CS 486/686: Introduction to Artificial Intelligence

Machine Learning

Plan for Today

- Introduction to Machine Learning
  - Components
  - Common Learning Tasks
  - Measuring Success
  - Bias
  - Learning as Search

- Supervised Learning
  - Basic Framework
  - Linear Classifiers

Introduction

Learning is the ability to improve one's behaviour based on experience

- The range of behaviours is expanded
  - The agent can do more
- The accuracy on tasks is improved
  - The agent can do things better
- The speed is improved
  - The agent can do things faster

What is Machine Learning

Definition (T Mitchell):

A computer program is said to **learn** from **experience** E with respect to some class of **tasks** T and **performance measures** P, if its performance at tasks in T, as measured by P, improves with experience E.

Examples

- Handwriting recognition
  - Tasks: Recognize and classify handwritten letters and digits
  - Experience: Database of pre-classified letters and digits
  - Performance measure: Percent of letters/digits correctly classified

- Game playing problem
  - Tasks: Playing the game
  - Experience: Playing practice games against itself (self-play)
  - Performance measure: Percentage of games won against an opponent

Common Learning Tasks

Supervised Classification

- Given a set of pre-classified training examples, classify a new instance

Unsupervised Learning

- Find natural classes for examples

Reinforcement Learning

- Determine what to do based on rewards and punishments

Transfer Learning

- Learning from an expert

Active Learning

- Actively seek to learn

Feedback

Learning tasks can be defined by the feedback the learner receives

Supervised Learning:

- What has to be learned is specified for each example

Unsupervised Learning:

- No classifications are given. Learner has to discover categories and patterns in the data

Reinforcement Learning:

- Feedback occurs after taking a sequence of actions. Credit assignment problem

Representations

- The representation of what we are learning is crucial
  - It determines how the learning algorithm will work

- The richer the representation the more useful it is for subsequent problem solving

- The richer the representation, the more difficult it is to learn

Measuring Performance

- We will always have some sort of **performance measure** so as to judge the learning
- The measure of performance is not on how well the agent does on the training examples, but on how well it performs with new examples
- Example
  - Agent P claims the negative examples it has seen are the only negative examples. All other instances are positive.
  - Agent N claims the positive examples it has seen as the only positive one. All other instances are negative.
  - What will happen?

Supervised Learning

Let H be the set of all possible hypothesis given our chosen representation

Learning is search though H to find a “good” h

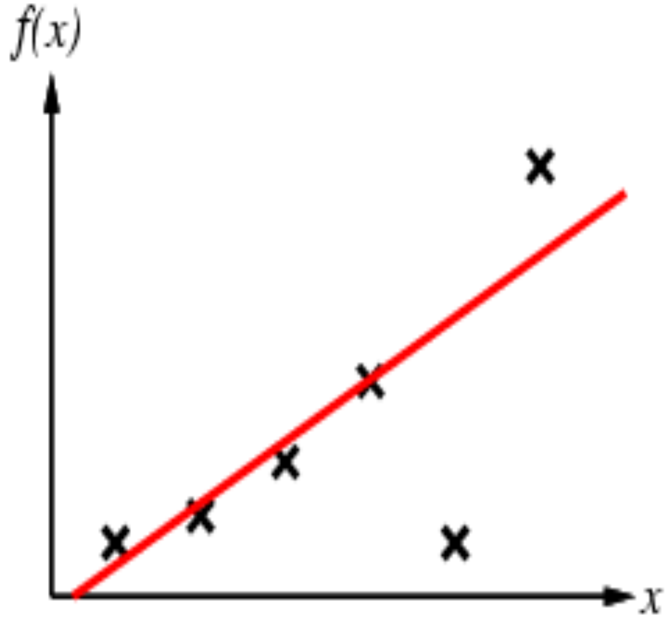
What does “good” mean?

- Usually that it **generalizes** well (i.e. performs well on unseen examples)

Inductive Learning Hypothesis

Any hypothesis found to approximate the target function well ***over a sufficiently large set of training examples*** will also approximate the target function well over any unobserved examples

Inductive Learning

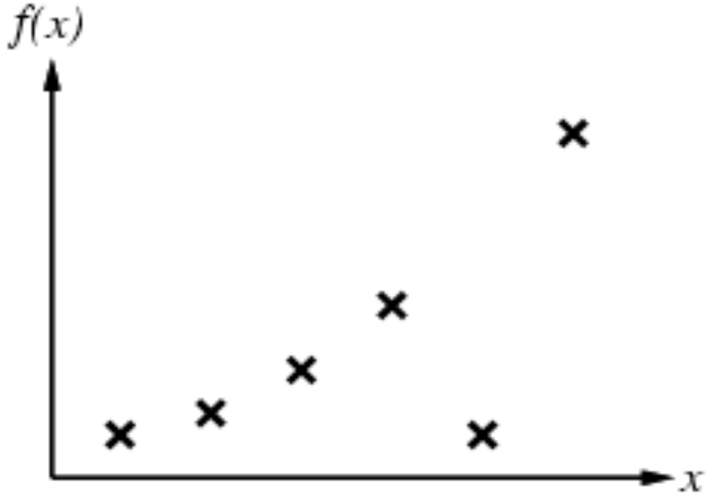


Construct/adjust h to agree with f on training set

h is **consistent** if it agrees with f on all examples

e.g. curve fitting

Inductive Learning

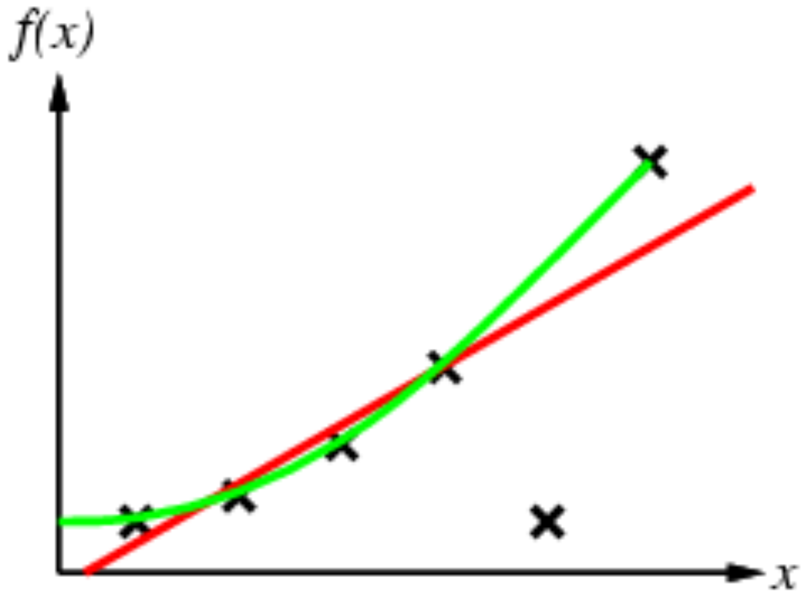


Construct/adjust h to agree with f on training set

h is **consistent** if it agrees with f on all examples

e.g. curve fitting

Inductive Learning

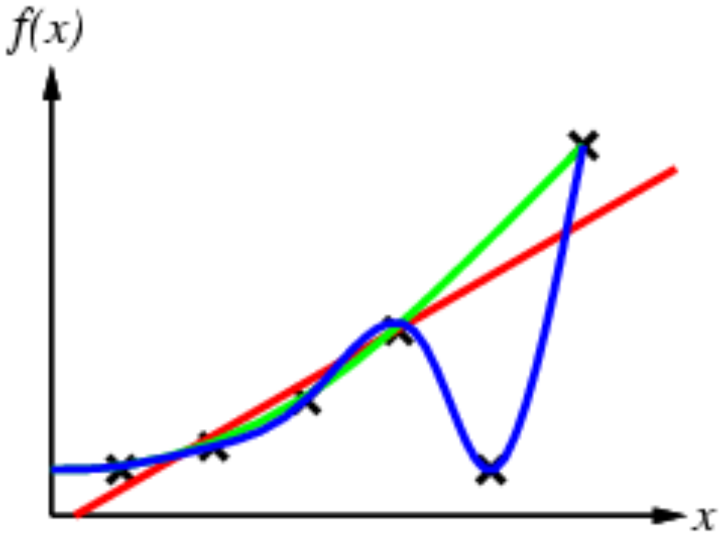


Construct/adjust h to agree with f on training set

h is **consistent** if it agrees with f on all examples

e.g. curve fitting

Inductive Learning

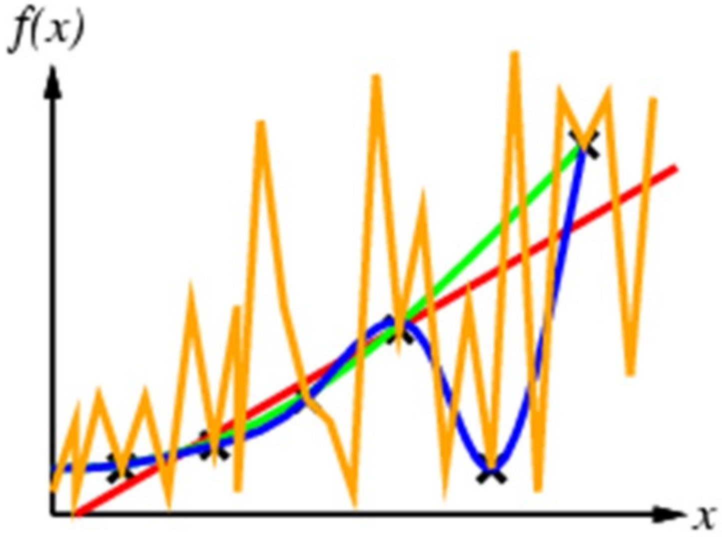


Construct/adjust h to agree with f on training set

h is **consistent** if it agrees with f on all examples

e.g. curve fitting

Inductive Learning



Bias (Ockham's Razor): Prefer the simplest hypothesis consistent with the data

Bias

- A tendency to prefer one hypothesis over another is a **bias**
- Saying a hypothesis is better than another isn't necessarily something that is obtained from the data
- To make any inductive process make predictions on unseen data, an agent must have a bias
- What is a good bias is an empirical question
  - Often prefer simpler hypothesis over complex (Ockham's Razor)

Learning as Search

- Given a representation, data, and a bias, we now have a search problem
- Learning is search though the space of possible representations looking for the representation that best fits the data, given the bias
- Search spaces are usually too large for systematic search (instead use gradient descent, stochastic simulation,….)
- A learning problem is made up of a search space, an evaluation function, and a search method

Some Notes About Data

Data is not perfect:

- The features given are inadequate to predict classification
- There are examples with missing features
- Data is just incorrect (e.g. labeled incorrectly)
- It is incomplete
- …

Overfitting

- Finding patterns in the data where there is no actual pattern

Supervised Learning

Given

- A set of input features $X_1,\ldots,X_n$
- A set of target features f(**X**) or $Y_1,\ldots,Y_k$
- A set of training examples where the values for the input features and target features are given for each example
- A set of test examples, where only the values for the input features are given

Predict the values for the target features for the test examples

- Classification: Yi are discrete
- Regression: Yi are continuous

- **Very Important: keep training and test sets separate!!!**

Supervised Learning

Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Xi

f(X)=Y

Goal: Return a function h that approximates f(x)

h is the **hypothesis**

Evaluating Performance of a Supervised Learning Algorithm

- Suppose Y is a feature and e is an example
  - Y(e) is the true value of feature Y for example e
  - Y*(e) is the predicted value of feature Y for example e

- The error of the prediction is a measure of how close Y*(e) is to Y(e)

- There are many ways of measuring error
  - Absolute error, sum-of-squares error, worst-case error, cost-based error, likelihood, entropy,…

Receiver Operating Curve (ROC)

- Not all errors are equal!
  - Predict a patient has a disease when they do not
  - Predict a patient does not have a disease when they do

Predicted

Actual

	T	F
T	True Positive (TP)	False Negative (FN)
F	False Positive (FP)	True Negative (TN)

Receiver Operating Curve (ROC)

Predicted

Actual

	T	F
T	True Positive (TP)	False Negative (FN)
F	False Positive (FP)	True Negative (TN)

- Recall=Sensitivity = TP/(TP+FN)
- Specificity = TN/(TN+FP)
- Precision = TP/(TP+FP)
- F-measure = 2*Precision*Recall/(Precision + Recall)

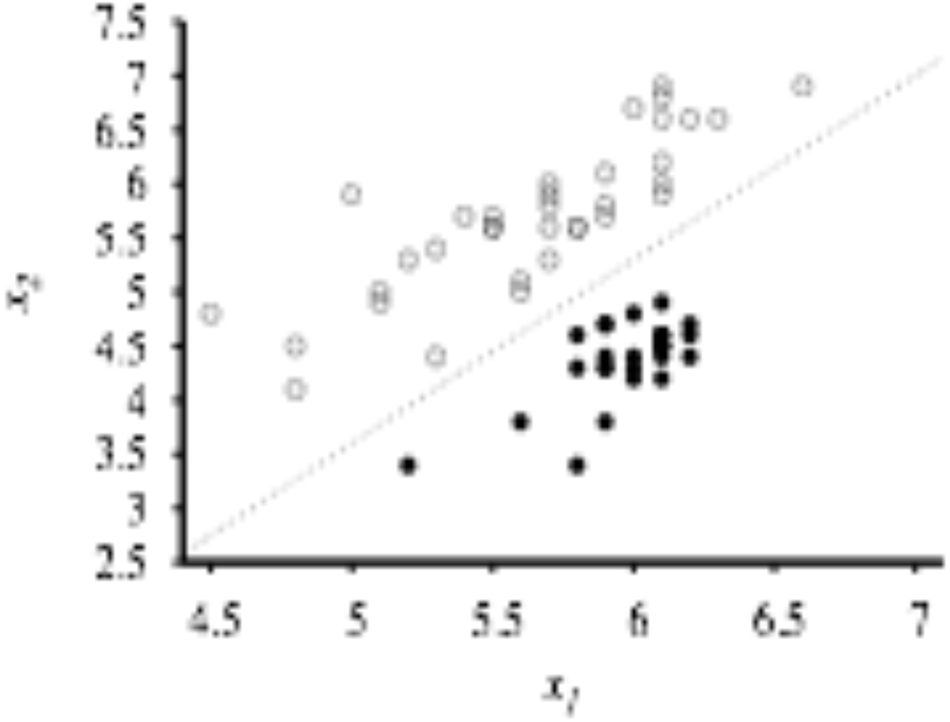
Supervised Learning

Many supervised learning algorithms can be seen as being derived from

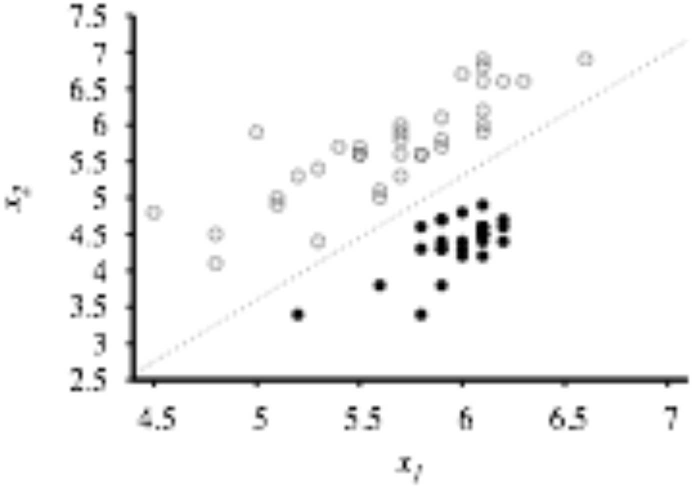
- **Linear Classifiers**
- Decision Trees
- Bayesian Classifiers (later in the semester)

Classification with Linear Thresholds

Imagine you have data of the form ($\mathbf{x}$,f($\mathbf{x}$)) where x in R^n and f($\mathbf{x}$) in {0,1}



Linear Threshold Classifiers



$$\mathbf{w} \cdot \mathbf{x} = w_0 + w_1 x_1 + w_2 x_2$$

$$h_{\mathbf{w}}(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Linear Threshold Classifiers

Learning Problem: Find the weights **w** such that $h_{\mathbf{w}}$ is a good classifier

$$h_{\mathbf{w}}(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{w} \cdot \mathbf{x} = w_0 + w_1 x_1 + w_2 x_2$$

Learning Problem: Find the weights **w** to minimize the loss function.

$$\text{Loss}(h_{\mathbf{w}}) = L_2(y, h_{\mathbf{w}}(\mathbf{x})) = \sum_{j=1}^{N} (y_j - h_{\mathbf{w}}(\mathbf{x}_j))^2$$

Gradient Descent

w ← any point in parameter space

Loop until convergence do

    For each w_i in **w** do

$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} \mathrm{Loss}(\mathbf{w})$$

α is the step size or learning rate. It can be a fixed constant or can decrease over time as the learning progresses

Update Rule (Perceptron Update Rule)

When updating weights

$$w_i \leftarrow w_i + \alpha(y - h_w(x))x\_i$$

Intuition:

Assessing Performance

A learning algorithm is **good** if it produces a hypothesis that does a good job of predicting classifications of unseen examples

There are theoretical guarantees (learning theory)

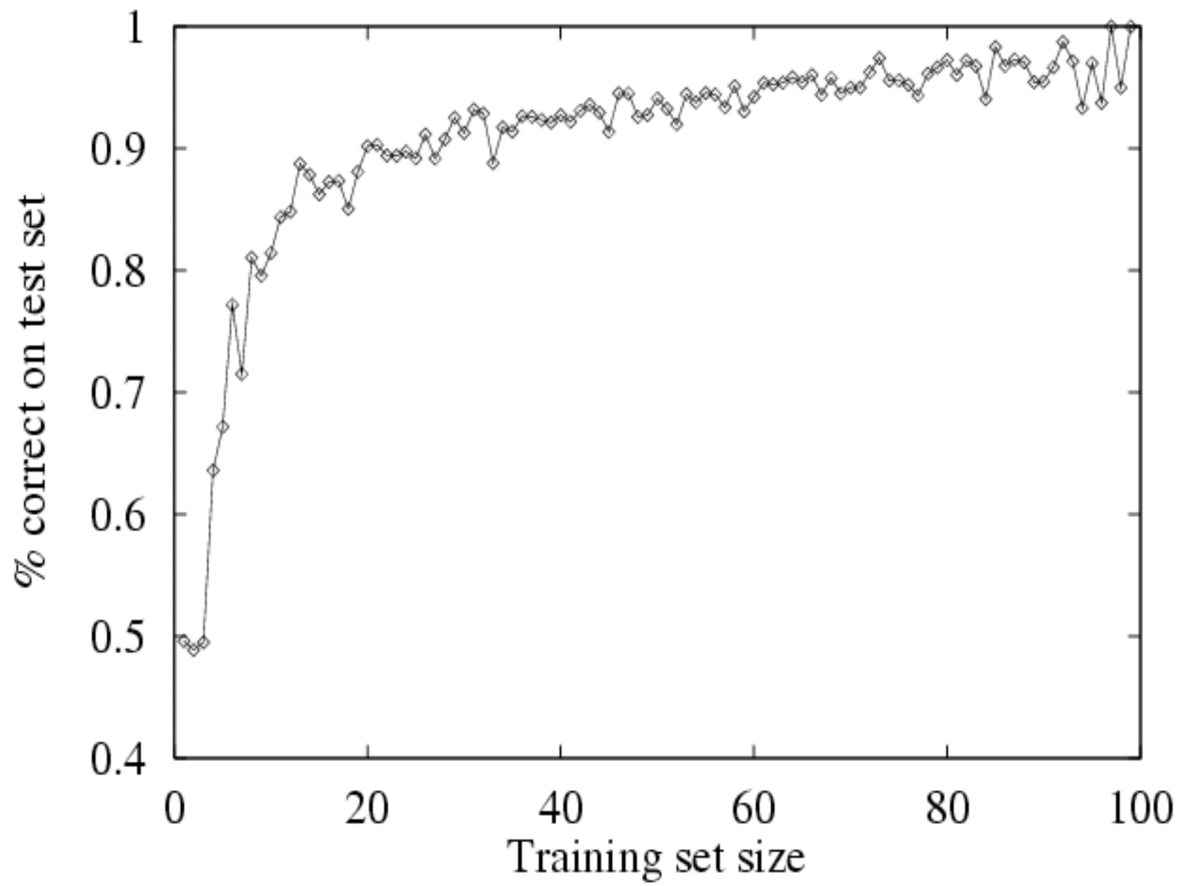
Can also test this

Assessing Performance

Test set

- Collect a large set of examples
- Divide them into 2 disjoint sets (training set and test set)
- Apply learning algorithm to training set to get h
- Measure percentage of examples in the test set that are correctly classified by h

Learning Curve



As the training set grows, accuracy increases

No Peeking at the Test Set

A learning algorithm should not be allowed to see the test set data before the hypothesis is tested on it

No Peeking!!

Every time you want to compare performance of a hypothesis on a test set ***you should use a new test set***!

What You Should Know

- Basic categories of Machine Learning
- General Supervised Learning Framework
  - Linear Threshold Classifiers
- Assessing Performance