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

\[ x^{(1)} = \langle \text{bell}, \text{fibrous}, \text{gray}, \text{false}, \text{foul}, \ldots \rangle \quad y^{(1)} = \text{edible} \]
\[ x^{(2)} = \langle \text{convex}, \text{scaly}, \text{purple}, \text{false}, \text{musty}, \ldots \rangle \quad y^{(2)} = \text{poisonous} \]
\[ x^{(3)} = \langle \text{bell}, \text{smooth}, \text{red}, \text{true}, \text{musty}, \ldots \rangle \quad y^{(3)} = \text{edible} \]
Standard feature types

• *nominal* (including Boolean)
  • no ordering among possible values
  e.g. $\text{color} \in \{\text{red, blue, green}\}$ (vs. $\text{color} = 1000$ Hertz)

• *ordinal*
  • possible values of the feature are totally ordered
  e.g. $\text{size} \in \{\text{small, medium, large}\}$

• *numeric (continuous)*
  $\text{weight} \in [0...500]$

• *hierarchical*
  • possible values are partially *ordered* in a hierarchy
  e.g. $\text{shape} \rightarrow \text{closed}$

  $$\text{polygon} \rightarrow \text{continuous}$$

  $\text{square} \quad \text{triangle} \quad \text{circle} \quad \text{ellipse}$
Feature hierarchy example

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

Structure of one feature!
Feature space
deep learning

We can think of each instance as representing a point in a \( d \)-dimensional feature space where \( d \) is the number of features.
Another view of feature vector

As a single table

<table>
<thead>
<tr>
<th></th>
<th>feature 1</th>
<th>feature 2</th>
<th>. . .</th>
<th>feature (d)</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance 1</td>
<td>0.0</td>
<td>small</td>
<td></td>
<td>red</td>
<td>true</td>
</tr>
<tr>
<td>instance 2</td>
<td>9.3</td>
<td>medium</td>
<td></td>
<td>red</td>
<td>false</td>
</tr>
<tr>
<td>instance 3</td>
<td>8.2</td>
<td>small</td>
<td></td>
<td>blue</td>
<td>false</td>
</tr>
<tr>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
</tr>
<tr>
<td>instance (n)</td>
<td>5.7</td>
<td>medium</td>
<td></td>
<td>green</td>
<td>true</td>
</tr>
</tbody>
</table>
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
\( \begin{align*}
\text{• training set of instances of unknown target function } f \\
( x^{(1)}, y^{(1)} ), ( x^{(2)}, y^{(2)} ), \ldots , ( x^{(m)}, y^{(m)} )
\end{align*} \)

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)

\[
\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}) \ldots (x^{(m)}, y^{(m)})\}
\]

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 \(x^{(i)}\) before seeing \(y^{(i)}\))

\[
\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots (x^{(i)}, y^{(i)})\}
\]

**time**
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 $x$.

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)

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

sunken is one possible value of the cap-shape feature
A learned decision tree

<table>
<thead>
<tr>
<th>odor</th>
<th>value</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>odor = a:</td>
<td>e</td>
<td>0.400</td>
</tr>
<tr>
<td>odor = c:</td>
<td>p</td>
<td>0.192</td>
</tr>
<tr>
<td>odor = f:</td>
<td>p</td>
<td>0.216</td>
</tr>
<tr>
<td>odor = l:</td>
<td>e</td>
<td>0.400</td>
</tr>
<tr>
<td>odor = m:</td>
<td>p</td>
<td>0.036</td>
</tr>
<tr>
<td>odor = n</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

if odor=almond, predict edible

<table>
<thead>
<tr>
<th>spore-print-color</th>
<th>value</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>spore-print-color = b:</td>
<td>e</td>
<td>0.480</td>
</tr>
<tr>
<td>spore-print-color = h:</td>
<td>e</td>
<td>0.480</td>
</tr>
<tr>
<td>spore-print-color = k:</td>
<td>e</td>
<td>0.1296</td>
</tr>
<tr>
<td>spore-print-color = n:</td>
<td>e</td>
<td>0.1344</td>
</tr>
<tr>
<td>spore-print-color = o:</td>
<td>e</td>
<td>0.480</td>
</tr>
<tr>
<td>spore-print-color = r:</td>
<td>p</td>
<td>0.072</td>
</tr>
<tr>
<td>spore-print-color = u:</td>
<td>e</td>
<td>0.000</td>
</tr>
<tr>
<td>spore-print-color = w:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>gill-size</th>
<th>value</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>gill-size = b:</td>
<td>e</td>
<td>0.528</td>
</tr>
<tr>
<td>gill-size = n</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>gill-spacing</th>
<th>value</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>gill-spacing = c:</td>
<td>p</td>
<td>0.320</td>
</tr>
<tr>
<td>gill-spacing = d:</td>
<td>e</td>
<td>0.000</td>
</tr>
<tr>
<td>gill-spacing = w:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>population</th>
<th>value</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>population = a:</td>
<td>e</td>
<td>0.000</td>
</tr>
<tr>
<td>population = c:</td>
<td>p</td>
<td>0.160</td>
</tr>
<tr>
<td>population = n:</td>
<td>e</td>
<td>0.000</td>
</tr>
<tr>
<td>population = s:</td>
<td>e</td>
<td>0.000</td>
</tr>
<tr>
<td>population = v:</td>
<td>e</td>
<td>0.480</td>
</tr>
<tr>
<td>population = y:</td>
<td>e</td>
<td>0.000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>spore-print-color</th>
<th>value</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>spore-print-color = y:</td>
<td>e</td>
<td>0.480</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>odor</th>
<th>value</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>odor = p:</td>
<td>p</td>
<td>0.256</td>
</tr>
<tr>
<td>odor = s:</td>
<td>p</td>
<td>0.576</td>
</tr>
<tr>
<td>odor = y:</td>
<td>p</td>
<td>0.576</td>
</tr>
</tbody>
</table>
Classification with a learned decision tree

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

\[ x = \langle \text{bell, fibrous, brown, false, foul, ...} \rangle \]

\[ y = \text{edible or poisonous?} \]
Unsupervised learning

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

\[ x^{(1)}, x^{(2)}, \ldots, x^{(m)} \]

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 \( X^{(1)}, X^{(2)} \ldots X^{(m)} \)

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 \( X^{(1)}, X^{(2)} \ldots X^{(m)} \)

**Output**
- Model \( h \in H \) that represents “normal” \( x \)

**Performance task**
- Given
  - A previously unseen \( x \)

**Determine**
- If \( x \) 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 \( \mathbf{x} \) with a lower-dimension feature vector while still preserving key properties of the data
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.
represent each face as a linear combination of *eigenfaces*

\[ \mathbf{x}^{(1)} = \alpha_1^{(1)} \times \mathbf{e}_1 + \alpha_2^{(1)} \times \mathbf{e}_2 + \ldots + \alpha_{20}^{(1)} \times \mathbf{e}_{20} \]

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

\[ \mathbf{x}^{(2)} = \alpha_1^{(2)} \times \mathbf{e}_1 + \alpha_2^{(2)} \times \mathbf{e}_2 + \ldots + \alpha_{20}^{(2)} \times \mathbf{e}_{20} \]

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

# 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.
THANK YOU

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