Announcements
Announcements

• Enrollment:
  • Waitlist is beginning to clear. Email me Thursday if you’re still on it AND have a reason for additional priority.
  • It will be offered next semester if you don’t get in.
Announcements

• Enrollment:
  • Waitlist is beginning to clear. Email me Thursday if you’re still on it AND have a reason for additional priority.
  • It will be offered next semester if you don’t get in.
• Recordings:
  • Available on Canvas. **Disclaimer:** No guarantee of availability. May not capture slide annotations.
Announcements

• Enrollment:
  • Waitlist is beginning to clear. Email me Thursday if you’re still on it AND have a reason for additional priority.
  • It will be offered next semester if you don’t get in.
• Recordings:
  • Available on Canvas. **Disclaimer:** No guarantee of availability. May not capture slide annotations.
• Background Knowledge:
  • Please look at homework 1 before add/drop deadline.
  • Please take background survey on Piazza.
Announcements

• Enrollment:
  • Waitlist is beginning to clear. Email me Thursday if you’re still on it AND have a reason for additional priority.
  • It will be offered next semester if you don’t get in.

• Recordings:
  • Available on Canvas. **Disclaimer:** No guarantee of availability. May not capture slide annotations.

• Background Knowledge:
  • Please look at homework 1 before add/drop deadline.
  • Please take background survey on Piazza.

• Homework 1 is due at 9:30 AM on Tuesday, September 19.
Office Hours
Office Hours

• My office hours are Tuesdays from 11 — 12pm in CS 5391.
  • Or by appointment.
  • I will meet students in the hall after lecture at 10:45 for quick questions and then walk back to my office.
  • If you need a longer discussion, please wait to either walk with me or meet me at my office.
Today’s Learning Outcomes
Today’s Learning Outcomes

• After today’s lecture:
Today’s Learning Outcomes

• After today’s lecture:
  • You will be able to explain the key aspects of a supervised learning problem.
Today’s Learning Outcomes

• **After today’s lecture:**
  • You will be able to explain the key aspects of a supervised learning problem.
  • Provide examples of unsupervised learning problems and explain why these are not supervised learning problems.
Today’s Learning Outcomes

• After today’s lecture:
  • You will be able to explain the key aspects of a supervised learning problem.
  • Provide examples of unsupervised learning problems and explain why these are not supervised learning problems.
  • Explain key challenges of reinforcement learning problems.
Outline

• Review from last time
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning

• Supervised learning concepts
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning

• Supervised learning concepts
  • Features, models, training, other terminology
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning

• Supervised learning concepts
  • Features, models, training, other terminology

• Unsupervised learning concepts
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning

• Supervised learning concepts
  • Features, models, training, other terminology

• Unsupervised learning concepts
  • Clustering, anomaly detection, dimensionality reduction
Outline

• **Review from last time**
  • Supervised, unsupervised, reinforcement learning

• **Supervised learning concepts**
  • Features, models, training, other terminology

• **Unsupervised learning concepts**
  • Clustering, anomaly detection, dimensionality reduction

• **Reinforcement learning concepts**
Outline

• **Review from last time**
  • Supervised, unsupervised, reinforcement learning

• **Supervised learning concepts**
  • Features, models, training, other terminology

• **Unsupervised learning concepts**
  • Clustering, anomaly detection, dimensionality reduction

• **Reinforcement learning concepts**
  • Exploration vs. Exploitation, credit-assignment.
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning

• Supervised learning concepts
  • Features, models, training, other terminology

• Unsupervised learning concepts
  • Clustering, anomaly detection, dimensionality reduction

• Reinforcement learning concepts
  • Exploration vs. Exploitation, credit-assignment.
Review: ML Overview: Definition
Review: ML Overview: Definition

What is machine learning?
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
ML Overview: Flavors

Supervised Learning
ML Overview: Flavors

Supervised Learning

• Learning from labelled examples.
ML Overview: Flavors

Supervised Learning
• Learning from labelled examples.

• Workflow:
  • Collect a set of examples {data point, label}: training set
  • “Train” a model to match data points to labels.
  • “Test” it on new, unseen data points.
ML Overview: Flavors

Supervised Learning

• Learning from labelled examples.

• **Workflow:**
  
  • Collect a set of examples \{data point, label\}: **training set**
  
  • “Train” a model to match data points to labels.
  
  • “Test” it on new, unseen data points.

• **Image classification:**

  indoor

  outdoor
ML Overview: Flavors

Supervised Learning

• **Example: Image classification**
ML Overview: Flavors

Supervised Learning

- **Example: Image classification**
- Recall Task/Performance measure/Experience definition
ML Overview: Flavors

Supervised Learning

- **Example: Image classification**
- Recall Task/Performance measure/Experience definition
  - Task: distinguish **indoor** vs **outdoor**
ML Overview: Flavors

Supervised Learning

• **Example: Image classification**

• Recall **Task/Performance measure/Experience definition**
  - **Task:** distinguish *indoor* vs *outdoor*
  - **Performance measure:** probability of misclassifying
ML Overview: Flavors

Supervised Learning

• **Example: Image classification**

• Recall **Task/Performance measure/Experience definition**
  - **Task**: distinguish *indoor* vs *outdoor*
  - **Performance measure**: probability of misclassifying
  - **Experience**: labeled examples
ML Overview: Flavors

Unsupervised Learning
ML Overview: Flavors

Unsupervised Learning

• Data, but no labels. No input/output.
ML Overview: Flavors

Unsupervised Learning

• Data, but no labels. No input/output.
• Goal: find some structure in the dataset
ML Overview: Flavors

Unsupervised Learning

• Data, but no labels. No input/output.
• Goal: find some structure in the dataset

• Workflow:
Unsupervised Learning

• Data, but no labels. No input/output.
• Goal: find some structure in the dataset

• **Workflow:**
  • Collect a set {data points}
ML Overview: Flavors

Unsupervised Learning
• Data, but no labels. No input/output.
• Goal: find some structure in the dataset

• Workflow:
  • Collect a set \{data points\}
  • Perform some algorithm on it
ML Overview: Flavors

Unsupervised Learning

• Data, but no labels. No input/output.
• Goal: find some structure in the dataset

• Workflow:
  • Collect a set \{data points\}
  • Perform some algorithm on it
ML Overview: Flavors

Unsupervised Learning

• Example: Clustering
ML Overview: Flavors

Unsupervised Learning

• Example: Clustering
  • Task: produce distinct clusters for a set of data
ML Overview: Flavors

Unsupervised Learning

• **Example: Clustering**
  - Task: produce distinct clusters for a set of data
  - Performance measure: closeness to underlying structure
ML Overview: Flavors

Unsupervised Learning

• **Example: Clustering**
  - **Task:** produce distinct clusters for a set of data
  - **Performance measure:** closeness to underlying structure
  - **Experience:** available datapoints
ML Overview: Flavors

Reinforcement Learning
ML Overview: Flavors

Reinforcement Learning

• Agent interacting with the world; gets rewards for actions
ML Overview: Flavors

Reinforcement Learning

• Agent interacting with the world; gets rewards for actions
• Goal: learn to perform some activity
ML Overview: Flavors

Reinforcement Learning
• Agent interacting with the world; gets rewards for actions
• Goal: learn to perform some activity
• Workflow:
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
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 (mapping from environment states to actions) to maximize rewards.
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 (mapping from environment states to actions) to maximize rewards.
  • **Deploy** in new environment
ML Overview: Flavors

Reinforcement Learning

• Example: Controlling aircraft
ML Overview: Flavors

Reinforcement Learning

• **Example: Controlling aircraft**
  • **Task:** keep the aircraft in the air, steer towards a desired goal
ML Overview: Flavors

Reinforcement Learning

• **Example: Controlling aircraft**
  • Task: keep the aircraft in the air, steer towards a desired goal
  • Performance measure: reward for reaching goal quickly
ML Overview: Flavors

Reinforcement Learning

• **Example: Controlling aircraft**
  • Task: keep the aircraft in the air, steer towards a desired goal
  • Performance measure: reward for reaching goal quickly
  • Experience: data (state/action/reward) from previous flights
Break & Quiz
Q1-1: Which of the following is generally NOT a supervised learning task?

1. Predicting house prices from past home sales.
2. Email spam detection
3. Handwriting recognition
4. Eigenvalue calculation
Q1-1: Which of the following is generally NOT a supervised learning task?

1. Predicting house prices from past home sales.
2. Email spam detection
3. Handwriting recognition
4. **Eigenvalue calculation**

Eigenvalue calculation is a mathematical problem, and we do not have any labels for this problem.
Outline

• **Review from last time**
  • Supervised, unsupervised, reinforcement learning

• **Supervised learning concepts**
  • Features, models, training, other terminology

• **Unsupervised learning concepts**
  • Clustering, anomaly detection, dimensionality reduction

• **Reinforcement learning concepts**
  • Exploration vs. Exploitation, credit-assignment.
Supervised Learning

• Can I eat this?
Supervised Learning

• Can I eat this?

• Safe or poisonous?
Supervised Learning

- Can I eat this?
- Safe or poisonous?
- Never seen it before
Supervised Learning

• Can I eat this?

• Safe or poisonous?

• Never seen it before

• How to decide?
Supervised Learning: Training Instances
Supervised Learning: Training Instances

• I know about other mushrooms:

• Training set of *labeled examples/instances/labeled data*
Supervised Learning: Training Instances

• I know about other mushrooms:

  safe

• Training set of labeled examples/instances/labeled data
Supervised Learning: Training Instances

• I know about other mushrooms:

  safe

  poisonous

• Training set of labeled examples/instances/labeled data
Supervised Learning: Formal Setup

Problem setting:
Supervised Learning: Formal Setup

Problem setting:
• Set of possible instances $\mathcal{X}$
Supervised Learning: Formal Setup

Problem setting:
- Set of possible instances $\mathcal{X}$
- Unknown *target function* $f : \mathcal{X} \rightarrow \mathcal{Y}$
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): \( \mathcal{H} = \{ h \mid h : \mathcal{X} \rightarrow \mathcal{Y} \} \)
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): \( \mathcal{H} = \{ h | h : \mathcal{X} \rightarrow \mathcal{Y} \} \)

Given:
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): $\mathcal{H} = \{h | h: \mathcal{X} \rightarrow \mathcal{Y}\}$

Given:
- Training set of instances for unknown target function, where $y^{(i)} \approx f(x^{(i)})$
  
  $$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$$
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): \( \mathcal{H} = \{ h | h: \mathcal{X} \rightarrow \mathcal{Y} \} \)

Given:
- Training set of instances for unknown target function, where \( y^{(i)} \approx f(x^{(i)}) \)
  \( (x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)}) \)
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): $\mathcal{H} = \{ h | h : \mathcal{X} \rightarrow \mathcal{Y} \}$

Given:
- Training set of instances for unknown target function,
  where $y^{(i)} \approx f(x^{(i)})$
  $$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$$
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): \( \mathcal{H} = \{ h | h : \mathcal{X} \rightarrow \mathcal{Y} \} \)

Given:

- Training set of instances for unknown target function, where \( y^{(i)} \approx f(x^{(i)}) \)

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

safe  poisonous  safe
Supervised Learning: Formal Setup
Supervised Learning: Formal Setup

Problem setting
Supervised Learning: Formal Setup

Problem setting
- Set of possible instances $\mathcal{X}$
Supervised Learning: Formal Setup

Problem setting
- Set of possible instances $\mathcal{X}$
- Unknown target function $f : \mathcal{X} \rightarrow \mathcal{Y}$
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*) \( \mathcal{H} = \{ h | h : \mathcal{X} \rightarrow \mathcal{Y} \} \)
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) \( \mathcal{H} = \{ h | h : \mathcal{X} \rightarrow \mathcal{Y} \} \)

Given:
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) $\mathcal{H} = \{ h \mid h: \mathcal{X} \rightarrow \mathcal{Y} \}$

Given:

- Training set of instances for unknown target function $f$,
  
  $$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$$
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) \( \mathcal{H} = \{ h | h : \mathcal{X} \rightarrow \mathcal{Y} \} \)

Given:

- 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
Supervised Learning: Objects

Three types of sets
- Input space, output space, hypothesis class
Supervised Learning: Objects

Three types of sets

• Input space, output space, hypothesis class

\[ \mathcal{X}, \mathcal{Y}, \mathcal{H} \]
Supervised Learning: Objects

Three types of sets

- Input space, output space, hypothesis class
  \[ \mathcal{X}, \mathcal{Y}, \mathcal{H} \]

- Examples:
Supervised Learning: Objects

Three types of sets
- Input space, output space, hypothesis class
  \( \mathcal{X}, \mathcal{Y}, \mathcal{H} \)

- Examples:
  - Input space: feature vectors
Supervised Learning: Objects

Three types of sets
- Input space, output space, hypothesis class
  \[ \mathcal{X}, \mathcal{Y}, \mathcal{H} \]

- Examples:
  - Input space: feature vectors
  \[ \mathcal{X} \subseteq \mathbb{R}^d \]
Supervised Learning: Objects

Three types of sets
- Input space, output space, hypothesis class
  \( \mathcal{X}, \mathcal{Y}, \mathcal{H} \)
- Examples:
  - Input space: feature vectors
    \( \mathcal{X} \subseteq \mathbb{R}^d \)
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:
Supervised Learning: Objects

Three types of sets
- Input space, output space, hypothesis class
  \[ X, Y, \mathcal{H} \]

- Examples:
  - Input space: feature vectors
    \[ X \subseteq \mathbb{R}^d \]
  - Output space:
    - Binary classification
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 classification
    \[ \mathcal{Y} = \{-1, +1\} \]
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 classification
      \( \mathcal{Y} = \{-1, +1\} \)
      safe  poisonous
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 classification
      \[ \mathcal{Y} = \{-1, +1\} \]  
    - Continuous

Examples:
- Input space: feature vectors
- Output space:
  - Binary classification
    - safe
    - poisonous
Supervised Learning: Objects

Three types of sets
- Input space, output space, hypothesis class
  \[ X, Y, \mathcal{H} \]

Examples:
- Input space: feature vectors
  \[ X \subseteq \mathbb{R}^d \]
- Output space:
  - Binary classification
    \[ Y = \{-1, +1\} \]
  - Continuous
    \[ Y \subseteq \mathbb{R} \]
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 classification
      \[ \mathcal{Y} = \{-1, +1\} \]
    - Continuous
      \[ \mathcal{Y} \subseteq \mathbb{R} \]
Input Space: Feature Vectors

• Need a way to represent instance information:
Input Space: Feature Vectors

• Need a way to represent instance information:

\[ x^{(1)} = \langle \text{bell, fibrous, gray, false, foul} \rangle \]
Input Space: Feature Vectors

• Need a way to represent instance information:

\[ x^{(1)} = \langle \text{bell, fibrous, gray, false, foul} \rangle \]
**Input Space: Feature Vectors**

- Need a way to represent instance information:

\[ x^{(1)} = \langle \text{bell}, \text{fibrous}, \text{gray}, \text{false}, \text{foul} \rangle \]
Input Space: Feature Vectors

• Need a way to represent instance information:

\[ x^{(1)} = \langle \text{bell, fibrous, gray, false, foul} \rangle \]
**Input Space:** Feature Vectors

- Need a way to represent instance information:

\[ x^{(1)} = \langle \text{bell}, \text{fibrous}, \text{gray}, \text{false}, \text{foul} \rangle \]
Input Space: Feature Vectors

• Need a way to represent instance information:

\[ x^{(1)} = \langle \text{bell, fibrous, gray, false, foul} \rangle \]
Input Space: Feature Vectors

- Need a way to represent instance information:

\[ x^{(1)} = \langle \text{bell}, \text{fibrous}, \text{gray}, \text{false}, \text{foul} \rangle \]
Input Space: Feature Vectors

• Need a way to represent instance information:

\[ x^{(1)} = \langle \text{bell, fibrous, gray, false, foul} \rangle \]

• For each instance, store features as a vector.
**Input Space: Feature Vectors**

- Need a way to represent instance information:

  - For each instance, store features as a vector.

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

- For each instance, store features as a vector.

  - What kinds of features can we have?
Input Space: Feature Types
Input Space: Feature Types

• *nominal* (including Boolean)
  • no ordering among values (e.g. $\text{animal} \in \{\text{dog, cat, fish}\}$)
Input Space: Feature Types

• *nominal* (including Boolean)
  • no ordering among values (e.g. \textit{animal} \in \{\textit{dog, cat, fish}\})

• *ordinal*
  • values of the feature are totally ordered (e.g. \textit{size} \in \{\textit{small, medium, large}\})
Input Space: Feature Types

• **nominal** (including Boolean)
  - no ordering among values (e.g. \( \text{animal} \in \{\text{dog, cat, fish}\} \))

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

• **numeric** (continuous)
  - \( \text{height} \in [0, 100] \) inches
Input Space: Feature Types

- **nominal** (including Boolean)
  - no ordering among values (e.g. $\text{animal} \in \{\text{dog, cat, fish}\}$)

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

- **numeric** (continuous)
  - $\text{height} \in [0, 100]$ inches

- **hierarchical**
  - possible values are partially ordered in a hierarchy, e.g. $\text{shape}$
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, purple=u, red=e, white=w, yellow=y

stalk-color-below-ring: brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, purple=u, 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=n, urban=u, waste=w, woods=d
Input Space: Feature Spaces
Input Space: Feature Spaces

• *If all features are numeric*, we can think of each instance as a point in a $d$-dimensional Euclidean feature space where $d$ is the number of features.
Input Space: Feature Spaces

• *If all features are numeric*, we can think of each instance as a point in a $d$-dimensional Euclidean feature space where $d$ is the number of features.

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

Output space: Classification vs. Regression

Choices of $\mathcal{Y}$ have special names:
Output space: Classification vs. Regression

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

- Discrete: “classification”. The elements of $\mathcal{Y}$ are classes
Output space: Classification vs. Regression

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

- Discrete: "classification". The elements of $\mathcal{Y}$ are **classes**
  - Note: binary classification is special case when there are two classes.
Choices of $\mathcal{Y}$ have special names:

- Discrete: "classification". The elements of $\mathcal{Y}$ are **classes**
  - Note: binary classification is special case when there are two classes.
Output space: Classification vs. Regression

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

• Discrete: “classification”. The elements of $\mathcal{Y}$ are classes
  • Note: binary classification is special case when there are two classes.

• Continuous: “regression”
Output space: Classification vs. Regression

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

- Discrete: “classification”. The elements of $\mathcal{Y}$ are classes
  - Note: binary classification is special case when there are two classes.

- Continuous: “regression”
  - Example: linear regression
Output space: Classification vs. Regression

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

- **Discrete**: “classification”. The elements of $\mathcal{Y}$ are **classes**
  - Note: binary classification is special case when there are two classes.

- **Continuous**: “regression”
  - Example: linear regression

![Images of flowers: Versicolor, Setosa, Virginica]

![Scatter plot example]
Output space: Classification vs. Regression

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

• Discrete: “classification”. The elements of $\mathcal{Y}$ are classes
  • Note: binary classification is a special case when there are two classes.

• Continuous: “regression”
  • Example: linear regression

• There are other types...
Hypothesis class

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

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

- Recall: hypothesis class / model space.
Hypothesis class

We have talked about $X$, $Y$ what about $H$?

• Recall: hypothesis class / model space.
  • Theoretically, could be all maps from $X$ to $Y$
Hypothesis class

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

- Recall: hypothesis class / model space.
  - Theoretically, could be all maps from $\mathcal{X}$ to $\mathcal{Y}$
  - Does not work! We’ll see why later.
Hypothesis class

We have talked about $X$, $Y$ what about $H$?

• Recall: hypothesis class / model space.
  • Theoretically, could be all maps from $X$ to $Y$
  • Does not work! We’ll see why later.

• Instead, pick specific class of models. E.g. linear models:
Hypothesis class

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

- Recall: hypothesis class / model space.
  - Theoretically, could be all maps from $\mathcal{X}$ to $\mathcal{Y}$
  - Does not work! We’ll see why later.

- Instead, pick specific class of models. E.g. linear models:

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

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

• Recall: hypothesis class / model space.
  • Theoretically, could be all maps from $\mathcal{X}$ to $\mathcal{Y}$
  • Does not work! We’ll see why later.

• Instead, pick specific class of models. E.g. 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

\[ h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d \]
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)
Hypothesis class: Linear Functions

**Example** class of models: linear models

\[ h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d \]
Hypothesis class: Linear Functions

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

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

- How many linear functions are there?
Hypothesis class: Linear Functions

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

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

- How many linear functions are there?
  - Can any function be fit by a linear model?
Hypothesis class: Other Examples
Hypothesis class: Other Examples

Example classes of models: neural networks
Hypothesis class: Other Examples

Example classes of models: neural networks
Hypothesis class: Other Examples

Example classes of models: neural networks

\[ f^{(k)}(x) = \sigma(W_k^T f^{(k-1)}(x)) \]
Hypothesis class: Other Examples

Example classes of models: neural networks

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

Feedforward network
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:
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
**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
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?
Back to 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 \( f \),

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

Goal: model \( h \) that best approximates \( f \)
Supervised Learning: Training
Supervised Learning: Training

Goal: find model $h$ that best approximates $f$
Supervised Learning: Training

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

- One way: empirical risk minimization (ERM)
Supervised Learning: Training

**Goal:** find model \( h \) that best approximates \( f \)

- One way: empirical risk minimization (ERM)

\[
\hat{f} = \underset{h \in \mathcal{H}}{\text{arg min}} \frac{1}{n} \sum_{i=1}^{n} \ell(h(x^{(i)}), y^{(i)})
\]
Supervised Learning: Training

Goal: find model $h$ that best approximates $f$

• One way: empirical risk minimization (ERM)

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

**Goal:** find 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

Model prediction


Supervised Learning: Training

**Goal:** find 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
- Model prediction
- Loss function: how far is the prediction from the label?
Batch vs. Online Learning
Batch vs. Online Learning

- **Batch learning**: get all your instances at once
Batch vs. Online Learning

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

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

\[\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}\]
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
  \[\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\} \quad \{(x^{(m+1)}, y^{(m+1)})\}\]
Supervised Learning: Predicting

Now that we have our learned model, we can use it for predictions.
Supervised Learning: Predicting

Now that we have our learned model, we can use it for predictions.
Supervised Learning: Predicting

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

\[ x = \{\text{bell, fibrous, brown, false, foul, ...}\} \]
Supervised Learning: Predicting

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

\[ x = \{\text{bell, fibrous, brown, false, foul, ...}\} \]
Supervised Learning: Predicting

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

\[ x = \{ \text{bell, fibrous, brown, false, foul, ...} \} \]
Recall supervised learning workflow
Recall supervised learning workflow

- Collect a set of examples \{data, labels\}: \textit{training set}
Recall supervised learning workflow

- Collect a set of examples \{data, labels\}: \textit{training set}
Recall supervised learning workflow

• Collect a set of examples \{data, labels\}: \textit{training set}
  
• “\textit{Train}” a model to match these examples
  
  • E.g. Choose a hypothesis class and perform ERM
Recall supervised learning workflow

• Collect a set of examples \{data, labels\}: \textit{training set} \checkmark

• “\textit{Train}” a model to match these examples
  • E.g. Choose a hypothesis class and perform ERM

• “\textit{Test}” it on new data
Recall supervised learning workflow

• Collect a set of examples \{data, labels\}: training set

• “Train” a model to match these examples
  • E.g. Choose a hypothesis class and perform ERM

• “Test” it on new data
Recall supervised learning workflow

• Collect a set of examples \{data, labels\}: **training set**

• "Train" a model to match these examples
  • E.g. Choose a hypothesis class and perform ERM

• "Test" it on new data
Recall supervised learning workflow

• Collect a set of examples \{data, labels\}: **training set**

• “Train” a model to match these examples
  • E.g. Choose a **hypothesis class** and perform **ERM**

\[
\hat{f} = \operatorname{arg~min}_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(h(x^{(i)}), y^{(i)})
\]

  - **Model prediction**
  - **Hypothesis Class**
  - **Loss function**

• “Test” it on new data
From linear to polynomial regression
From linear to polynomial regression

Another class of models: polynomials:
From linear to polynomial regression

Another class of models: polynomials:

\[ h_\theta(x) = \theta_d x^d + \theta_{d-1} x^{d-1} + \ldots + \theta_1 x + \theta_0 \]
From linear to polynomial regression

Another class of models: polynomials:

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

• We can get a perfect fit by setting \( d \) to be very large.
  • E.g. In 1D, set \( d = n-1 \)
From linear to polynomial regression

Another class of models: polynomials:

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

• We can get a perfect fit by setting \( d \) to be very large.
  • E.g. In 1D, set \( d = n-1 \)

• So, are we done?
From linear to polynomial regression

Another class of models: polynomials:

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

• We can get a perfect fit by setting \( d \) to be very large.
  • E.g. In 1D, set \( d = n-1 \)

• So, are we done?
  • How sensitive to noise?
From linear to polynomial regression

Another class of models: polynomials:

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

• We can get a perfect fit by setting \( d \) to be very large.
  • E.g. In 1D, set \( d = n-1 \)

• So, are we done?
  • How sensitive to noise?
  • How will they **extrapolate**?
Generalization

Fitting data isn’t the only task, we want to generalize.
Generalization

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

• Apply learned model to unseen data:
Generalization

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

- Apply learned model to unseen data:
  - Underlying data distribution:
Generalization

Fitting data isn’t the only task, we want to generalize.

• Apply learned model to unseen data:
  • Underlying data distribution: \((x, y) \sim \mathcal{D}\)
Generalization

Fitting data isn’t the only task, we want to generalize.

• Apply learned model to unseen data:
  • Underlying data distribution: $(x, y) \sim \mathcal{D}$

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

Fitting data isn’t the only task, we want to generalize.

• Apply learned model to unseen data:
  • Underlying data distribution: \((x, y) \sim \mathcal{D}\)
    \[
    \mathbb{E}_\mathcal{D}[\ell(\hat{f}(x), y)]
    \]
• Can study generalization either theoretically or empirically.
Generalization

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

- Apply learned model to unseen data:
  - Underlying data distribution: 
    \[ (x, y) \sim \mathcal{D} \]
    \[ \mathbb{E}_\mathcal{D}[\ell(\hat{f}(x), y)] \]

- Can study generalization either theoretically or empirically.
  - For theory: need assumptions, ie, training instances are iid
Generalization

Fitting data isn’t the only task, we want to generalize.

- Apply learned model to unseen data:
  - Underlying data distribution: \((x, y) \sim D\)
  
  \[ E_D[\ell(\hat{f}(x), y)] \]

- Can study generalization either theoretically or empirically.
  - For theory: need assumptions, ie, training instances are iid
  - Not always the case!
Generalization

Fitting data isn’t the only task, we want to generalize.

• Apply learned model to unseen data:
  • Underlying data distribution: \((x, y) \sim \mathcal{D}\)

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

• Can study generalization either theoretically or empirically.
  • For theory: need assumptions, ie, training instances are iid
  • Not always the case!
    • Sequential data
Break & Quiz
Q2-1: Which of the following is a NOMINAL feature as introduced in the lecture?

1. Cost $\in [0, 100]$
2. Awarded $\in \{True, False\}$
3. Steak $\in \{\text{Rare, Medium Rare, Medium, Medium Well, Well Done}\}$
4. Attitude $\in \{\text{strongly disagree, disagree, neutral, agree, strongly agree}\}$
Q2-1: Which of the following is a NOMINAL feature as introduced in the lecture?

1. Cost $\in [0, 100]$
2. **Awarded** $\in \{\text{True, False}\}$
3. Steak
   $\in \{\text{Rare, Medium Rare, Medium, Medium Well, Well Done}\}$
4. Attitude
   $\in \{\text{strongly disagree, disagree, neutral, agree, strongly agree}\}$
Q2-2: What is the dimension of the following feature space?
The CIFAR-10 dataset contains 60,000 32x32 color images in 10 different classes.
(convert each data to a vector)

1. 10
2. 60,000
3. 3072
4. 1024
Q2-2: What is the dimension of the following feature space?
The CIFAR-10 dataset contains 60,000 32x32 color images in 10 different classes. (convert each data to a vector)

1. 10
2. 60,000
3. 3072
4. 1024
Q2-2: What is the dimension of the following feature space?  
The CIFAR-10 dataset contains 60,000 32x32 color images in 10 different classes. (convert each data to a vector)

1. 10
2. 60,000
3. **3072**
4. 1024

Every color image has 3 channels (RGB) and 32*32 pixels, so the dimension is 3*32*32=3072.
Q2-3: Are these statements true or false?
(A) Instances from time series are independent and identically distributed.
(B) The primary objective of supervised learning is to find a model that achieves the highest accuracy on the training data.

1. True, True
2. True, False
3. False, True
4. False, False
Q2-3: Are these statements true or false?
(A) Instances from time series are independent and identically 
distributed.
(B) The primary objective of supervised learning is to find a model that 
achieves the highest accuracy on the training data.

1. True, True
2. True, False
3. False, True
4. False, False
Q2-3: Are these statements true or false?
(A) Instances from time series are independent and identically distributed.
(B) The primary objective of supervised learning is to find a model that achieves the highest accuracy on the training data.

1. True, True
2. True, False
3. False, True
4. False, False
Outline
Outline

• Review from last time
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning
• Supervised learning concepts
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning

• Supervised learning concepts
  • Features, models, training, other terminology
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning

• Supervised learning concepts
  • Features, models, training, other terminology

• Unsupervised learning concepts
Outline

• Review from last time
  • Supervised, unsupervised, reinforcement learning

• Supervised learning concepts
  • Features, models, training, other terminology

• Unsupervised learning concepts
  • Clustering, anomaly detection, dimensionality reduction
Outline

• **Review from last time**
  • Supervised, unsupervised, reinforcement learning

• **Supervised learning concepts**
  • Features, models, training, other terminology

• **Unsupervised learning concepts**
  • Clustering, anomaly detection, dimensionality reduction

• **Reinforcement learning concepts**
Outline

- **Review from last time**
  - Supervised, unsupervised, reinforcement learning

- **Supervised learning concepts**
  - Features, models, training, other terminology

- **Unsupervised learning concepts**
  - Clustering, anomaly detection, dimensionality reduction

- **Reinforcement learning concepts**
  - Exploration vs. Exploitation, credit-assignment.
Unsupervised Learning: Setup

- Given instances
Unsupervised Learning: Setup

• Given instances \( \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\} \)
Unsupervised Learning: Setup

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

• **Goal**: discover interesting regularities/structures/patterns that characterize the instances. For example:
Unsupervised Learning: Setup

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

• **Goal:** discover interesting regularities/structures/patterns that characterize the instances. For example:
  - Clustering
Unsupervised Learning: Setup

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

• **Goal**: discover interesting regularities/structures/patterns that characterize the instances. For example:
  • Clustering
  • Anomaly detection
Unsupervised Learning: Setup

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

• **Goal**: discover interesting regularities/structures/patterns that characterize the instances. For example:
  • Clustering
  • Anomaly detection
  • Dimensionality reduction
Clustering: Setup

- Given instances \( \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\} \)
Clustering: Setup

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

- **Goal:** find model \( h \) divides the training set into clusters with
Clustering: Setup

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

- **Goal**: find model \( h \) divides the training set into clusters with
  - intra-cluster similarity
Clustering: Setup

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

• **Goal**: find model \( h \) divides the training set into clusters with
  • intra-cluster similarity
  • inter-cluster dissimilarity
Clustering: Setup

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

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

- Clustering *irises*:
Clustering: Setup

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

• **Goal**: find model \( h \) divides the training set into clusters with
  - intra-cluster similarity
  - inter-cluster dissimilarity

• Clustering *irises*:
Anomaly Detection: Setup

- Given instances \( \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\} \)
Anomaly Detection: Setup

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

• Goal: model \( h \) that represents “normal” \( x \)
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
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
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 \( \{ x^{(1)}, x^{(2)}, \ldots, x^{(n)} \} \)
Dimensionality Reduction: Setup

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

- **Goal**: model \( h \) that represents \( x \) with
Dimensionality Reduction: Setup

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

• **Goal:** model \( h \) that represents \( x \) with
  • lower-dim. feature vectors
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
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

• 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
Dimensionality Reduction: Setup

Example: Eigenfaces
Dimensionality Reduction: Setup

Example: Eigenfaces

\[ \text{Example: Eigenfaces} \]

\[ = \alpha_1^{(1)} \times + \alpha_2^{(1)} \times + \ldots + \alpha_{20}^{(1)} \times \]
Dimensionality Reduction: Setup

Example: Eigenfaces

\[ x^{(1)} = \alpha_1^{(1)} \times + \alpha_2^{(1)} \times + \ldots + \alpha_{20}^{(1)} \times \]

\[ x^{(1)} = \langle \alpha_1^{(1)}, \alpha_2^{(1)}, \ldots, \alpha_{20}^{(1)} \rangle \]
Dimensionality Reduction: Setup

Example: Eigenfaces

\[ x^{(1)} = \alpha_1^{(1)} \times \text{face} + \alpha_2^{(1)} \times \text{face} + \ldots + \alpha_{20}^{(1)} \times \text{face} \]

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

\[ = \alpha_1^{(2)} \times \text{face} + \alpha_2^{(2)} \times \text{face} + \ldots + \alpha_{20}^{(2)} \times \text{face} \]
Dimensionality Reduction: Setup

Example: Eigenfaces

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

\[ \sum \alpha_{1}^{(2)} \times \text{face1} + \alpha_{2}^{(2)} \times \text{face2} + \ldots + \alpha_{20}^{(2)} \times \text{face20} \]

\[ x^{(1)} = \langle \alpha_1^{(2)}, \alpha_2^{(2)}, \ldots, \alpha_{20}^{(2)} \rangle \]
Dimensionality Reduction: Setup

Example: Eigenfaces

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

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

What dimension are we using now?
Q3-1: Which generally is NOT an unsupervised learning task?

1. Principal component analysis
2. Fraud detection
3. CIFAR-10 image classification
4. Community detection
Q3-1: Which generally is NOT an unsupervised learning task?

1. Principal component analysis
2. Fraud detection
3. CIFAR-10 image classification
4. Community detection
Q3-1: Which generally is NOT an unsupervised learning task?

1. Principal component analysis
2. Fraud detection
3. CIFAR-10 image classification
4. Community detection

1. Principal component analysis is a problem of dimensionality reduction.
2. You can think fraud detection as an anomaly detection problem.
3. CIFAR-10 image classification is a classification task for labeled image data.
4. Community detection is some clustering problem.
Model Zoo

Lots of models!
Outline

• **Review from last time**
  • Supervised, unsupervised, reinforcement learning

• **Supervised learning concepts**
  • Features, models, training, other terminology

• **Unsupervised learning concepts**
  • Clustering, anomaly detection, dimensionality reduction

• **Reinforcement learning concepts**
  • Exploration vs. Exploitation, credit-assignment.
Reinforcement Learning

- Given: an agent that can take actions and a reward function specifying how good an action is.
Reinforcement Learning

• Given: an agent that can take actions and a reward function specifying how good an action is.
• **Goal:** learn to choose actions that maximize future reward total.
Reinforcement Learning

• Given: an agent that can take actions and a reward function specifying how good an action is.
• **Goal:** learn to choose actions that maximize future reward total.
Reinforcement Learning

- Given: an agent that can take actions and a reward function specifying how good an action is.
- **Goal:** learn to choose actions that maximize future reward total.
Reinforcement Learning
Reinforcement Learning

• Given: an agent that can take actions and a reward function specifying how good an action is.

• **Goal:** learn to choose actions that maximize future reward total.
Reinforcement Learning

• Given: an agent that can take actions and a reward function specifying how good an action is.
• **Goal:** learn to choose actions that maximize future reward total.
Reinforcement Learning

- Given: an agent that can take actions and a reward function specifying how good an action is.
- **Goal:** learn to choose actions that maximize future reward total.

Agent collects data $s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_T, a_T, r_T$. 
**Reinforcement Learning**

- **Given:** an agent that can take actions and a reward function specifying how good an action is.
- **Goal:** learn to choose actions that maximize future reward total.

Agent collects data $s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_T, a_T, r_T$.

Learn policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that maximizes $\sum_{t=0}^{\infty} \gamma^t r_t$. 
Reinforcement Learning Key Problems
Reinforcement Learning Key Problems

1. Problem: actions may have delayed effects.
Reinforcement Learning Key Problems

1. Problem: actions may have delayed effects.
   • Requires credit-assignment
Reinforcement Learning Key Problems

1. Problem: actions may have delayed effects.
   • Requires **credit-assignment**

2. Problem: maximal reward action is unknown
Reinforcement Learning Key Problems

1. Problem: actions may have delayed effects.
   - Requires **credit-assignment**

2. Problem: maximal reward action is unknown
   - Exploration-exploitation trade-off
Reinforcement Learning Key Problems

1. Problem: actions may have delayed effects.
   - Requires **credit-assignment**

2. Problem: maximal reward action is unknown
   - Exploration-exploitation trade-off

Multi-armed Bandit
Reinforcement Learning Key Problems

1. Problem: actions may have delayed effects.
   - Requires **credit-assignment**

2. Problem: maximal reward action is unknown
   - Exploration-exploitation trade-off

“..the problem [exploration-exploitation] was proposed [by British scientist] to be dropped over Germany so that German scientists could also waste their time on it.”

- Peter Whittle

Multi-armed Bandit
Learning Outcomes

• **After today’s lecture:**
  • You will be able to explain the key aspects of a supervised learning problem.
  • Provide examples of unsupervised learning problems and explain why these are not supervised learning problems.
  • Explain key challenges of reinforcement learning problems.
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, and Fred Sala