Artificial Intelligence Programming

Machine Learning

Cindi Thompson

Department of Computer Science

University of San Francisco

Introduction

- *Machine learning* is one of the richest and most active areas of AI.
- Recent explosion in available data, coupled with problems that are difficult to exhaustively model.
- *Data mining* is a close cousin of machine learning.

Machine Learning

- Previously, we've assumed that background knowledge was given to us by experts.
  - Focused on how to use that knowledge
- Today, we'll talk about acquiring that knowledge from observation.

What does it mean for an agent to learn?

Agent Learning

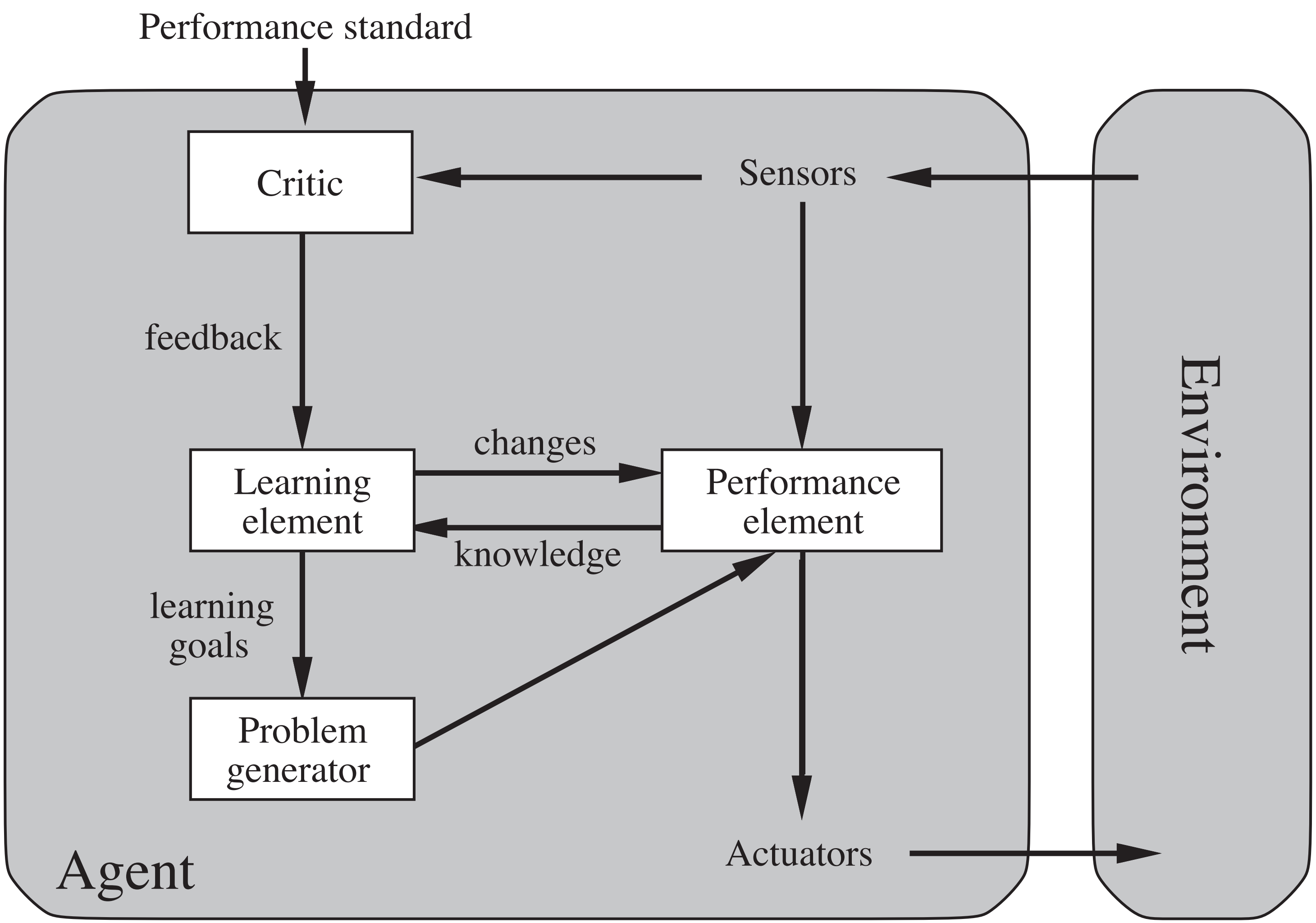
What does it mean for an agent to learn?

- Agent acquires new knowledge
- Agent changes its behavior
- Agent improves its performance measure on a given task

Agent Learning

- A learning agent has a *performance element* and a learning element.
  - The performance element is what an agent uses to decide what to do.
  - This is what we've studied up to now.
- The learning element is what allows the agent to modify the performance element.
  - This might mean adding or changing rules or facts, modifying a heuristic, changing a successor function
  - In order to modify its behavior, an agent needs information telling it how well it is performing.
  - This information is called feedback

Learning Agent



What is Learning?

- A program is said to learn from experiences E with respect to a set of tasks T and a performance measure P if its performance on T, as measured by P, improves with experience E.
- This means that, for a well-formulated learning problem, we need:
  - A set of tasks, T, the agent must perform
  - A way to measure its performance, P
  - A way to quantify the experience, E, the agent receives

What is Learning?

- We can also think about the learning task in isolation.
- Learning is the process of discovering patterns in data.
- These patterns should allow us to better understand the data and make predictions about it.

Representing Data

- We will assume that our data consists of a set of *instances*
- Each instance consists of one or more *attributes*.
    - These might correspond to an agent's percepts.
- Each attribute may take one one or more *values*
- Attributes may be nominal or numeric.
    - Nominal is sometimes subdivided into enumerated, discrete, categorical
    - Boolean is a special case of categorical.

Tennis Example

- Consider the problem of an agent deciding whether we should play tennis on a given day.
- There are four observable percepts:
  - Outlook (sunny, rainy, overcast)
  - Temperature (hot, mild, cool)
  - Humidity (high, normal)
  - Wind (strong, weak)
- We don't have a model, but we do have some data about past decisions.
- Can we induce a general rule for when to play tennis?

Tennis Example

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

ZeroR

- In assignment 1, you implemented a simple learning algorithm called ZeroR.
- It discovered a broad pattern in the dataset:
  - Which classification was most common?
- ZeroR tells us the most likely classification, assuming the other attributes are not relevant.
  - We'll call this sort of assumption a *bias*.

Other Learning Examples

- Speech recognition
  - Task: successfully recognize spoken words
  - Performance measure: fraction of words correctly recognized
  - Experience: A database of labeled, spoken words
- Learning to drive a car
  - Task: Drive on a public road using vision sensors
  - Performance: average distance driven without error
  - Experience: sequence of images and reactions from a human driver.
- Learning to play backgammon
  - Task: play backgammon
  - Performance measure: number of games won against humans of the appropriate caliber.
  - Experience: Playing games against itself.

Discussion

- Notice that not all performance measures are the same.
  - In some cases, we want to minimize all errors. In other cases, some sorts of errors can be more easily tolerated than others.
- Also, not all experience is the same.
  - Are examples labeled?
  - Does a learning agent immediately receive a reward after selecting an action?
  - How is experiental data represented? Symbolic? Continuous?
- Also: What is the final product?
  - Do we simply need an agent that performs correctly?
  - Or is it important that we understand *why* the agent performs correctly?

Types of Learning Problems

- One way to think about learning is to focus on the learning task.
  - Classification: What class do instances belong to?
  - Association: What is the relationship between attributes?
  - Numeric: Predict a numeric value from a set of attributes.
  - Clustering: Find groupings for instances.

Classification

- In this case, instances belong to a set of (two or more) classes.
- Given an instance, correctly predict which class it belongs to.
  - For a particular day, should we play tennis?
  - Is a given patient expected to have a recurrence of cancer?
  - What species of iris is a particular plant?
- ZeroR is a classifier.
- This is probably the most common learning task.

Association

- Association is the process of discovering relationships between attributes.
- These relationships may be logical:
  - Outlook==overcast $\rightarrow$ Play-tennis == yes
- or statistical:
  - P(overcast, play-tennis) = 0.8
- Classification deals with a specific dependent variable, whereas association looks at any set of variables, but we can build a classifier from association rules.

Function approximation

- Given a set of inputs, find a real-valued function that approximates the correct output.
- Rather than mapping inputs to a class, we map them to a real number.
- Usually, the approximation is structurally simpler than the data.
  - Linear regression is an example of a function approximation.

Clustering

- Clustering is the process of grouping data instances together into groups based on similarity.
- Unlike classification, we don't start with a predefined set of classes.
- We try to find natural groupings of instances.
- Very popular with documents, demographic data, some vision systems

Types of learning problems

- Another way to characterize learning problems is by the sorts of data and feedback our agent has access to.
  - Batch vs incremental
  - Supervised vs unsupervised
  - Active vs passive
  - Online vs offline

Batch vs Incremental

- We can think about problems or algorithms being batch or incremental.
- A *batch* learning algorithm is one in which all of the data is available at once to the agent.
- An incremental learning algorithm is one that can continue to incorporate new data over time as it becomes available.
- In principle, batch learning is more effective, but it may not fit the characteristics of all problems.

Supervised vs Unsupervised

- A *supervised* learning algorithm/problem is one in which the learner has access to labeled training data.
- *Unsupervised* algorithms/problems are ones in which no labeled training data is available.
  - The recommender systems used by Amazon and Netflix are examples of this.
- Supervised learning is easier, but it assumes that you have access to labeled training data.

Active vs Passive

- In *active learning*, the learning agent is able to construct examples and find out thier classification.
  - For example, a spam classifier that could create emails and ask a teacher whether it was spam or not.
- In *passive learning*, the learning agent must work with the examples that are presented.
  - ZeroR is a passive learner.
- Active learning is more effective, as the agent can choose examples that lets it better “hone” its hypothesis, but may not fit with a particular problem.

Online vs Offline

- An *offline* learing algorithm is able to separate learning from execution.
  - Learning and performance are separate
  - Offline learning is easier, computational complexity is less of a factor.
- An *online* learning algorithm allows an agent to mix learning and execution.
  - Agent takes actions, receives feedback, and updates its performance component.
  - Online learning more realistic, fast algorithms a requirement.

The main difference between incremental and online is that the latter gets feedback about how it did, while the former just gets new labeled examples.

Some simple examples

- Let's look at some simple examples of learning algorithms.
- Consider (once again) our playTennis example.
- Suppose we have the following data:
  - sunny, overcast, high, weak : yes
  - sunny, overcast, low, strong: yes
  - rainy, overcast, normal, weak: no
- We need to select a hypothesis that explains all of these examples
  - H1: sunny : yes
  - H2: sunny and overcast: yes
  - H3: $\neg$ rainy, overcast, normal, weak : yes
- Which do we pick?

Representing a hypothesis

- Before we can answer, we need to decide how our hypothesis will be represented.
  - Probability distributions?
  - All possible logic expressions?
  - Only conjunctions?
  - Negation?
- Simpler hypotheses can be learned more quickly
- May not fit data as well.

Representing a hypothesis

How will our hypothesis be represented?

- Complex hypotheses may fit the data more accurately
- But what if we get too complex?

ZeroR

- In assignment 1, you implemented a simple algorithm known as ZeroR (for Zero-rules).
- Zero-R ignores all attributes and just chooses the most common classification.
  - For this data, it will predict 'yes'.
  - Hypothesis is only a single class.
- Its *bias* is to assume that the other attributes are not relevant.
- This is simple, but probably not correct most of the time.

Find-S

- Suppose we agree that hypotheses consist of a single attribute value or “don’t care” for each attribute.
  - sunny and don’t care; sunny and overcast are possible
  - sunny or rainy is not.
- This is called a *representational bias*.
- Stronger representational biases let us learn more quickly.
- Find the most specific hypothesis that explains our data.
  - $sunny \wedge overcast \rightarrow yes$ is more specific.

Hypothesis spaces

- We can arrange all potential hypotheses from specific-to-general in a lattice.
- Our learning problem is now to search this space of hypotheses to find the best hypothesis that is consistent with out data.
- Learning can then be thought of as a search problem.
- The way in which those hypotheses (successors) are considered is called the *learning bias*.
- Every algorithm has a representational bias and a learning bias.
  - Understanding them can help you know how your learning algorithm will generalize.

Separating

- In some cases, we can arrange our data in an n-dimensional space.
- We can then phrase our problem as finding a line or hyperplane that best separates our data into classes.
- How do we quantify “best”?
- How do we choose between equally good hyperplanes?
- Goal: select the hyperplane that best predicts new data.

Measuring Performance

- How do we evaluate the performance of a classifying learning algorithm?
- Traditional measures are precision, recall, and accuracy
- Precision is the fraction of examples classified as belonging to class x that are really of that class.
    - How well does our hypothesis avoid *false positives*?
- Recall is the fraction of true members of class x that are actually captured by our hypothesis.
    - How well does our hypothesis avoid *false negatives*?

Accuracy: for *all* classes, what fraction were labeled correctly?

Precision vs recall

- Often, there is a tradeoff of precision vs recall.
  - In our playTennis example, what if we say we always play tennis?
  - This will have a high recall, but a low precision.
  - What if we say we'll never play tennis except where we played before?
  - High precision, low recall.
- Try to make a compromise that best suits your application.
- What is a case where a false positive would be worse than a false negative?
- What is a case where a false negative would be better than a false positive?

Evaluation

- Typically, in evaluating the performance of a learning algorithm, we'll be interested in the following sorts of questions:
  - Complexity: What is the training time (time needed to construct a hypothesis)? Classification time (time needed to classify a new instance)?
  - Does performance improve as the number of training examples increases?
  - How do precision and recall trade off as the number of training examples changes?
  - How does performance change as the problem gets easier/harder?
- So what does 'performance' mean?

Issues with data

- In many real world problems, data does not come in nice, neat categories.
- A number of potential problems can arise:
  - Data cleaning
  - Noise
  - Missing values

Data cleaning

- This is the process of transforming your data for use by your algorithm.
- This might involve:
  - Normalization
  - Discretization
  - Attribute (feature) selection
- This can be very involved, but can greatly improve performance.

Noise

- *Noise* is a general term that refers to incorrect data.
  - For example, a tennis instance was supposed to be classified as ’yes’, but was labeled ’no’ instead.
- This could be the result of inaccurate sensors, data corruption, human error, or a nondeterministic environment.
- Some learning algorithms are very sensitive to noise, but others tolerate it quite well.
  - Statistical algorithms handle noise better than logical algorithms.

Missing data

- A related problem to noise is missing data.
- For some instances, an attribute was not measured, or the value was lost.
- Some algorithms can handle this, but usually some sort of pre-processing or cleaning is needed.
  - Replace with default value
  - Replace with most common value
  - Replace with randomly selected value.
- Question: What does it mean that this data is missing? Is it simply an error, or does it indicate something else? (for example, a specimen that was badly damaged)

CS662

- In this class, we'll look at the following learning algorithms:
  - Decision trees (logical classifiers)
  - Naive Bayes (probabilistic classifiers)
  - Rule learning (can employ logic and probability)
  - Reinforcement learning (online learning)
  - Neural networks

Summary

- What is learning?
- Types of learning problems
- Characteristics
- Data handling issues