Програмиране в изкуствения интелект—Зимен семестър 1999/2000.
Преподавател: Иван А. Держански.

Лекция 11: Машинно самообучение

индуктивно самообучение (учене): извличане на знание от примери. Предполага се, че знанието представлява определение на някакво понятие, а примерите (зададени във вид на съвкупности от характеристики на предмети, ситуации и т. н.) са **положителни** и **отрицателни**. Целта на ученето е да се формулира определение на понятието, което да **покрива** (почти) всички положителни примери и (почти) никои от отрицателните. Полученото определение служи за **класифициране** на кандидати, които не са сред примерите.

(Тази постановка е частен случай на по-общата, в която примерите са разделени на повече от две категории и съответно всеки кандидат получава една от три, четири и т. н. възможни класификации. Характеристиките също могат да бъдат зададени не като отговори на да/не-въпроси, а като атрибути с две или повече възможни стойности за всеки.)

Ученето винаги става с някаква **склонност (bias)**, защото една и съща група примери (стига те да не изчерпват всички възможни кандидати) може да задава много голям брой различни понятия (броят на различните двоични функции с n аргумента е 2^{2^n}).

1 Учене на класификационни дървета

Класификационно дърво: графично представяне на процеса на класифицирането на кандидат. Всеки лист съдържа една от възможните класификации, а всеки от останалите възли проверява някой атрибут, като с всяка възможна стойност на атрибута се свързва по едно поддърво—наследник на този възел. Ако класификационното дърво е двоично, то се нарича **дърво на решенията (decision tree)**.

Алгоритъм: Дадени са примери $E_1, \ldots, E_m$, като $E_i = \langle x_{i,1}, \ldots, x_{i,n}, y_i \rangle$ ($x_{i,j}$ е стойността на j-тия атрибут за i-тия пример, а y_i е класификацията на примера). Нека $k = |\{y_1, \ldots, y_m\}|$.

ако $k = 1$,

то дървото е лист, съдържащ стойността $y = y_1 = \ldots y_m$;

иначе ако $0 < n$,

то от съвкупността атрибути се избира x_j; дървото има корен x_j и k поддървета, всяко от които е дърво, получено чрез същия алгоритъм от примерите $\{E_i | y_i = y, 1 \leq i \leq m\}$ за някоя стойност $y \in \{y_1, \ldots, y_m\}$;

иначе атрибутите не стигат, за да се класифицират примерите, защото

- данните са недостатъчни,
- данните са «шумни» (съдържат експериментални грешки),
- областта е недетерминистична.

В такъв случай може да се прибегне до вероятностно решение (дървото е лист, съдържащ някаква средна стойност на $\{y_1, \ldots, y_m\}$ или процедура, изработваща недетерминистично една от тези стойности).

Стремежът е да се построи най-компактното от многото (в общия случай) възможни дървета, които определят дадените примери.

Оценяване на дървото: зададените примери се разделят (по случаен начин) на **учебни** и **тестови**; дървото се построява, като се използват само учебните, и след това неговата класификация на тестовите се сравнява с дадената. Това се повтаря многократно с различни разбивания на примерите на учебни и тестови, след което се избира най-доброто дърво.

2 Учене на логически изрази

Задачата представлява търсене в пространство от хипотези $\{H_1, \ldots, H_n\}$ при предположение, че $H_1 \vee \cdots \vee H_n$. За всяка хипотеза един пример е **лъжлив положителен**, ако е отрицателен, но тя го покрива, и **лъжлив отрицателен**, ако е положителен, но тя не го покрива.

2.1 Стратегия «текуща най-добра хипотеза» (current-best-hypothesis search)

Основава се на последователно разглеждане на примерите и подлагане на текущата най-добра хипотеза на недетерминистичните операции **обобщение** при откриване на лъжлив отрицателен пример и **уточнение** при откриване на лъжлив положителен. При всяка корекция се проверява съвместимостта с всички предходни примери. Търсенето е с обратен ход, поради което е скъпо.

2.2 Стратегия «най-малко обвързване» (least-commitment search)

Използува две **множества от гранични хипотези**: G на най-общите хипотези, съвместими с всички разгледани примери, и S—на най-специфичните. Нито една хипотеза от G не може да се обобщи, без да покрие някой отрицателен пример; нито една хипотеза от S не може да се уточни, без да престане да покрива някой положителен пример.

Примерите се разглеждат последователно, като се съпоставят с всяка хипотеза H от двете гранични множества. Действие предизвикват само лъжливите положителни и отрицателни примери, както следва:

	примерът е лъжлив положителен	примерът е лъжлив отрицателен
$H \in G$	H се изважда от G и се замества с всички възможни уточнения	H се изважда от G
$H \in S$	H се изважда от S	H се изважда от S и се замества с всички възможни обобщения

Обратен ход не е нужен, защото разглеждането на всеки пример има траен ефект. Възможни са три изхода:

- G и S не се срещат: допустима е повече от една хипотеза.
- G и S се срещат в една хипотеза, която е резултат от ученето.
- G и S се разминават: данните са противоречиви.