Machine Learning: Overview

CS 760@UW-Madison

Goals for the 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
  - model (hypothesis)
  - training set
  - supervised learning
  - classification (concept learning) vs. regression
  - batch vs. online learning
  - 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 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

$$\begin{array}{llllllll}
 & \text{cap-shape} & \text{cap-surface} & \text{cap-color} & \text{bruises} & \text{odor} & & \text{class} \\
\mathbf{x}^{(1)} = \langle & \text{bell}, & \text{fibrous}, & \text{gray}, & \text{false}, & \text{foul}, \ldots \rangle & y^{(1)} = & \text{edible} \\
\mathbf{x}^{(2)} = \langle & \text{convex}, & \text{scaly}, & \text{purple}, & \text{false}, & \text{musty}, \ldots \rangle & y^{(2)} = & \text{poisonous} \\
\mathbf{x}^{(3)} = \langle & \text{bell}, & \text{smooth}, & \text{red}, & \text{true}, & \text{musty}, \ldots \rangle & y^{(3)} = & \text{edible} \\
 & & \vdots & & & & & \vdots
\end{array}$$

Standard feature types

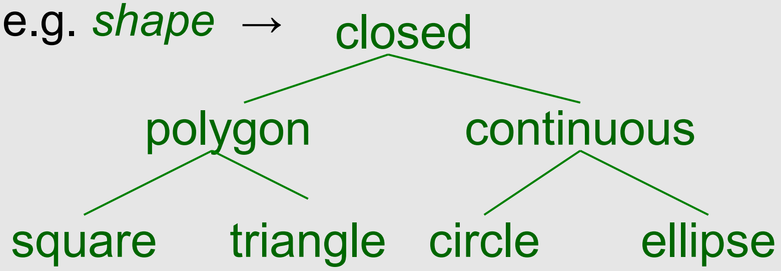
- *nominal* (including Boolean)
  - no ordering among possible values
    e.g. *color* ∈ {*red*, *blue*, *green*} (vs. *color* = *1000* Hertz)

- *ordinal*
  - possible 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* →



closed

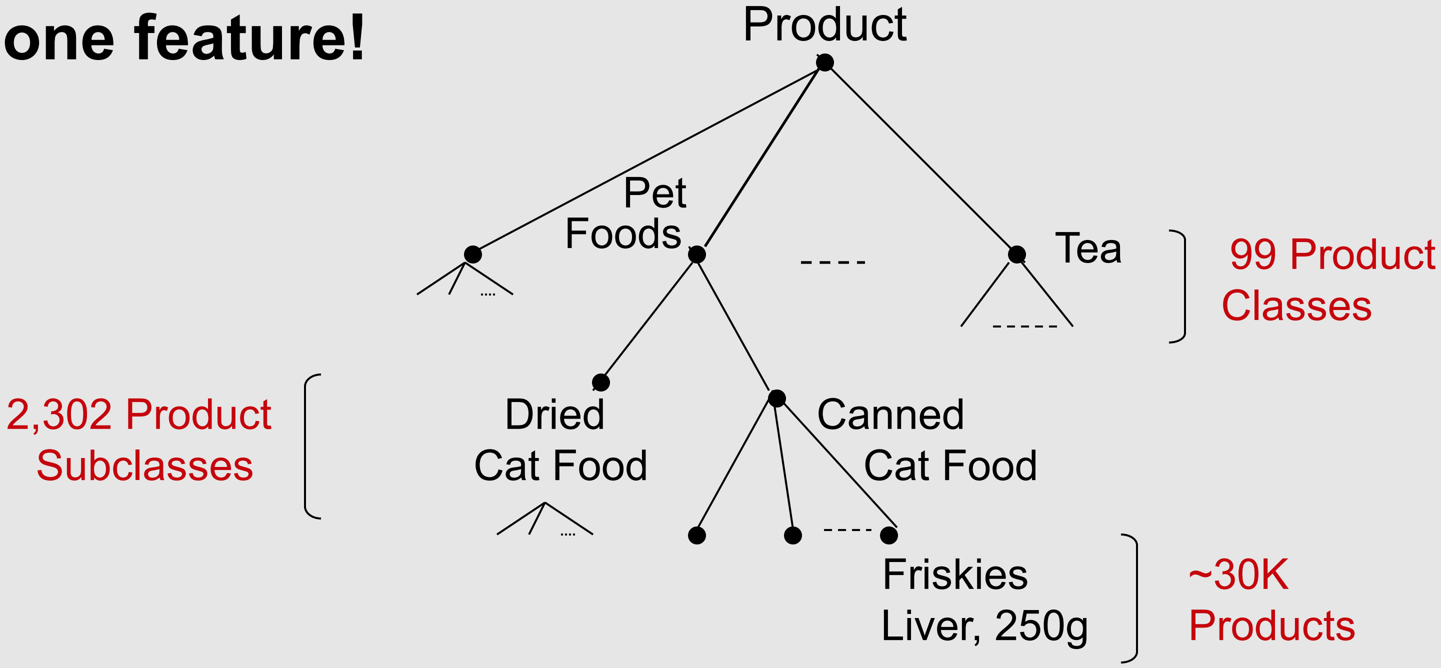
polygon continuous

square triangle circle ellipse

Feature hierarchy example

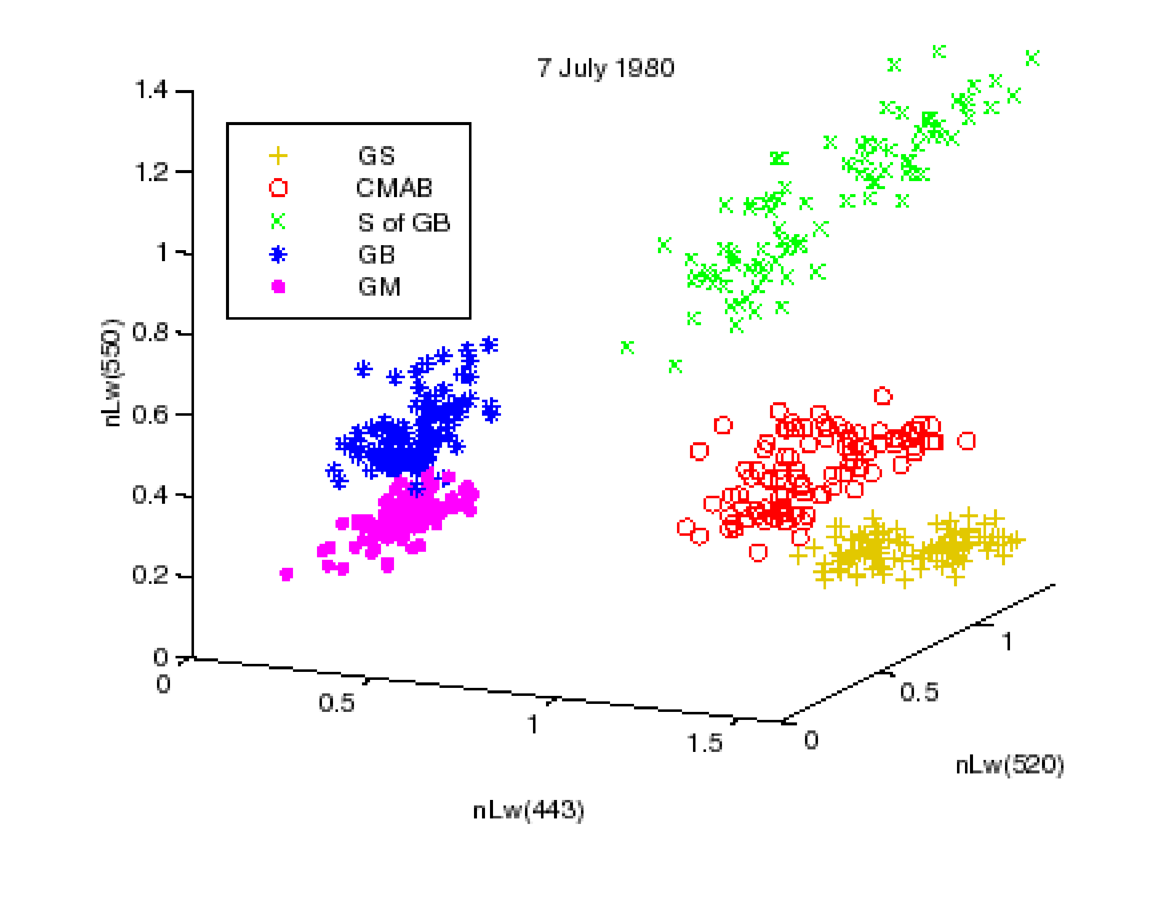
Lawrence et al., *Data Mining and Knowledge Discovery* 5(1-2), 2001

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 feature vector

As a single 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

Learning Settings

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

$$\left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right) \ \ldots \ \left(\mathbf{x}^{(m)}, y^{(m)}\right)$$

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

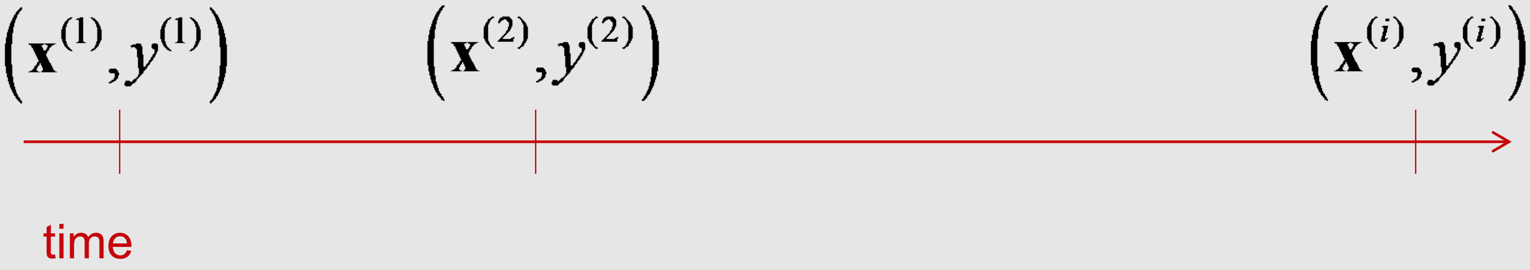
- there are also tasks in which each y is more structured object like a *sequence* of discrete labels (as in e.g. image segmentation, machine translation)

Batch vs. online learning

In batch learning, the learner is given the training set as a batch (i.e. all at once)

$$\left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right) \ldots \left(\mathbf{x}^{(m)}, y^{(m)}\right)$$

In online learning, the learner receives instances sequentially, and updates the model after each (for some tasks it might have to classify/make a prediction for each $\mathbf{x}^{(i)}$ before seeing $y^{(i)}$)



$\left(\mathbf{x}^{(1)}, y^{(1)}\right)$ $\left(\mathbf{x}^{(2)}, y^{(2)}\right)$ $\left(\mathbf{x}^{(i)}, y^{(i)}\right)$

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

- there are also 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

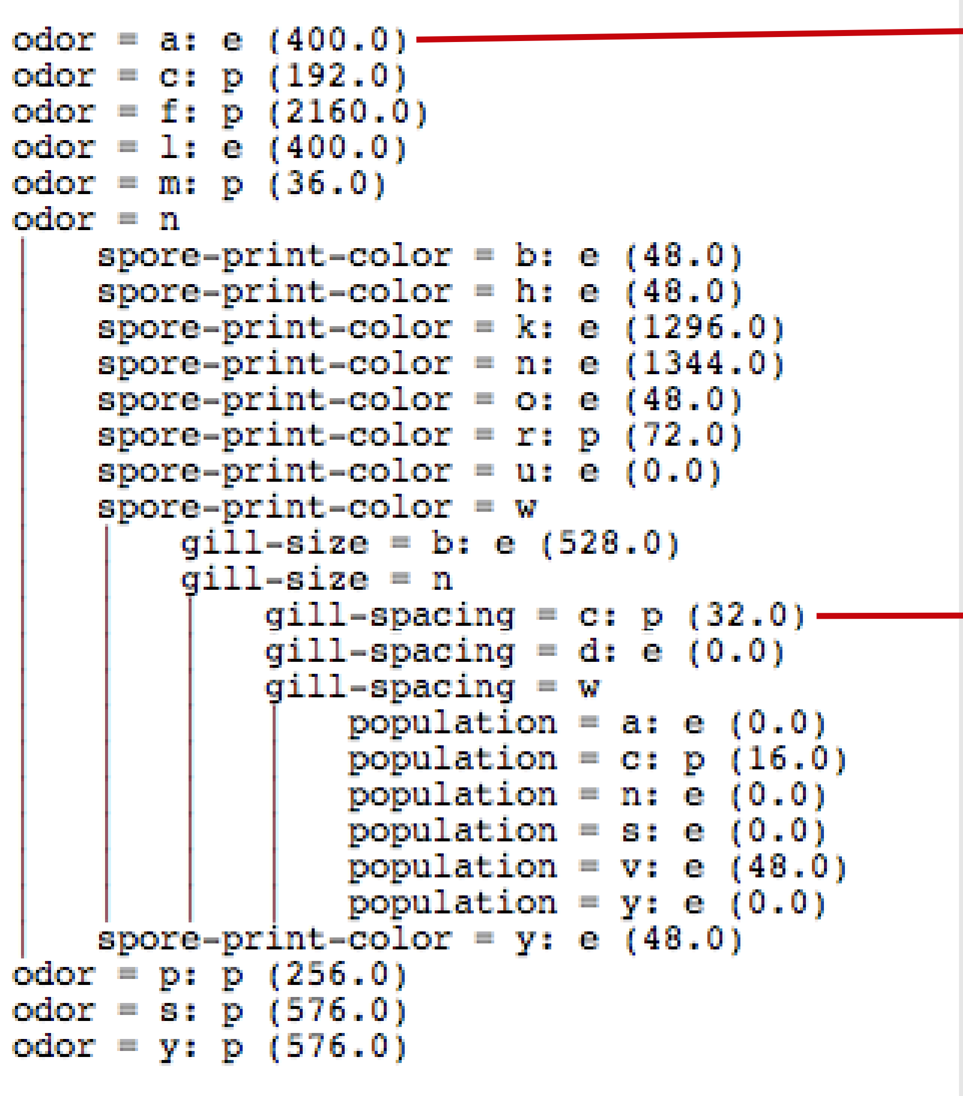
- decision trees
- neural networks
- support vector machines
- Bayesian networks
- ensembles of the above
- etc.

Mushroom features (UCI 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)
```

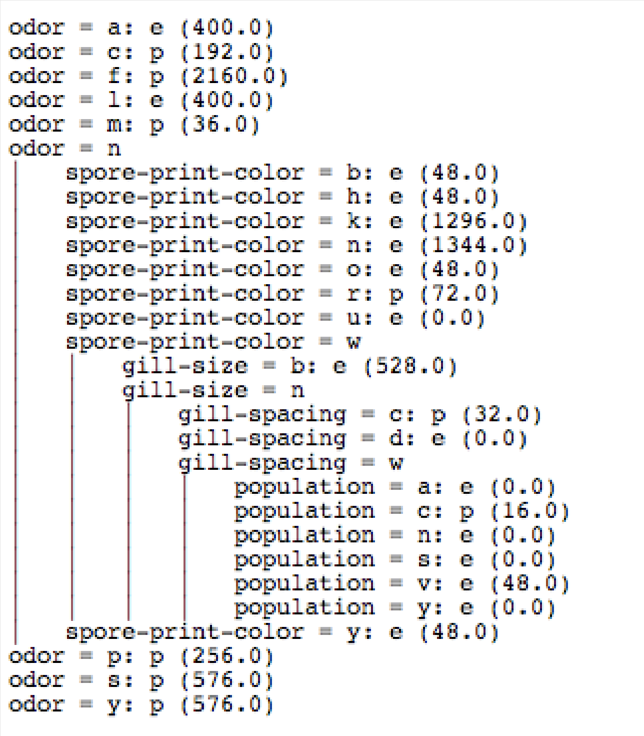
if odor=almond, predict edible

if odor=none ∧
spore-print-color=white ∧
gill-size=narrow ∧
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

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



```
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)
```

$y =$ edible or poisonous?

Unsupervised learning

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

$$\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \ \ldots \ \mathbf{x}^{(m)}$$

goal: discover interesting regularities/structures/patterns that characterize the instances

common unsupervised learning tasks

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

Clustering

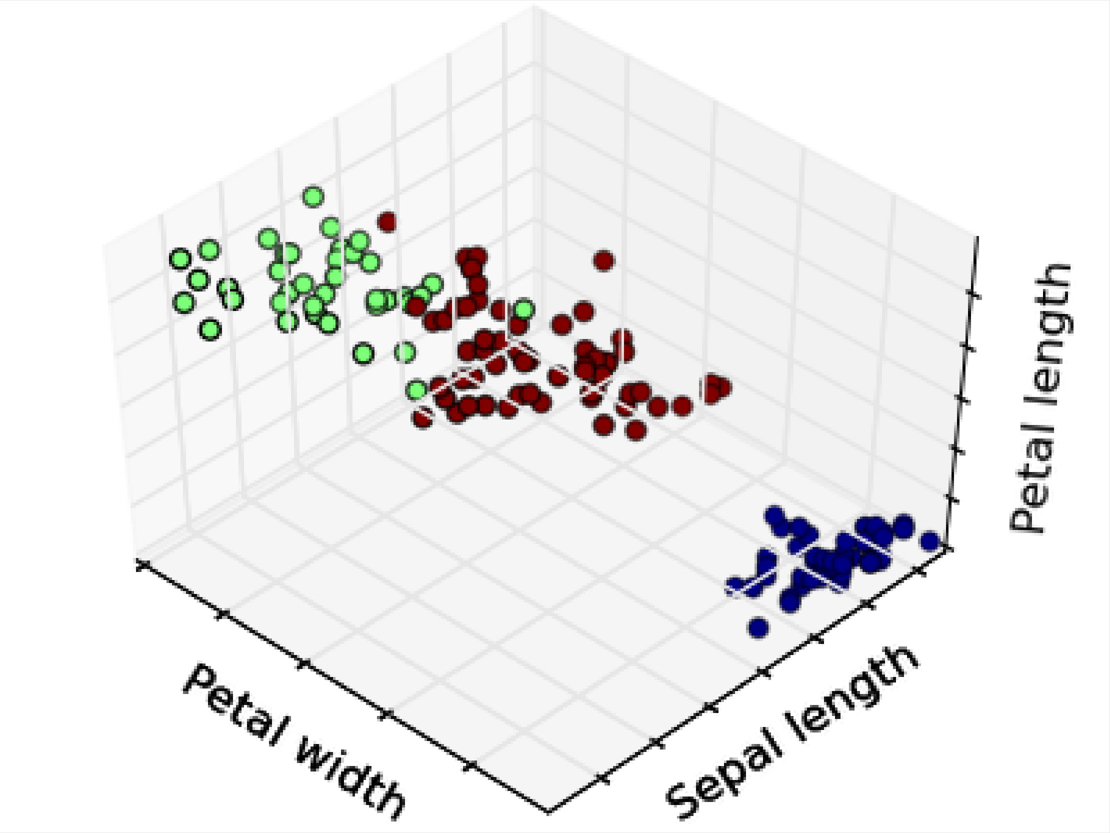
given

- training set of instances $\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \; \ldots \; \mathbf{x}^{(m)}$

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 $\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \; \ldots \; \mathbf{x}^{(m)}$

output

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

performance task

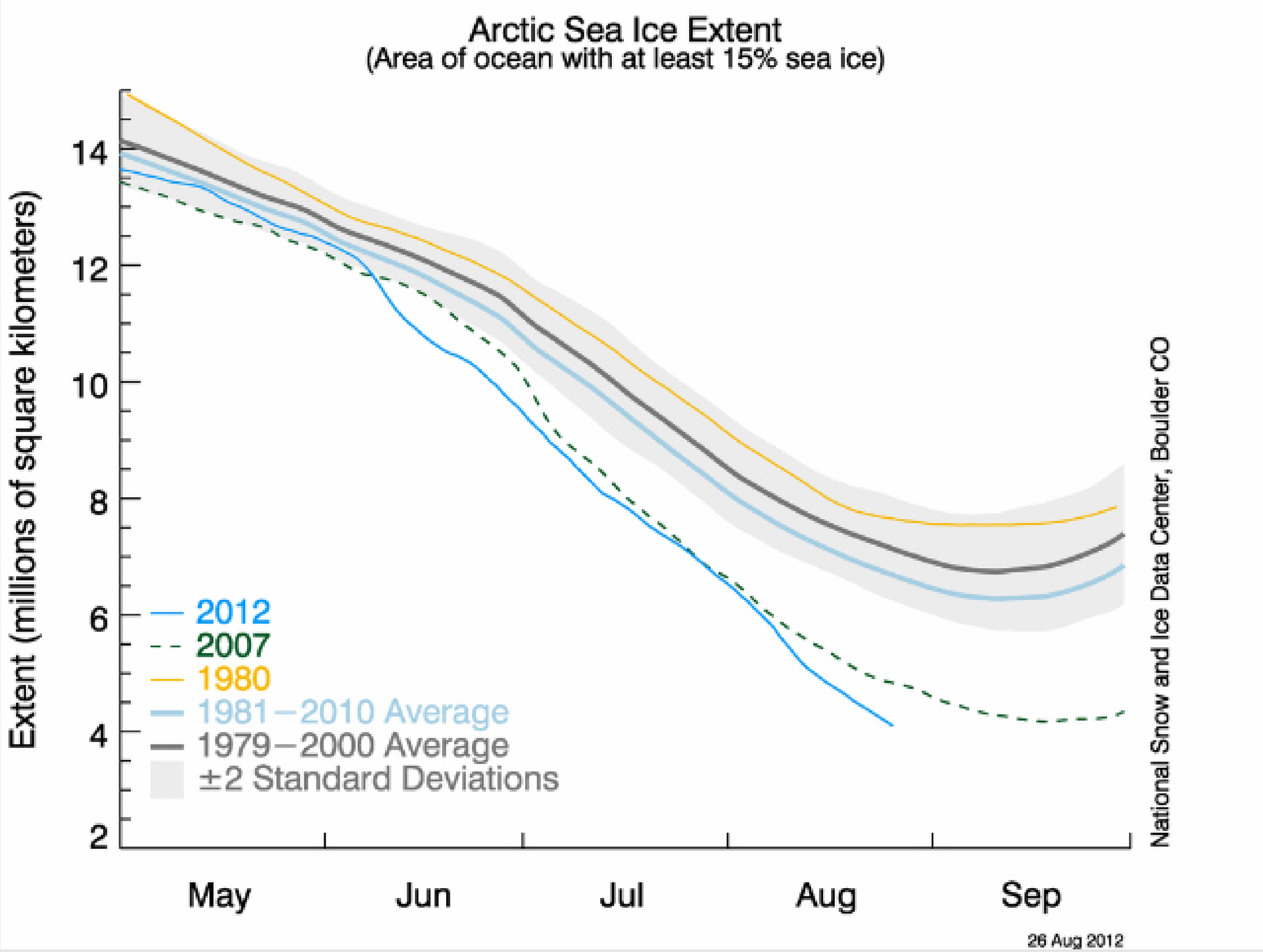
given

- a previously unseen $\boldsymbol{x}$

determine

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

Anomaly detection example



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

Does the data for 2012 look anomalous?

Dimensionality reduction

given

- training set of instances $\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \ \ldots \ \mathbf{x}^{(m)}$

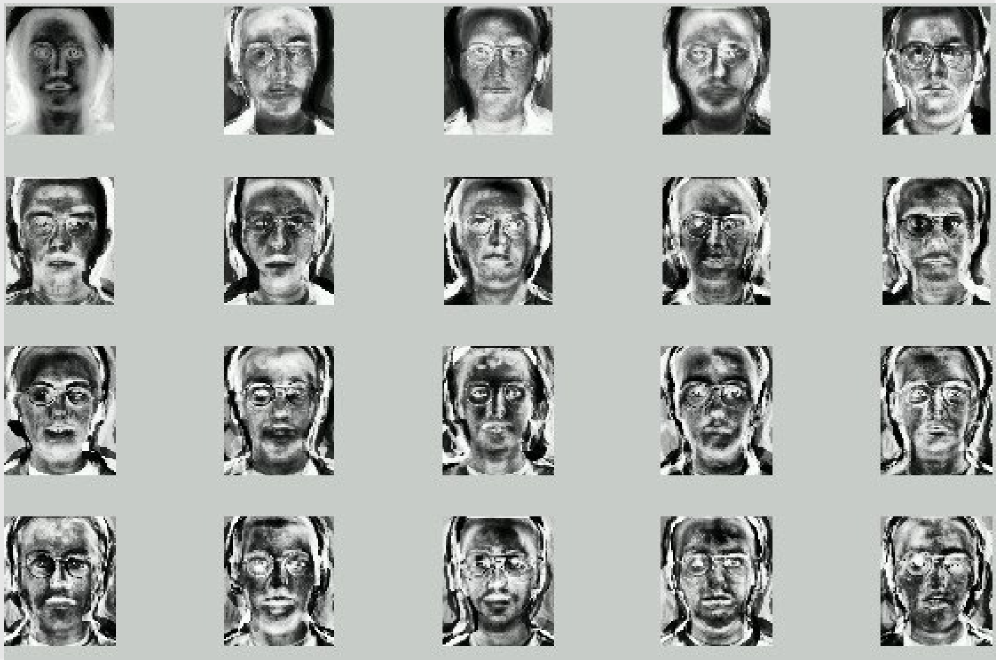
output

- 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*

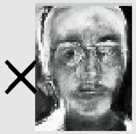
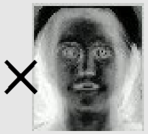


Dimensionality reduction example

represent each face as a linear combination of *eigenfaces*

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

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

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

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

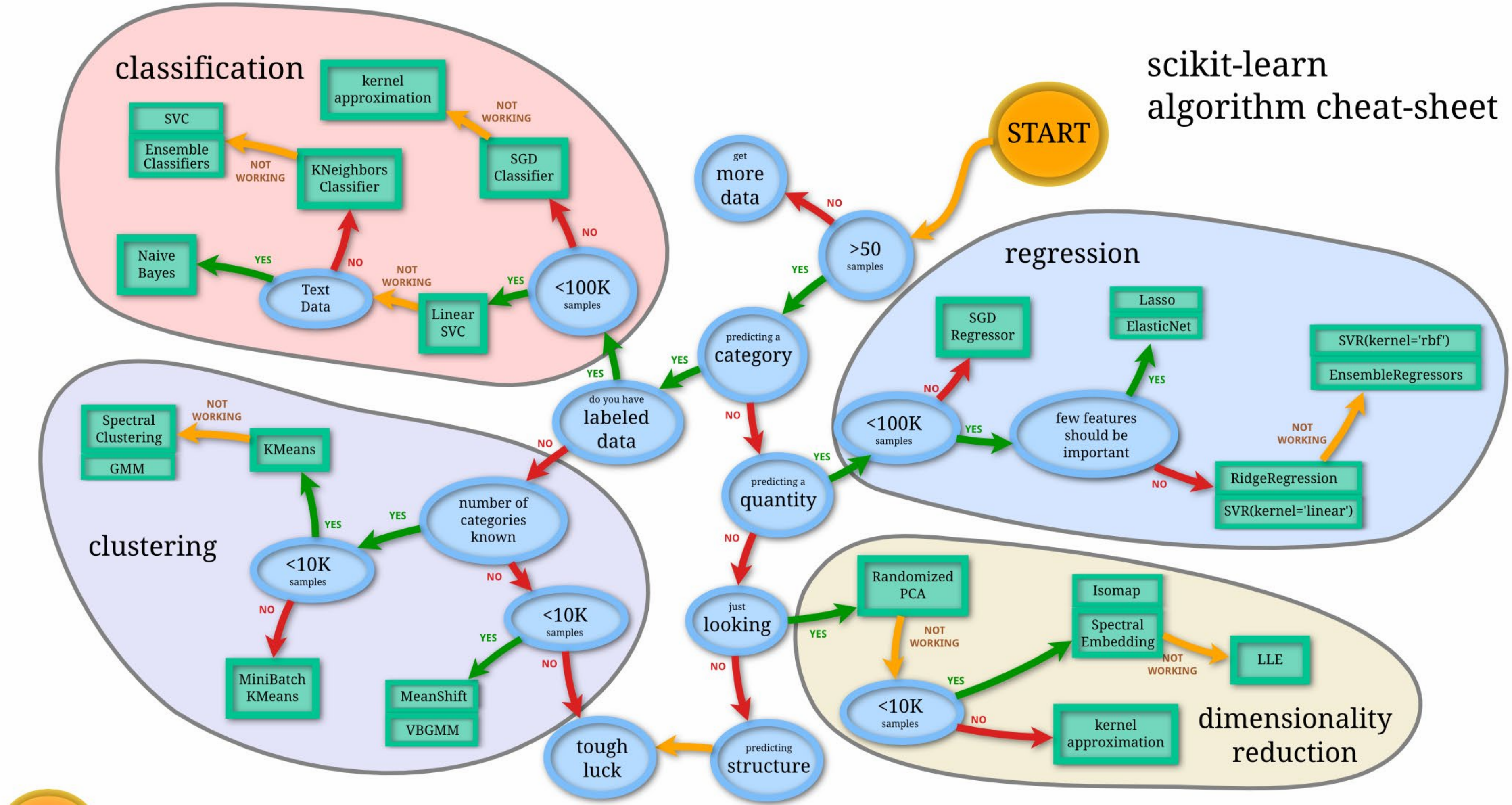
# of features is now 20 instead of # of pixels in images

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.*

Zoo of machine learning models



Note: only a subset of ML methods

Figure from scikit-learn.org

Even a subarea has its own collection

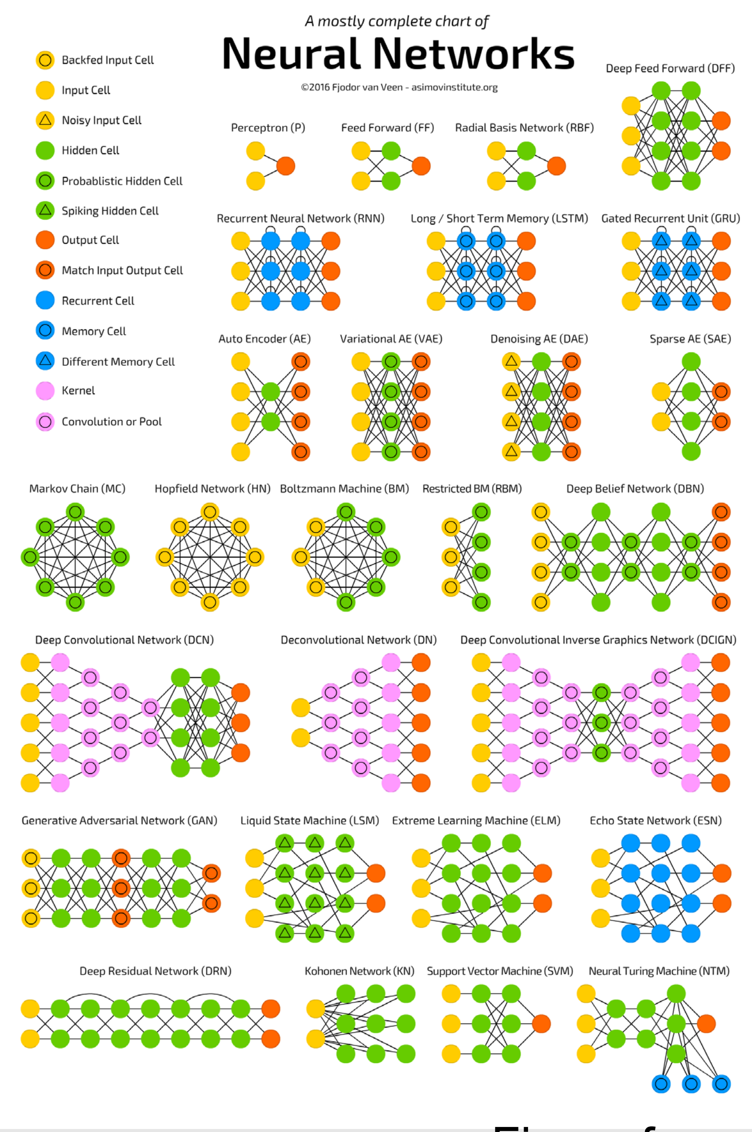
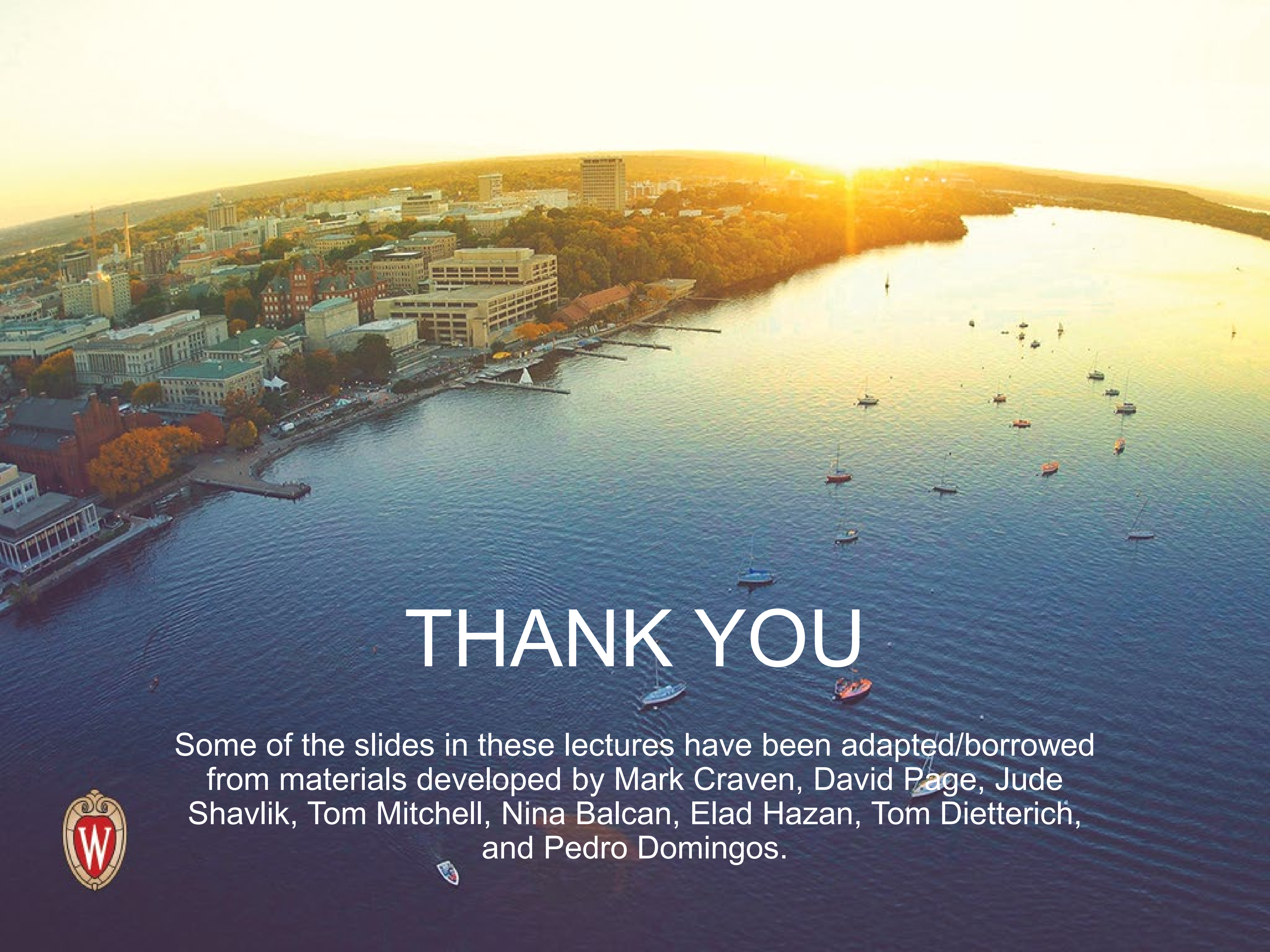

A mostly complete chart of
Neural Networks
©2016 Fjodor van Veen - asimovinstitute.org
Backfed Input Cell
Input Cell
Noisy Input Cell
Hidden Cell
Probablistic Hidden Cell
Spiking Hidden Cell
Output Cell
Match Input Output Cell
Recurrent Cell
Memory Cell
Different Memory Cell
Kernel
Convolution or Pool
Deep Feed Forward (DFF)
Perceptron (P)
Feed Forward (FF)
Radial Basis Network (RBF)
Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Auto Encoder (AE)
Variational AE (VAE)
Denoising AE (DAE)
Sparse AE (SAE)
Markov Chain (MC)
Hopfield Network (HN)
Boltzmann Machine (BM)
Restricted BM (RBM)
Deep Belief Network (DBN)
Deep Convolutional Network (DCN)
Deconvolutional Network (DN)
Deep Convolutional Inverse Graphics Network (DCIGN)
Generative Adversarial Network (GAN)
Liquid State Machine (LSM)
Extreme Learning Machine (ELM)
Echo State Network (ESN)
Deep Residual Network (DRN)
Kohonen Network (KN)
Support Vector Machine (SVM)
Neural Turing Machine (NTM)

Figure from asimovinstitute.org



THANK YOU

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, and Pedro Domingos.