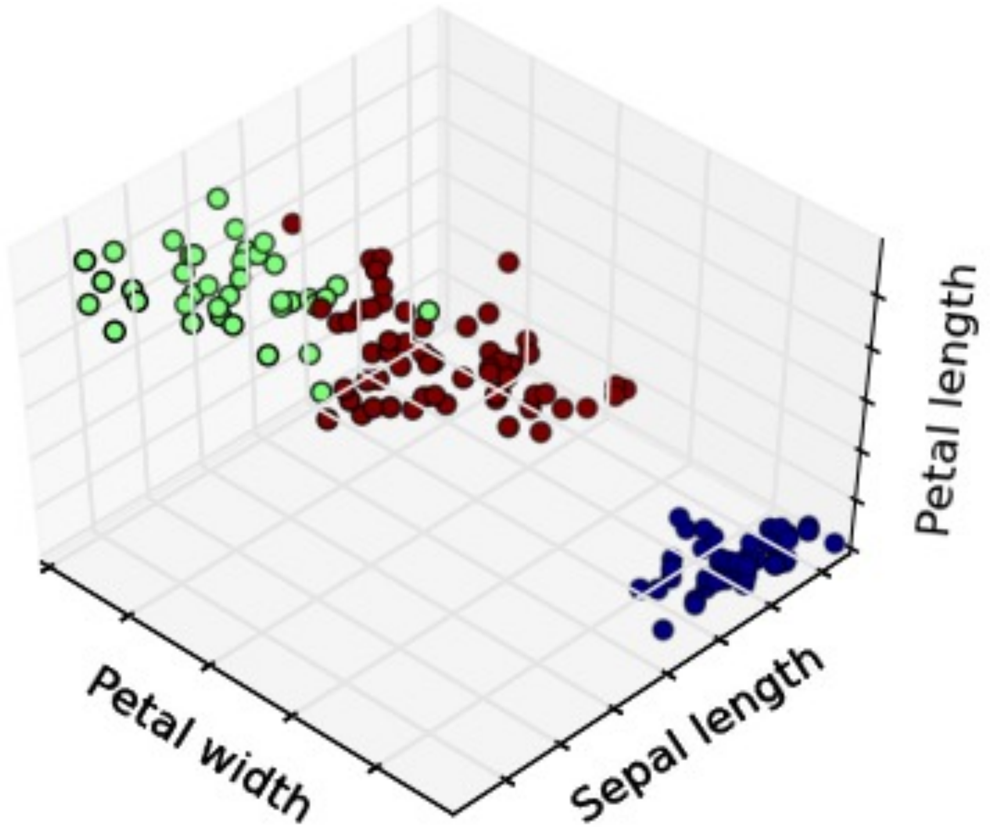
- **Goal**: discover interesting regularities/structures/patterns that characterize the instances. Ex:
  - clustering
  - anomaly detection
  - dimensionality reduction

Clustering: Setup

- Given instances $\{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$

- **Goal**: model *h* divides the training set into clusters with
  - intra-cluster similarity
  - inter-cluster dissimilarity

- Clustering *irises*:



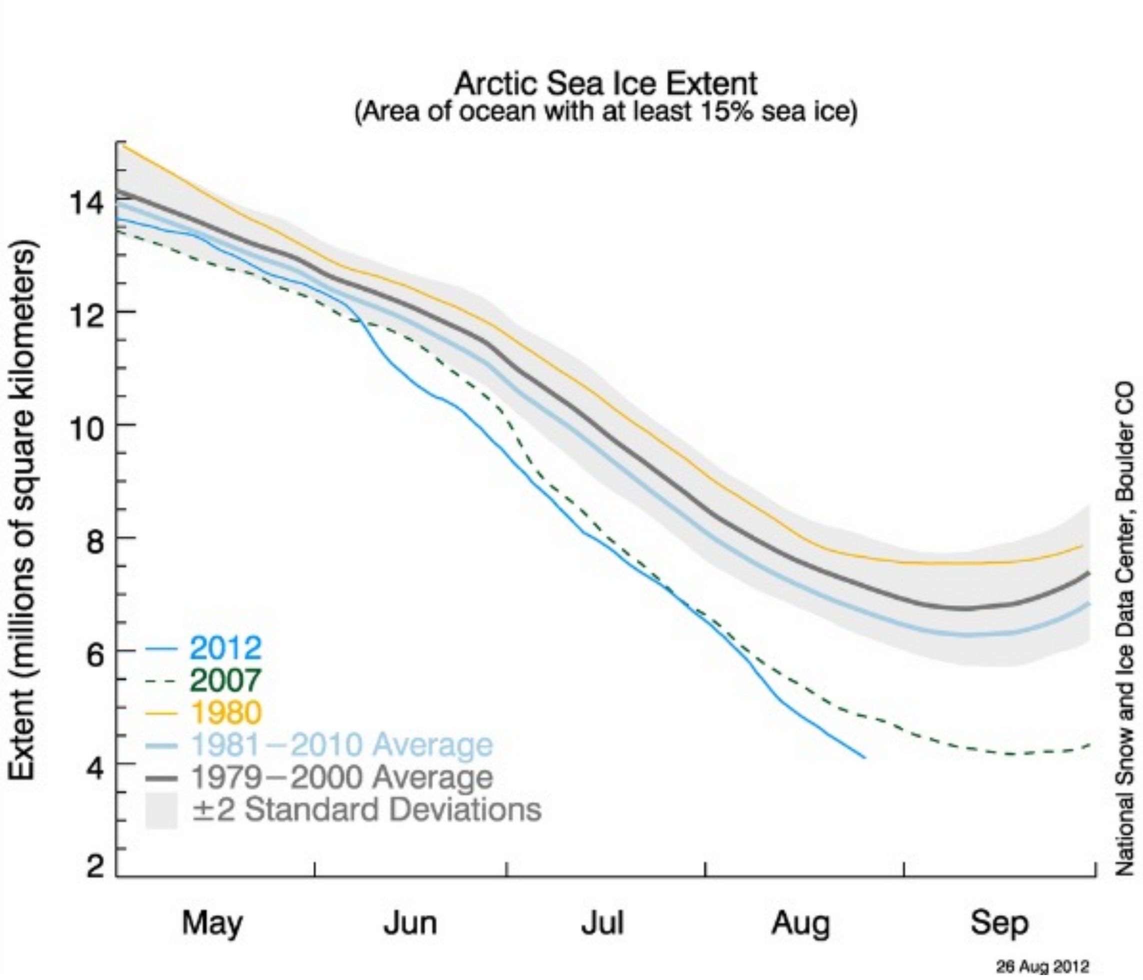
Anomaly Detection: Setup

- Given instances $\{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$

- **Goal**: model *h* that represents "normal" *x*
  - Can apply to new data to find anomalies

Let's say our model is represented by: 1979-2000 average, ±2 stddev

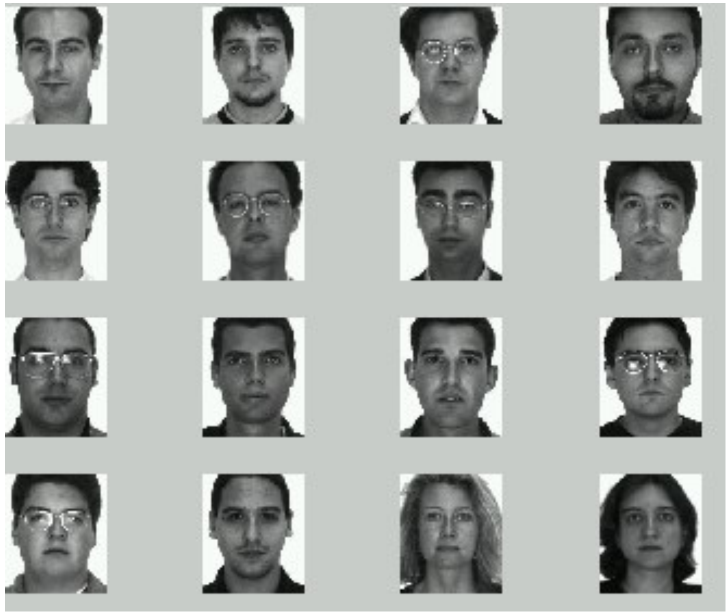
Does the data for 2012 look anomalous?

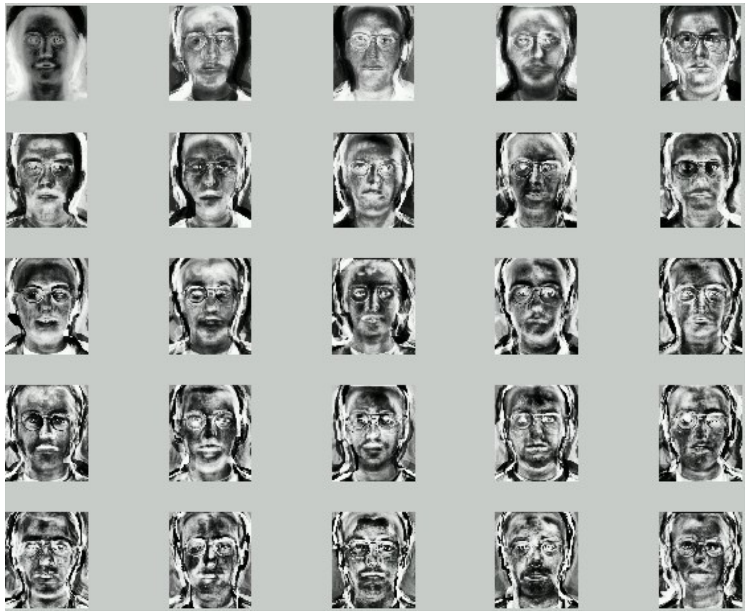


Dimensionality Reduction: Setup

- Given instances $\{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$

- **Goal**: model *h* that represents *x* with
  - lower-dim. feature vectors
  - preserving information
- Example: Eigenfaces





Dimensionality Reduction: Setup

Example: Eigenfaces

 $= \alpha_1^{(1)} \times$  $+ \alpha_2^{(1)} \times$  $+ \ldots + \alpha_{20}^{(1)} \times$ 

$$x^{(1)} = \langle \alpha_1^{(1)}, \alpha_2^{(1)}, \ldots, \alpha_{20}^{(1)} \rangle$$

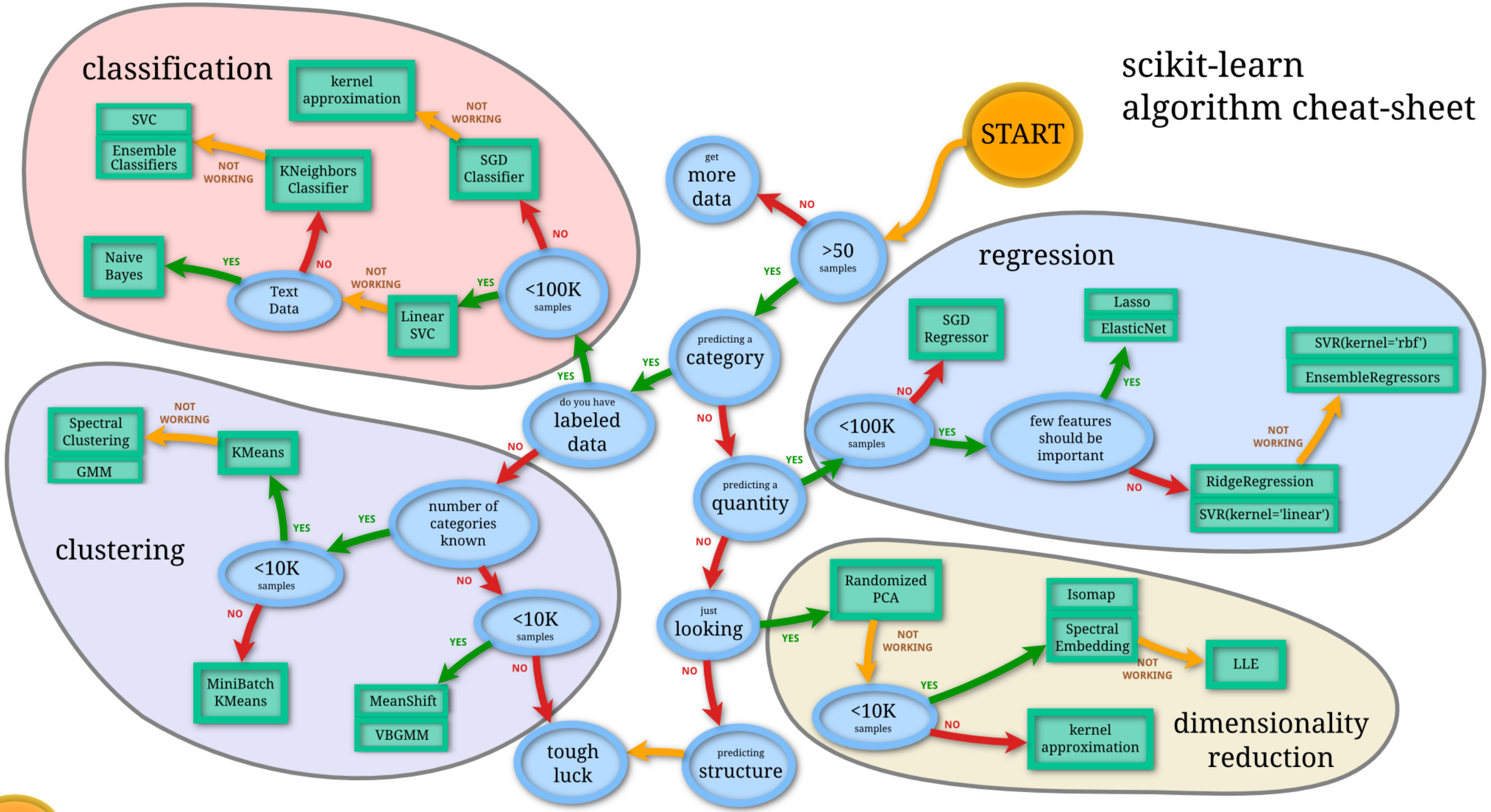
 $= \alpha_1^{(2)} \times$  $+ \alpha_2^{(2)} \times$  $+ \ldots + \alpha_{20}^{(2)} \times$ 

$$x^{(1)} = \langle \alpha_1^{(2)}, \alpha_2^{(2)}, \ldots, \alpha_{20}^{(2)} \rangle$$

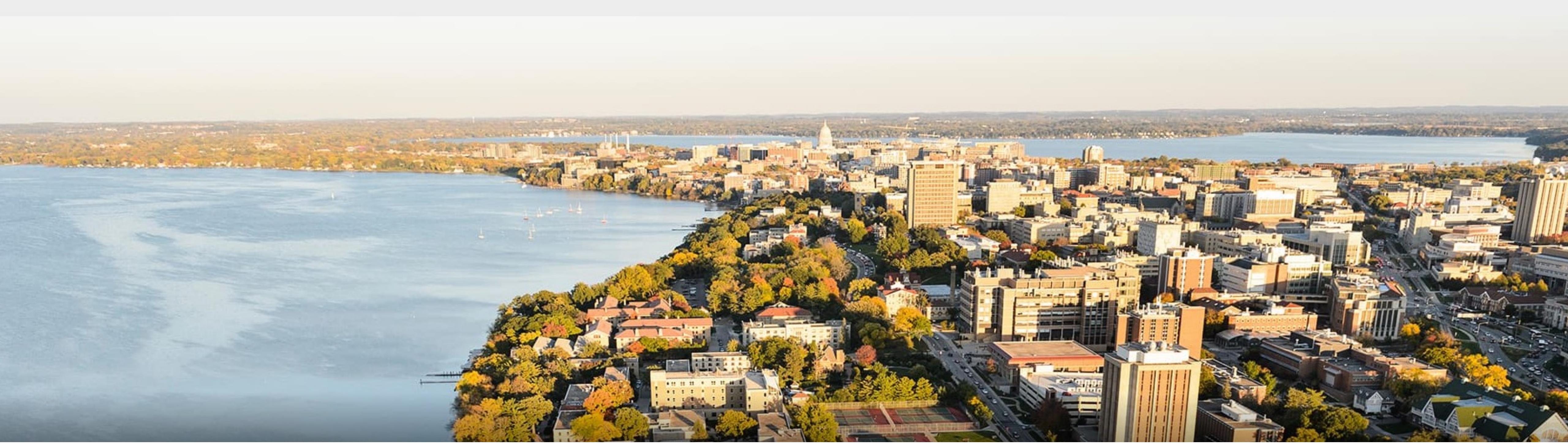
What dimension are we using now?

Model Zoo

Lots of models!



Back



Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov