Introduction to Machine Learning

Connectionist and Statistical Language Processing

Frank Keller

keller@coli.uni-sb.de

Computerlinguistik
Universität des Saarlandes

Overview

- definition of learning
- sample data set
- terminology: concepts, instances, attributes
- learning rules
- learning decision trees
- types of learning
- evaluating learning performance
- learning bias

Definition of Learning

From Mitchell (1997: 2):

> 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.

From Witten and Frank (2000: 6):

> things learn when they change their behavior in a way that makes them perform better in the future.

In practice this means: we have sets of examples from which we want to extract regularities.

A Sample Data Set

Fictional data set that describes the weather conditions for playing some unspecified game.

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes

outlook	temp.	humidity	windy	play
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Terminology

- *Instance:* single example in a data set. Example: each of the rows in the table on the preceding slide.
- *Attribute:* an aspect of an instance. Example: outlook, temperature, humidity, windy. Also called *feature*. Attributes can take categorical or numeric values.
- *Value:* category that an attribute can take. Example: sunny, overcast, rainy for the attribute outlook. The attribute temperature could also take numeric values.
- *Concept:* the thing to be learned. Example: a classification of the instances into play and no play.

Learning Rules

Example for a set of rules learned from the example data set:

if outlook = sunny and humidity = high	then play = no
if outlook = rainy and windy = true	then play = no
if outlook = overcast	then play = yes
if humidity = normal	then play = yes
if none of the above	then play = yes

This is called a *decision list*, and is use as follows: use the first rule first, if it doesn't apply, use the second one, etc.

These are *classification rules* that assign an output class (play or not) to each instance.

Learning Rules

Here is a different set of rules learned from the data set:

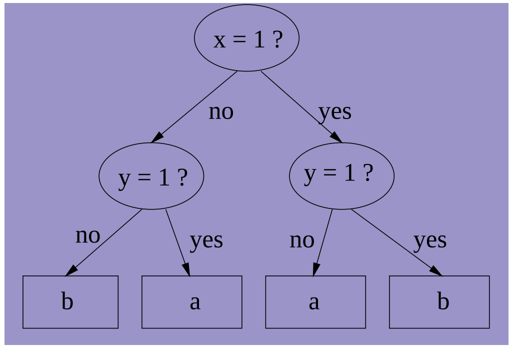
if temperature = cool	then humidity = normal
if humidity = normal and windy = false	then play = yes
if outlook = sunny and play = no	then humidity = high
if windy = false and play = no	then outlook = sunny and humidity = high

These are *association rules* that describe associations between different attribute values.

Learning Decision Trees

Example: XOR (familiar from connectionist networks).

x	y	class
0	0	b
0	1	a
1	0	a
1	1	b



Nodes represent decisions on attributes, leaves represent classifications.

Learning Decision Trees

Any decision tree can be turned into a set of rules.

Traverse the tree depth first and create a conjunction for each node that you hit.

if x = 0 and y = 0	then class = b
if x = 0 and y = 1	then class = a
if x = 1 and y = 0	then class = a
if x = 1 and y = 1	then class = b

Learning Decision Trees

However, this methods creates rule set that can be highly redundant. A better rule set would be:

if x = 0 and y = 1	then class = a
if x = 1 and y = 0	then class = a
Otherwise	class = b

Inverse problem: it is not straightforward to turn a rule set into a decision tree (redundancy: *replicated subtree problem*).

Types of Learning

Machine learning is not only about classification. The following main classes of problems exist:

- *Classification learning:* learn to put instances into pre-defined classes
- *Association learning:* learn relationships between the attributes
- *Clustering:* discover classes of instances that belong together
- *Numeric prediction:* learn to predict a numeric quantity instead of a class

Comparison with Connectionist Nets

Connectionist nets are machine learning engines that can be used for these four tasks. Examples include:

- *Classification learning:* competitive network: selects one unit in the output layer (target class)
- *Association learning:* pattern associator: recalls input patterns based on similarity
- *Clustering:* self-organizing map: reduces dimensions in the input space based on similarity
- *Numeric prediction:* perceptron: outputs a real-valued function in the output layer

Evaluating Learning Performance

What does it mean for a model to successfully learn a concept?

- *descriptive:* captures the training data;
- *predictive:* generalizes to unseen data;
- *explanatory:* provides a plausible description of the concept to be learned.

Descriptive Evaluation

Measures of model fit commonly used in computational linguistics (originally proposed for information retrieval):

Precision: how many data points the model gets right:

$$\text{Precision} = \frac{|\text{data points modeled correctly}|}{|\text{data points modeled}|}$$

Recall: how many data points the model accounts for:

$$\text{Recall} = \frac{|\text{data points modeled correctly}|}{|\text{total data points}|}$$

Descriptive Evaluation

Measure of model fit commonly used in connectionist nets: *Mean Squared Error*:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(H_i - H_i')^2$$

H_i: quantity observed in the data set
H_i': quantity predicted by model
n: number of instances in the data set

Traditionally, precision/recall has been used for classification tasks, while MSE has been used for numeric prediction tasks.

Predictive Evaluation

Is the model able to generalize? Can it deal with unseen data, or does it overfit the data? Test on *held-out data*:

- split data to be modeled in *training* and *test* set;
- train the model (determine its parameters) on training set;
- apply model to *training set*; compute model fit
- apply model to *test set*; compute model fit;
- difference between compare model fit on training and test data measured the model's ability to *generalize*.

Explanatory Evaluation

The concept of explanatory adequacy is elusive. Does the model provide a plausible description of the concept to be learned?

- Classification: does it base its classification on plausible classification rules?
- Association: does it discover plausible relationships in the data?
- Clustering: Does it come up with plausible clusters?

The meaning of ‘plausible’ to be defined by a human expert.

Learning Bias

To generalize successfully, a machine learning system uses a *learning bias* to guide it through the space of possible concepts.

Language bias: the language in which the result is expressed determines which concepts can be learned.

Example: a learner that can learn disjunctive rules will get different results than one that doesn’t use disjunction.

An important factor is *domain knowledge*, e.g., the knowledge that certain combinations of attributes can never occur.

Learning Bias

Search bias: the way the space of possible concepts is searched determines the outcome of learning.

Example: greedy search: try to find the best rule at each stage and add it to the rule set; beam search: pursue a number of alternative rule sets in parallel.

Two common search biases are *general-to-specific* (start with a general concept description and refine) and *specific-to-general* (start with a specific example and generalize).

Learning Bias

Overfitting-avoidance bias: avoid learning a concept that overfits, i.e., just enumerates the training data: this will give very bad results on test data, as it lacks the ability to generalized to unseen instances.

Example: consider simple concepts first, then proceed to more complex one, e.g., avoid a complex rule set with one rule for each training instance.

Approaches Dealt with in this Course

- Decision tree learning
- Bayesian learning
- Memory-based learning
- Clustering
- Applications of machine learning

References

Mitchell, Tom. M. 1997. *Machine Learning.* New York: McGraw-Hill.

Witten, Ian H., and Eibe Frank. 2000. *Data Mining: Practical Machine Learing Tools and Techniques with Java Implementations.* San Diego, CA: Morgan Kaufmann.