هوش مصنوعی پیشرفته

فصل ۱۸

یادگیری از مثال‌ها

Learning from Examples

کاظم فولادی
دانشکده مهندسی برق و کامپیوتر
دانشگاه تهران

http://courses.fouladi.ir/aai

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

◆

مقدمه

عامل‌های یادگیرنده

یادگیری

LEARNING AGENTS

یادگیری، مکانیزم‌های تصمیم‌گیری عامل را به‌گونه‌ای تغییر می‌دهد که کارایی آن بهبود یابد.

عامل‌های یادگیرنده

یادگیری

LEARNING AGENTS

یادگیری به یک عامل اجازه می‌دهد که در **محیط‌های ابتدائاً ناشناخته** عمل کند و سپس از آنچه از آنچه دانایی اولیه‌اش به‌تنهایی ممکن بود اجازه بدهد، **شایسته‌تر** شود.

ساخت عامل‌های هوش مصنوعی	
ایجاد مؤلفه‌ی یادگیرنده و آموزش عامل	برنامه‌نویسی از صفر

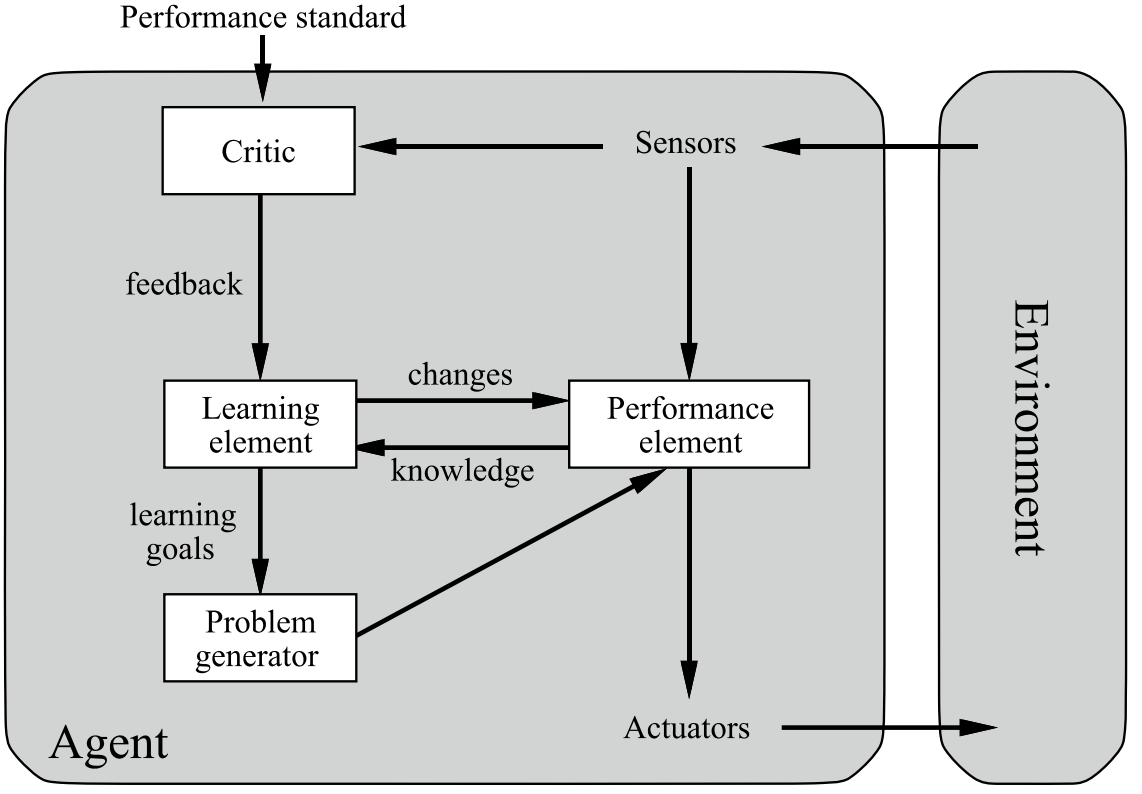
پیشنهاد آلن تورینگ (۱۹۵۰)

دو پرسش برای طراحی عامل یادگیرنده:

(۱) **عنصر انجام‌دهنده چه باشد؟** (یکی از ساختارهای چهارگانه برنامه عامل)
(۲) **چگونه یادگیری انجام شود؟** (روش‌های گوناگون و مؤلفه‌های مختلف ساختار عامل)

عامل‌های یادگیرنده

LEARNING AGENTS



عامل‌های یادگیرنده

اجزای داخلی

معیار کارآیی استاندارد: Performance standard

ثابت است.
باید به طور کامل خارج از عامل باشد
(عامل نباید بتواند آن را تغییر دهد).

منتقد
Critic

به عنصر یادگیرنده فیدبک می‌دهد:

با توجه به یک معیار استاندارد کارآیی ثابت به عنصر یادگیرنده می‌گوید که عامل چه‌قدر خوب عمل کرده است.

مسئول انتخاب کنش بیرونی:

می‌تواند هر یک از ساختارهای چهارگانه‌ی برنامه‌ی عامل را داشته باشد.

عنصر انجام‌دهنده
Performance element

عنصر یادگیرنده
Learning element

مسئول بهبود بخشیدن به کارآیی:

تغییرات عنصر انجام‌دهنده را برای بهتر عمل کردن در آینده مشخص می‌کند.

مولد مسئله
Problem generator

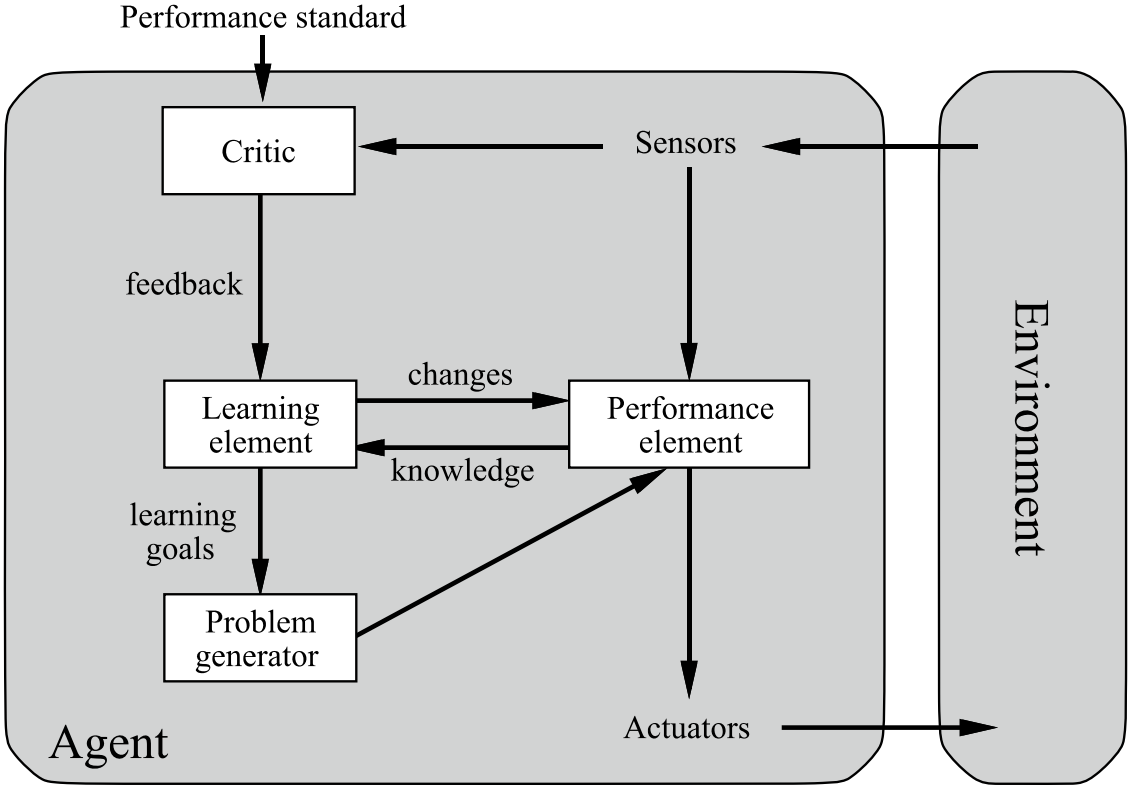
مسئول پیشنهاد کنش‌های منجر به تجربه‌های تازه و آموزنده:

پیشنهاد کنش‌های کاوشگرانه بر اساس اهداف یادگیری

یادگیری در عامل هوشمند: فرآیند تغییر هر جزء عامل به این منظور که آن جزء تطابق بیشتری با اطلاعات فیدبک داشته باشد و از این طریق کارآیی کل عامل بهبود یابد.

عامل‌های یادگیرنده

LEARNING AGENTS



هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۱

صورت‌های یادگیری

عنصر یادگیری

LEARNING ELEMENT

هر مؤلفه‌ی یک عامل می‌تواند به‌وسیله‌ی یادگیری از داده‌ها بهبود پیدا کند.

مؤلفه‌های مؤثر در طراحی عنصر یادگیری			
عنصر انجام‌دهنده *Performance Element*	**مؤلفه‌ی کارکردی** *Functional Component*	**بازنمایی** *Representation*	**فیدبک** *Feedback*
چه نوع عنصر انجام‌دهنده‌ای استفاده می‌شود؟	کدام مؤلفه‌ی کارکردی باید یادگرفته شود؟	مؤلفه‌ی کارکردی چگونه بازنمایی می‌شود؟	چه نوع فیدبکی در دست است؟
نوع معماری برنامه‌ی عامل	مؤلفه‌ای که باید بهبود یابد	بازنمایی داده‌ها و مؤلفه	فیدبک موجود برای یادگیری

+

دانایی پیشین
Prior Knowledge

دانایی پیشین
که عامل در اختیار دارد

عنصر یادگیری

مثال

LEARNING ELEMENT

مؤلفه‌های مؤثر در طراحی عنصر یادگیری			
عنصر انجام‌دهنده *Performance Element*	**مؤلفه‌ی کارکردی** *Functional Component*	**بازنمایی** *Representation*	**فیدبک** *Feedback*
عامل جستجوی تخاصمی *Adversarial Search Agent*	تابع ارزیابی *Evaluation Function*	تابع خطی وزن‌دار *Weighted Linear Function*	برد/باخت *Win/Loss*
عامل منطقی *Logical Agent*	مدل گذر *Transition Model*	اصول موضوع حالت مابعد *Successor-State Axioms*	برآمد *Outcome*
عامل مبتنی بر سودمندی *Utility-Based Agent*	مدل گذر *Transition Model*	شبکه‌ی بیزی پویا *Dynamic Bayes Network*	برآمد *Outcome*
عامل واکنشی ساده *Simple Reflex Agent*	تابع ادراک ـ کنش *Percept-Action Function*	شبکه‌ی عصبی *Neural Network*	کنش درست *Correct Action*

صورت‌های یادگیری

FORMS OF LEARNING

صورت‌های یادگیری					
یادگیری استقرائی *Inductive Learning*					**یادگیری استنباطی** *Deductive Learning*
یادگیری جزء به کل					**یادگیری کل به جزء**
یادگیری یک تابع یا قاعده‌ی عمومی (درست/نادرست) از روی جفت‌های خاص ورودی ـ خروجی					**یادگیری تحلیلی** *Analytical Learning*
یادگیری بانظارت *Supervised*	**یادگیری بی‌نظارت** *Unsupervised*	**یادگیری تقویتی** *Reinforcement*	**یادگیری نیمه‌نظارتی** *Semisupervised*		حرکت از یک قاعده‌ی عمومی شناخته‌شده به قاعده‌ی جدیدی که منطقاً استلزام می‌شود. (مفید است، زیرا امکان پردازش کارآمدتر را فراهم می‌کند.)
یادگیری نگاشت ورودی به خروجی با دیدن مثال‌های برچسب‌دار	یادگیری الگوهای درون ورودی بدون وجود فیدبک صریح (مثل clustering)	یادگیری از روی یک سری تقویت‌ها (پاداش‌ها و جریمه‌ها)	یادگیری با وجود تعداد کمی مثال برچسب‌دار و مجموعه‌ی بزرگی از داده‌های بی‌برچسب		
تقسیم‌بندی بر اساس **نوع فیدبک** موجود برای یادگیری					
سه نوع اصلی یادگیری					

یادگیری استقرائی

آنچه ساینس (science) نامیده می‌شود

INDUCTIVE LEARNING (A.K.A. SCIENCE)

یادگیری استقرائی *Inductive Learning*			
یادگیری جزء به کل			
یادگیری یک تابع یا قاعده‌ی عمومی (درست/نادرست) از روی جفت‌های خاص ورودی ـ خروجی			
یادگیری بانظارت *Supervised*	یادگیری بی‌نظارت *Unsupervised*	یادگیری تقویتی *Reinforcement*	یادگیری نیمه‌نظارتی *Semisupervised*

* در مورد ساینس، عامل کل نژاد بشری در کل زمان‌هاست.

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۲

یادگیری با نظارت

یادگیری بانظارت

SUPERVISED LEARNING

یادگیری بانظارت
Supervised Learning

یک **مجموعه‌ی آموزشی** از N جفت ورودی–خروجی نمونه داده شده است:

$$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$$

که در آن هر y_j به‌وسیله‌ی یک تابع مجهول f تولید شده است:

$$y = f(x)$$

یک تابع h را کشف کنید که تابع واقعی f را تقریب بزند.

یادگیری بانظارت

مفاهیم کلیدی

SUPERVISED LEARNING

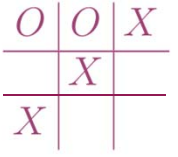
یادگیری بانظارت *Supervised Learning*	
مجموعه‌ی آموزشی *Training Set*	مجموعه‌ی جفت‌های ورودی–خروجی معلوم برای آموزش یادگیرنده
مجموعه‌ی آزمایشی *Test Set*	مجموعه‌ی جفت‌های ورودی–خروجی معلوم برای آزمایش یادگیرنده
هدف *Target*	تابع f که باید یاد گرفته شود
فرضیه *Hypothesis*	تابع مجهول h که باید تقریب مناسبی برای f باشد
فضای فرضیه *Hypothesis Space* $\mathcal{H}$	مجموعه‌ی همه‌ی توابع منتخب h برای تقریب f

یادگیری بانظارت

مثال: یادگیری ارزش حالت‌ها در بازی دوز

SUPERVISED LEARNING

f is the target function

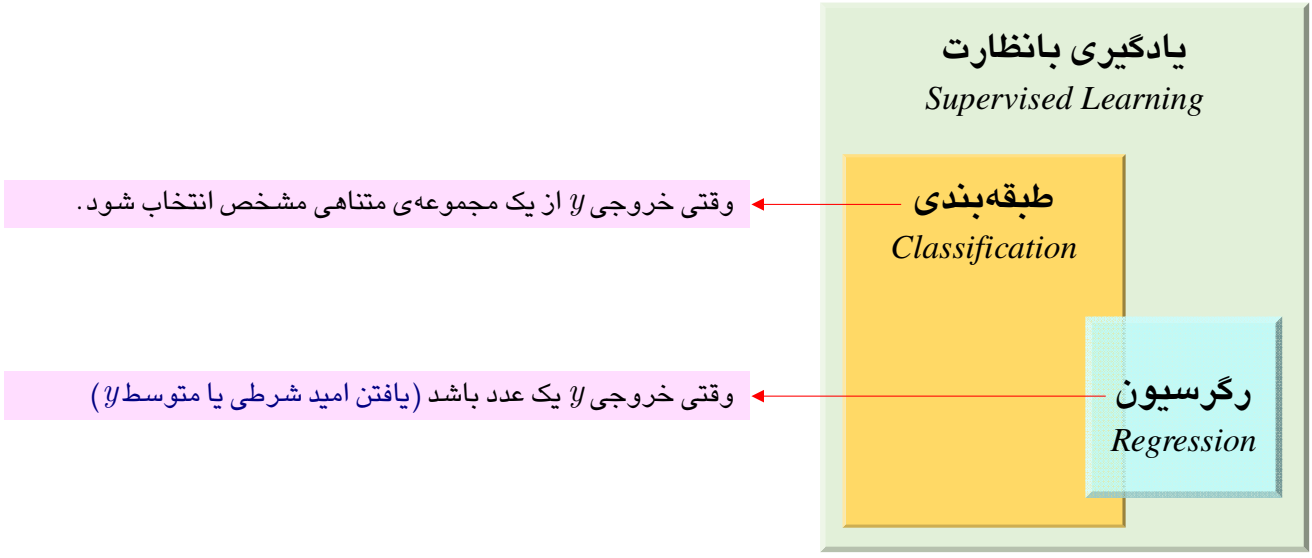
An example is a pair x, $f(x)$, e.g.,  , $+1$

Problem: find a(n) hypothesis h
such that $h \approx f$
given a training set of examples

یادگیری بانظارت

طبقه‌بندی و رگرسیون

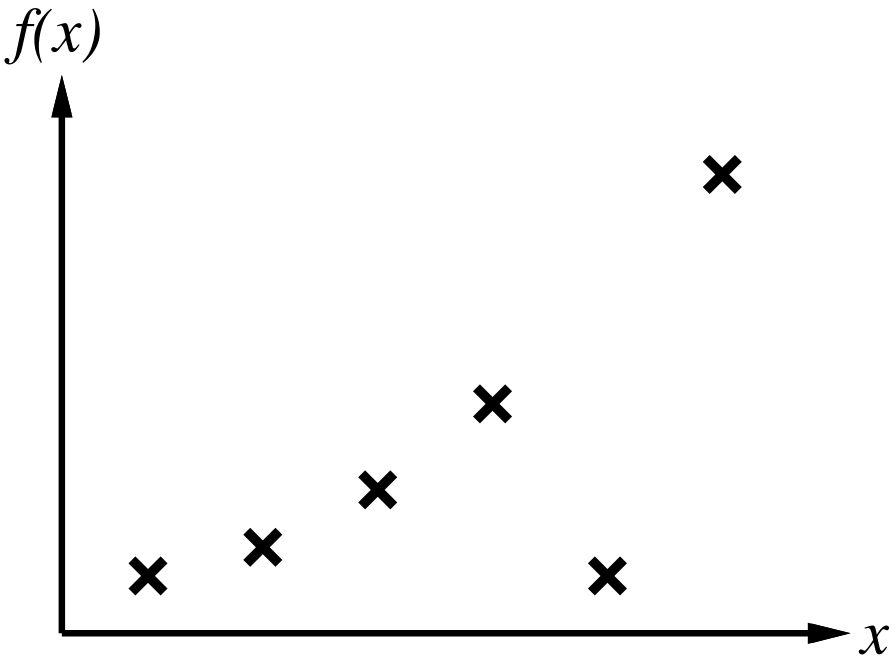
SUPERVISED LEARNING



روش یادگیری بانظارت

مثال: برازش منحنی (۱ از ۵)

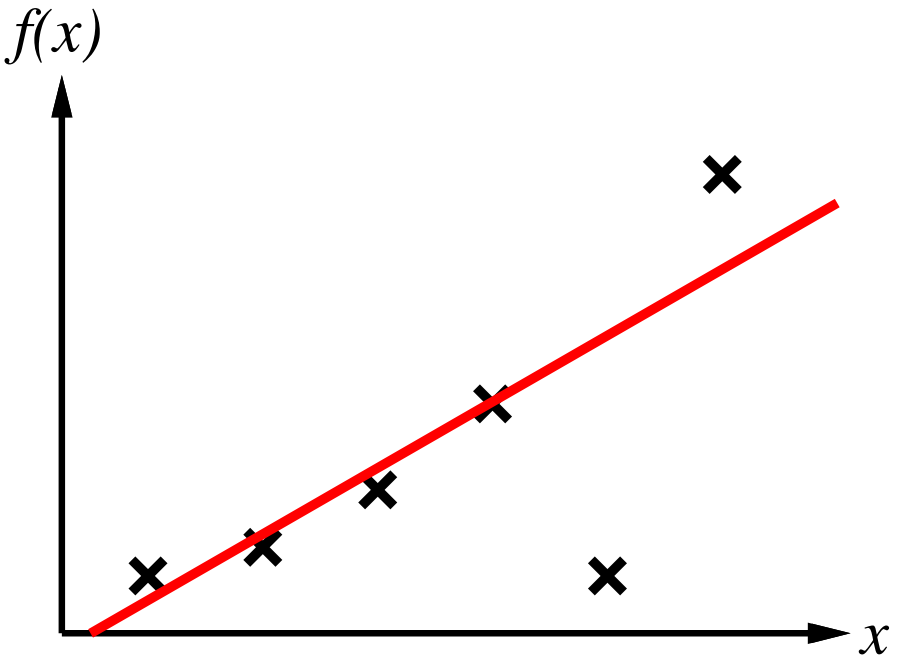
SUPERVISED LEARNING METHOD



روش یادگیری بانظارت

مثال: برازش منحنی (۲ از ۵)

SUPERVISED LEARNING METHOD

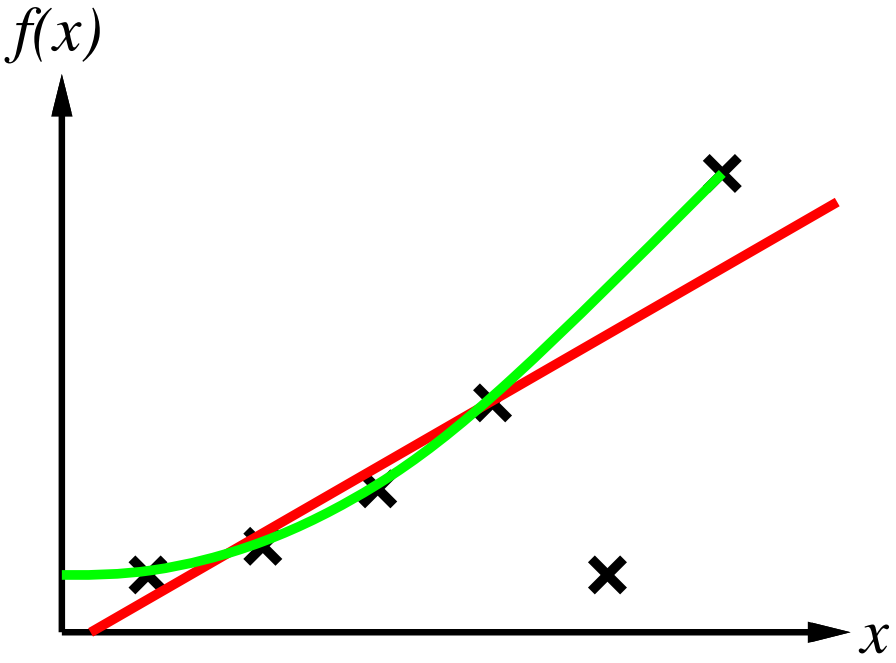


فرضیه بسیار ساده (چندجمله‌ای خطی)، ناسازگار، تعمیم‌پذیری خوب

روش یادگیری بانظارت

مثال: برازش منحنی (۳ از ۵)

SUPERVISED LEARNING METHOD

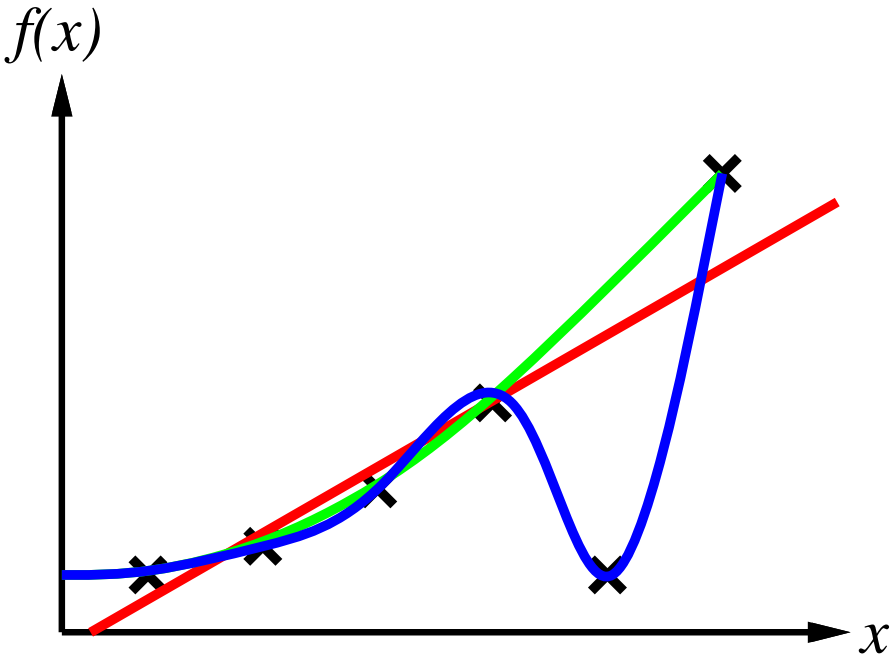


فرضیه ساده (چندجمله‌ای درجه‌دوم)، ناسازگار، تعمیم‌پذیری خوب

روش یادگیری بانظارت

مثال: برازش منحنی (۴ از ۵)

SUPERVISED LEARNING METHOD

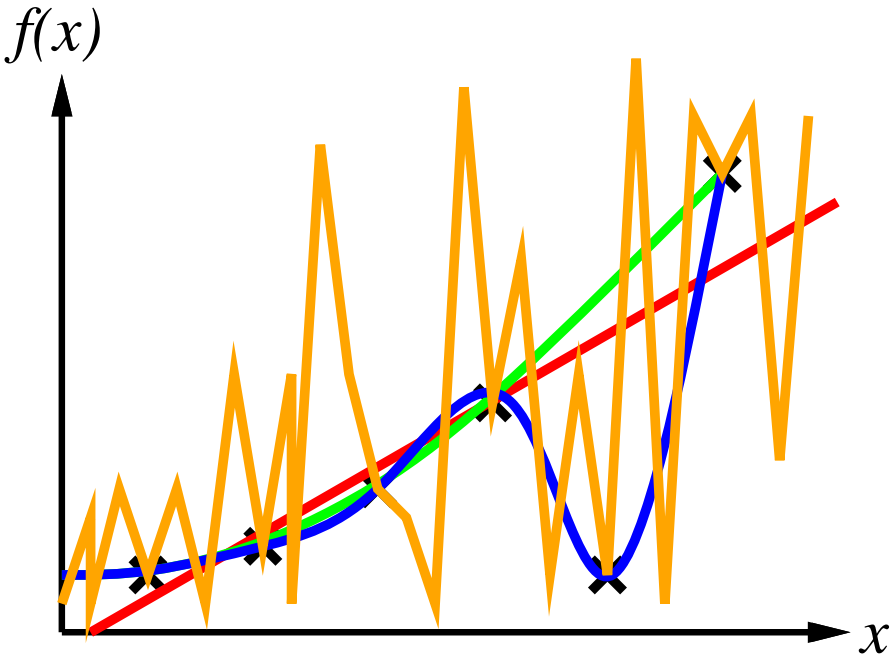


فرضیه پیچیده (چندجمله‌ای درجه بالا)، سازگار، تعمیم‌پذیری متوسط

روش یادگیری بانظارت

مثال: برازش منحنی (۵ از ۵)

SUPERVISED LEARNING METHOD



فرضیه بسیار پیچیده (چندجمله‌ای با درجه‌ی بسیار بالا)، سازگار، تعمیم‌پذیری پایین

یادگیری بانظارت

سازگاری و تعمیم

سازگاری *Consistency*	تعمیم *Generalization*
یک فرضیه سازگار است اگر بر روی همه‌ی نمونه‌های آموزشی درست باشد.	یک فرضیه تعمیم‌پذیر است اگر مقادیر خروجی نمونه‌های جدید را به‌درستی پیش‌بینی کند.

بده‌بستان میان **سازگاری ـ تعمیم‌پذیری:**
فرضیه‌های پیچیده با سازگاری کامل و فرضیه‌های ساده‌تر با تعمیم‌پذیری بالاتر

بده‌بستان میان **رسایی ـ پیچیدگی:**
رسایی یک فضای فرضیه و **پیچیدگی** یافتن یک فرضیه‌ی خوب در آن فضا

تیغه‌ی اوخامی

OCKHAM'S RAZOR

یک مسئله‌ی بنیادی در یادگیری استقرائی:

چگونه بین چند فرضیه‌ی سازگار یکی را انتخاب کنیم؟

ترجیح با **ساده‌ترین فرضیه‌ی سازگار با داده‌ها** است.

تیغه‌ی اوخامی
Ockham's Razor

بهترین مدل برای هر پدیده، ساده‌ترین مدل توصیف‌کننده‌ی آن است.

«به نام ویلیام اوخامی فیلسوف انگلیسی قرن 14 میلادی»

یادگیری بانظارت

محتمل‌ترین فرضیه برای داده‌های موجود

یادگیری بانظارت، با انتخاب فرضیه‌ی h^* انجام می‌شود:

محتمل‌ترین فرضیه با داشتن داده‌ها

$$h^* = \underset{h \in \mathcal{H}}{\operatorname{argmax}} P(h|data)$$

عبارت معادل از طریق قاعده‌ی بیز

$$h^* = \underset{h \in \mathcal{H}}{\operatorname{argmax}} P(data|h)\, P(h)$$

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۳

یادگیری درخت‌های تصمیم

بازنمایی‌های مبتنی بر خصیصه

مثال: انتظار برای میز در یک رستوران

ATTRIBUTE-BASED REPRESENTATIONS

مثال

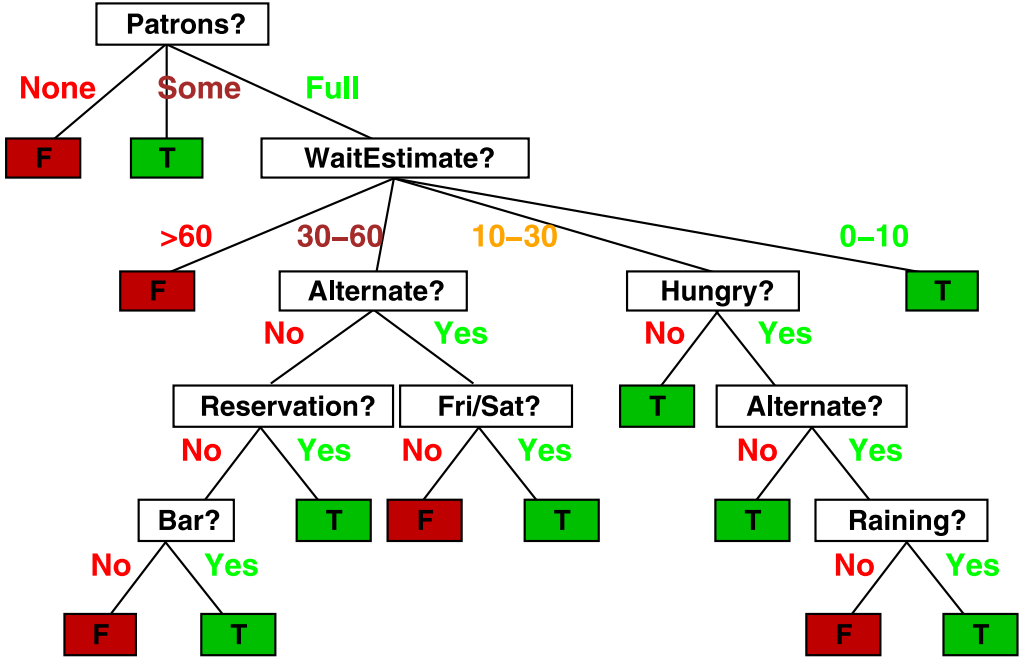
مثال‌هایی از موقعیت‌هایی که در آنها برای یک میز در رستوران صبر می‌کنیم یا خیر:

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*T*	*F*	*F*	*T*	*Some*	*$$$*	*F*	*T*	*French*	*0–10*	*T*
X_2	*T*	*F*	*F*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*30–60*	*F*
X_3	*F*	*T*	*F*	*F*	*Some*	*$*	*F*	*F*	*Burger*	*0–10*	*T*
X_4	*T*	*F*	*T*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*10–30*	*T*
X_5	*T*	*F*	*T*	*F*	*Full*	*$$$*	*F*	*T*	*French*	*>60*	*F*
X_6	*F*	*T*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Italian*	*0–10*	*T*
X_7	*F*	*T*	*F*	*F*	*None*	*$*	*T*	*F*	*Burger*	*0–10*	*F*
X_8	*F*	*F*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Thai*	*0–10*	*T*
X_9	*F*	*T*	*T*	*F*	*Full*	*$*	*T*	*F*	*Burger*	*>60*	*F*
X_{10}	*T*	*T*	*T*	*T*	*Full*	*$$$*	*F*	*T*	*Italian*	*10–30*	*F*
X_{11}	*F*	*F*	*F*	*F*	*None*	*$*	*F*	*F*	*Thai*	*0–10*	*F*
X_{12}	*T*	*T*	*T*	*T*	*Full*	*$*	*F*	*F*	*Burger*	*30–60*	*T*

درخت‌های تصمیم

مثال: انتظار برای میز در یک رستوران

DECISION TREES



درخت تصمیم: یک بازنمایی ممکن برای فرضیه‌ها

خصیصه‌های بی‌ربط در تصمیم‌گیری (type و price) در این درخت استفاده نشده است.

درخت‌های تصمیم

رسایی

EXPRESSIVENESS

درخت‌های تصمیم می‌توانند هر تابعی از خصیصه‌های ورودی را بیان کنند.

مثلاً: برای توابع بولی: هر سطر جدول درستی ← یک مسیر از ریشه تا برگ در درخت تصمیم

برای هر مجموعه‌ی آموزشی، یک درخت تصمیم سازگار وجود دارد:
با یک مسیر از ریشه تا یک برگ برای هر نمونه
(مگر اینکه f برحسب x غیرقطعی باشد.)
اما احتمال دارد که تعمیم‌پذیری آن برای مثال‌های جدید پایین باشد.

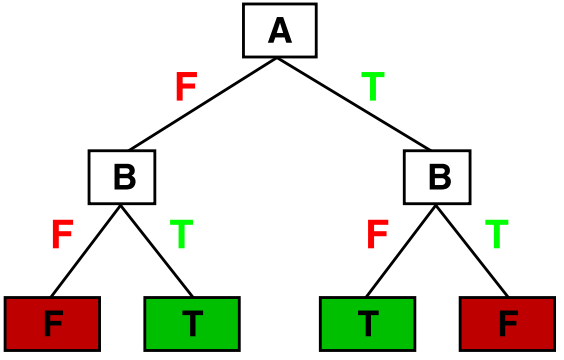
یافتن درخت‌های تصمیم متراکم‌تر ترجیح داده می‌شود.

درخت‌های تصمیم

مثال: تابع یای انحصاری

EXPRESSIVENESS

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



مثلاً: برای تابع بولی xor: هر سطر جدول درستی ← یک مسیر از ریشه تا برگ در درخت تصمیم

فضاهای فرضیه

HYPOTHESIS SPACES

How many distinct decision trees with n Boolean attributes??

= number of Boolean functions
= number of distinct truth tables with 2^n rows = 2^{2^n}

E.g., with 6 Boolean attributes, there are 18,446,744,073,709,551,616 trees

How many purely conjunctive hypotheses (e.g., $Hungry \wedge \neg Rain$)??

Each attribute can be in (positive), in (negative), or out
$\Rightarrow$ 3^n distinct conjunctive hypotheses

More expressive hypothesis space
- increases chance that target function can be expressed
- increases number of hypotheses consistent w/ training set
$\Rightarrow$ may get worse predictions

درخت تصمیم

الگوریتم یادگیری

DECISION TREE LEARNING

هدف: یافتن یک درخت تصمیم کوچک سازگار با مثال‌های آموزشی

ایده‌ی کلی: «مهم‌ترین» خصیصه را به‌عنوان ریشه‌ی (زیر)درخت انتخاب کنید (به صورت بازگشتی)

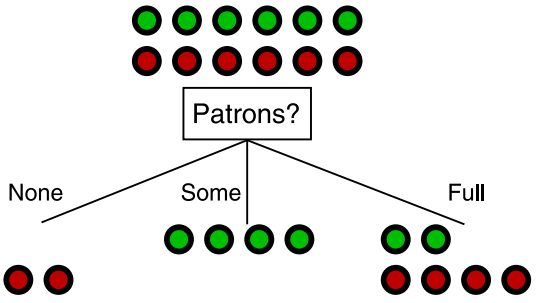
```
function DTL(examples, attributes, default) returns a decision tree
    if examples is empty then return default
    else if all examples have the same classification then return the classification
    else if attributes is empty then return MODE(examples)
    else
        best ← CHOOSE-ATTRIBUTE(attributes, examples)
        tree ← a new decision tree with root test best
        for each value v_i of best do
            examples_i ← {elements of examples with best = v_i}
            subtree ← DTL(examples_i, attributes − best, MODE(examples))
            add a branch to tree with label v_i and subtree subtree
        return tree
```

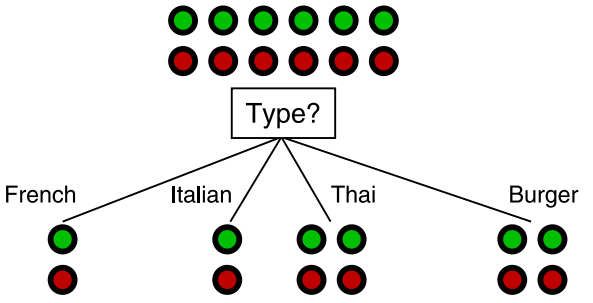
درخت تصمیم

انتخاب یک خصیصه

CHOOSING AN ATTRIBUTE

ایده: یک خصیصه‌ی خوب، مثال‌ها را به مجموعه‌هایی که (در حالت ایده‌آل) «همه مثبت» یا «همه منفی» هستند، می‌شکند.





Patrons? گزینه‌ی بهتری است:
در مورد طبقه‌بندی اطلاعات می‌دهد.

اطلاعات

آنتروپی

INFORMATION

اطلاعات *Information*	اطلاعات به پرسش‌ها پاسخ می‌دهد

هر چه در ابتدا من درمورد پاسخ ناآگاه‌تر باشم، اطلاعات بیشتری در پاسخ وجود دارد.

مقیاس اطلاعات *Information Scale*	۱ بیت = پاسخ به پرسش دودویی با احتمال پیشین $\langle 0.5,0.5 \rangle$

اطلاعات موجود در یک پاسخ وقتی احتمال پیشین برآمدها $\langle P_1, \ldots, P_n \rangle$ است:

$$H(\langle P_1, \ldots, P_n \rangle) = \sum_{i=1}^{n} - P_i \log_2 P_i$$

(موسوم به آنتروپی پیشین)

درخت تصمیم

استفاده از اطلاعات برای انتخاب خصیصه

USING INFORMATION

فرض می‌کنیم p مثال مثبت و n مثال منفی در ریشه‌ی درخت تصمیم داریم

$\Downarrow$

تعداد بیت لازم برای طبقه‌بندی یک مثال جدید:

$$H(\langle p/(p+n), n/(p+n)\rangle)$$

E.g., for 12 restaurant examples, $p = n = 6$ so we need 1 bit

یک خصیصه، مجموعه مثال‌های E را به زیرمجموعه‌های E_i تقسیم می‌کند
(که انتظار می‌رود تعداد بیت کمتری برای کامل کردن طبقه‌بندی نیاز داشته باشند.)

Let E_i have p_i positive and n_i negative examples

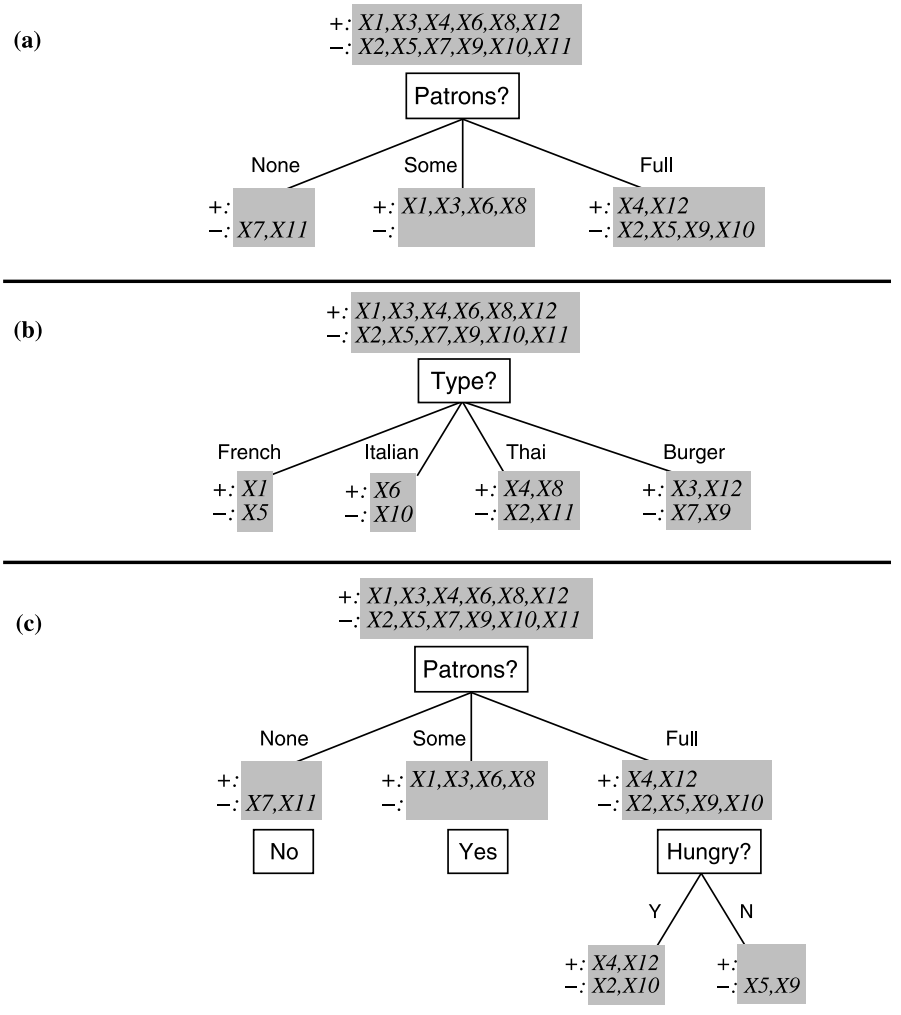
$\Rightarrow$ $H(\langle p_i/(p_i+n_i), n_i/(p_i+n_i)\rangle)$ bits needed to classify a new example

$\Rightarrow$ **expected** number of bits per example over all branches is

$$\sum_i \frac{p_i + n_i}{p + n} H(\langle p_i/(p_i + n_i), n_i/(p_i + n_i)\rangle)$$

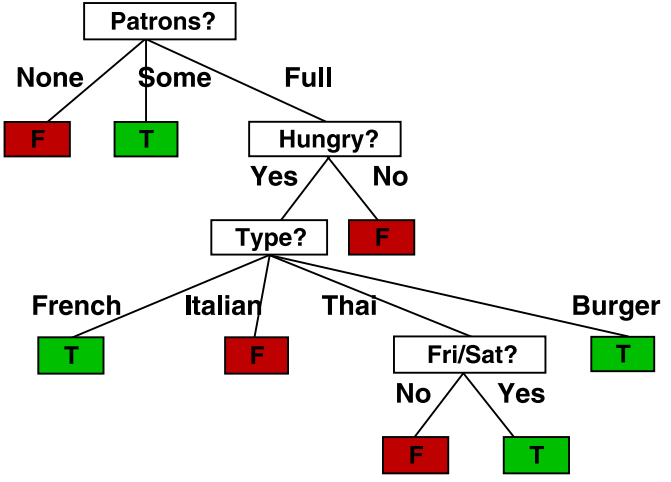
For $Patrons?$, this is 0.459 bits, for $Type$ this is (still) 1 bit

خصیصه‌ای انتخاب می‌شود که اطلاعات لازم باقیمانده را می‌نیمم کند.


(a)
+: X1,X3,X4,X6,X8,X12
−: X2,X5,X7,X9,X10,X11
Patrons?
None
Some
Full
+:
−: X7,X11
+: X1,X3,X6,X8
−:
+: X4,X12
−: X2,X5,X9,X10
(b)
+: X1,X3,X4,X6,X8,X12
−: X2,X5,X7,X9,X10,X11
Type?
French
Italian
Thai
Burger
+: X1
−: X5
+: X6
−: X10
+: X4,X8
−: X2,X11
+: X3,X12
−: X7,X9
(c)
+: X1,X3,X4,X6,X8,X12
−: X2,X5,X7,X9,X10,X11
Patrons?
None
Some
Full
+:
−: X7,X11
+: X1,X3,X6,X8
−:
+: X4,X12
−: X2,X5,X9,X10
No
Yes
Hungry?
Y
N
+: X4,X12
−: X2,X10
+:
−: X5,X9

درخت تصمیم

مثال



درخت تصمیم یاد گرفته شده از روی ۱۲ مثال مسئله‌ی رستوران

ارزیابی کارآیی

PERFORMANCE MEASUREMENT

چگونه می‌توانیم بدانیم که $h \approx f$؟
(مسئله‌ی استقرای دیوید هیوم)

۱) استفاده از قضیه‌های نظریه‌ی یادگیری محاسباتی/آماری

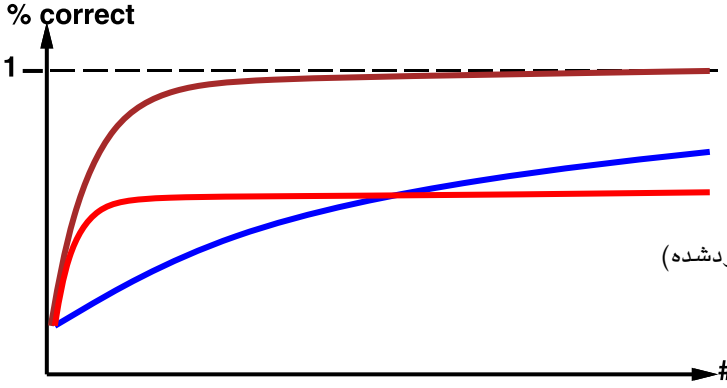
۲) آزمایش h بر روی یک مجموعه‌ی آزمایشی جدید از مثال‌ها
(با استفاده از توزیع مشابهی مانند مجموعه‌ی آموزشی روی فضای مثال‌ها)

ارزیابی کارآیی

منحنی یادگیری

PERFORMANCE MEASUREMENT

منحنی یادگیری *Learning Curve*	درصد پاسخ‌های درست روی مجموعه‌ی آزمایشی به‌صورت تابعی از اندازه‌ی مجموعه‌ی آموزشی



تحقق‌پذیر:
فضای فرضیه می‌تواند تابع هدف را بیان کند.

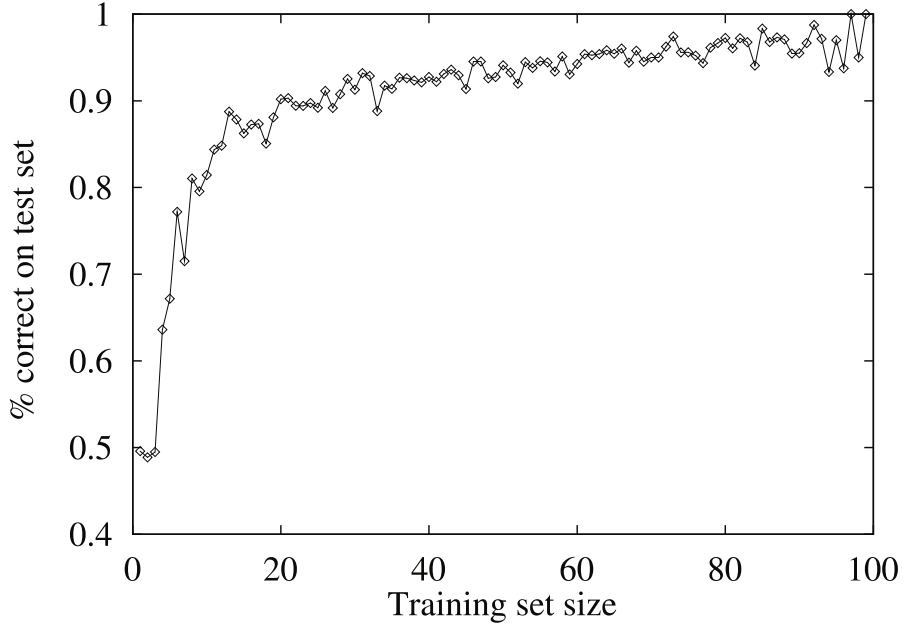
افزونه: مثلاً در اثر بار خصیصه‌های نامربوط

تحقق‌ناپذیر:
فضای فرضیه نمی‌تواند تابع هدف
را بیان کند. (در اثر خصیصه‌های گم‌شده یا کلاس فرضیه‌های محدودشده)

ارزیابی کارآیی

منحنی یادگیری: مثال

PERFORMANCE MEASUREMENT



منحنی یادگیری الگوریتم درخت تصمیم با ۱۰۰ نمونه‌ی تصادفی برای مسئله‌ی رستوران

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۴

ارزیابی و انتخاب بهترین فرضیه

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۵

نظریه‌ی یادگیری

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۶

رگرسیون و طبقه‌بندی با مدل‌های خطی

هوش مصنوعی پیشرفته

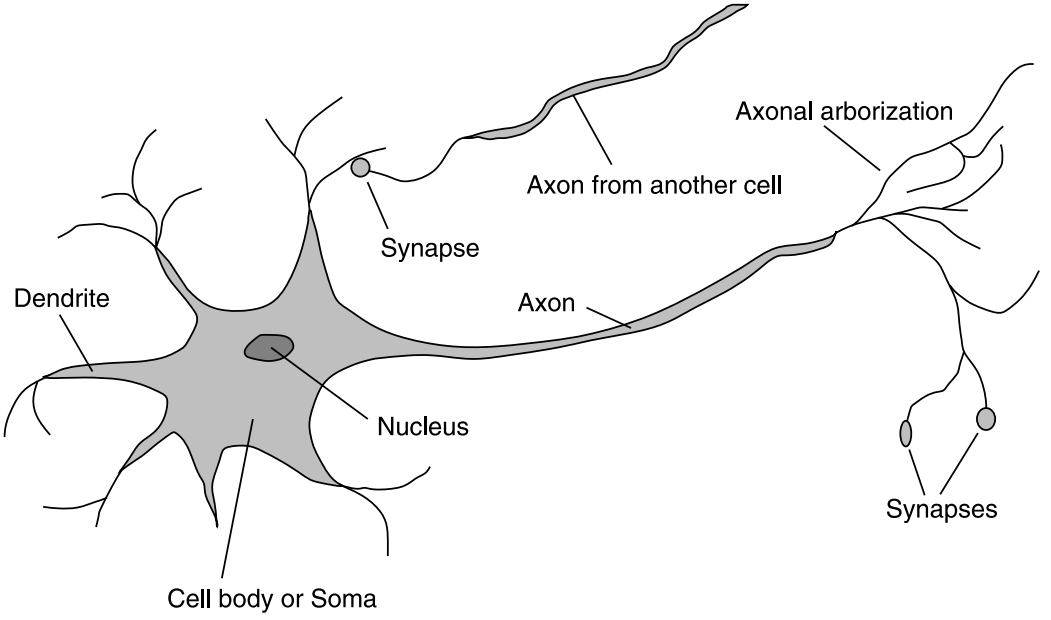
یادگیری از مثال‌ها

۷

شبکه‌های عصبی مصنوعی

مغز

BRAIN

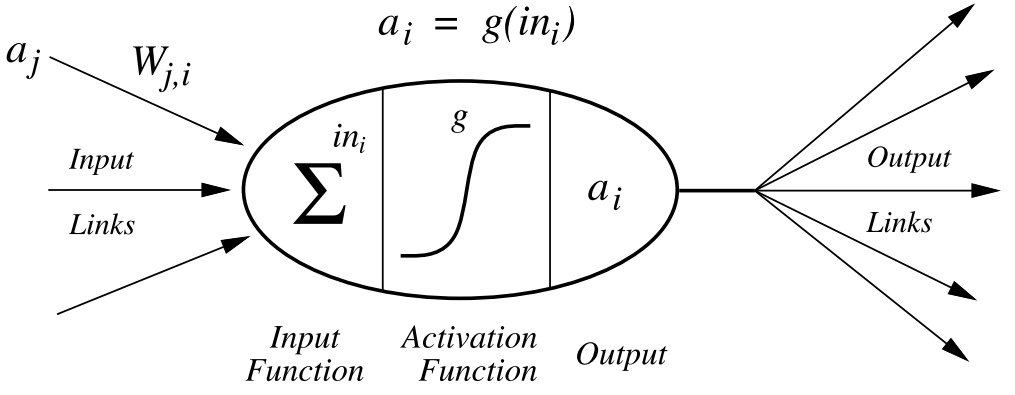


- 10^{11} neurons of > 20 types, 10^{14} synapses, 1ms–10ms cycle time
- Signals are noisy “spike trains” of electrical potential

شبکه‌های عصبی مصنوعی

واحد «مک‌کلوچ–پیتز»

McCulloch–Pitts "unit"



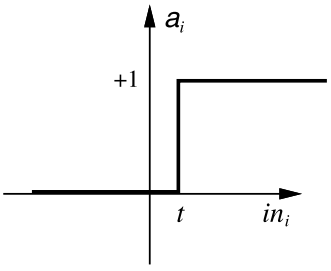
یک مدل بسیار ساده‌شده از نرون واقعی:

با هدف توسعه‌ی درک اینکه شبکه‌های متشکل از واحدهای ساده چه کاری می‌توانند انجام دهند.

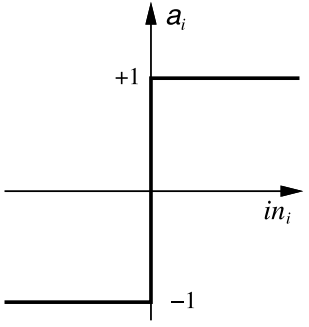
شبکه‌های عصبی مصنوعی

توابع فعال‌سازی

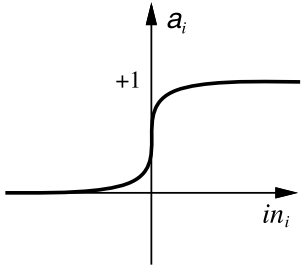
ACTIVATION FUNCTIONS



(a) Step function



(b) Sign function



(c) Sigmoid function

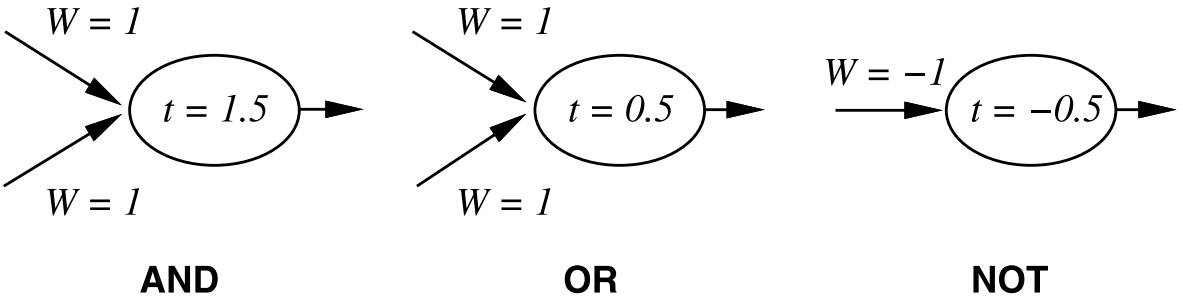
$1/(1+e^{-x})$

Changing the bias weight $W_{0,i}$ moves the threshold location

شبکه‌های عصبی مصنوعی

پیاده‌سازی توابع منطقی

IMPLEMENTING LOGICAL FUNCTIONS



شبکه‌های عصبی مصنوعی

ساختارهای شبکه

NETWORK STRUCTURES

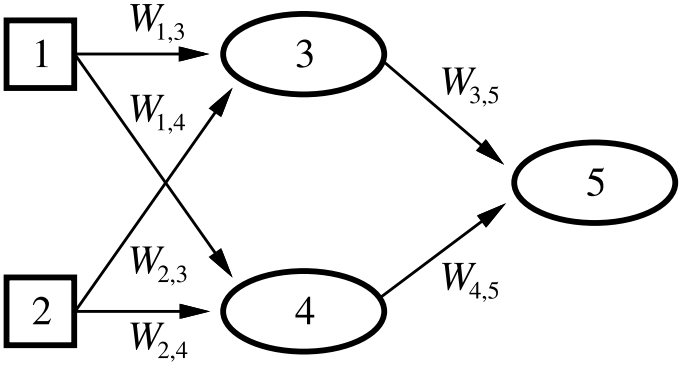
ساختارهای شبکه‌های عصبی مصنوعی			
شبکه‌های پیش‌خور *Feed-forward Networks*		**شبکه‌های بازگشتی** *Recurrent Networks*	
توابع بدون حالت داخلی را پیاده‌سازی می‌کنند.		توابع دارای حالت داخلی (مانند فلیپ‌فلاپ) را پیاده‌سازی می‌کنند.	
بدون فیدبک		با فیدبک / چرخه‌های جهت‌دار با تاخیرها	
پرسپترون تک‌لایه *Single-Layer Perceptron*	**پرسپترون چندلایه** *Single-Layer Perceptron*	**شبکه‌های هاپفیلد** *Hopfield Networks*	**ماشین‌های بولتزمن** *Boltzmann Machines*
		دارای وزن‌های متقارن تابع فعال‌سازی علامت حافظه‌ی پیوندی هولوگرافیک	تابع فعال‌سازی اتفاقی ≈ MCMC در شبکه‌های بیزی

$$g(x) = \text{sign}(x),\ a_i = \pm 1$$

شبکه‌های عصبی مصنوعی

ساختارهای شبکه: شبکه‌ی پیش‌خور: مثال

FEED-FORWARD EXAMPLE



شبکه‌ی پیش‌خور = یک خانواده‌ی پارامتری‌شده از توابع غیرخطی

$$\begin{aligned} a_5 &= g(W_{3,5} \cdot a_3 + W_{4,5} \cdot a_4) \\ &= g(W_{3,5} \cdot g(W_{1,3} \cdot a_1 + W_{2,3} \cdot a_2) + W_{4,5} \cdot g(W_{1,4} \cdot a_1 + W_{2,4} \cdot a_2)) \end{aligned}$$

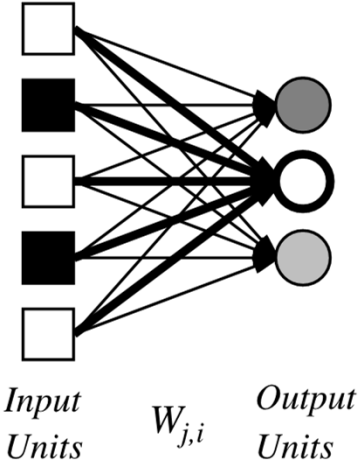
تنظیم وزن‌ها، تابع را تغییر می‌دهد: انجام یادگیری از این طریق

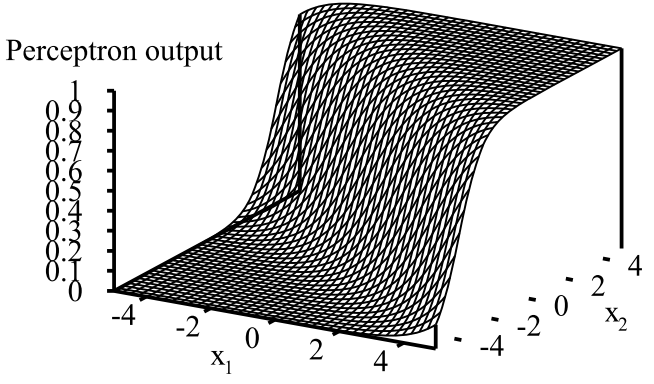
شبکه‌های عصبی مصنوعی

پرسپترون‌های تک‌لایه

SINGLE-LAYER PERCEPTRONS

خروجی همه‌ی واحدها همگی جداگانه عمل می‌کنند (هیچ وزن مشترکی وجود ندارد).





شکل کلی رویه‌ی تصمیم‌گیری (حالت دو متغیر ورودی):
تنظیم وزن‌ها، مکان، جهت و شیب صخره را حرکت می‌دهد.

شبکه‌های عصبی مصنوعی

پرسپترون تک‌لایه: رسایی پرسپترون‌ها

EXPRESSIVENESS OF PERCEPTRONS

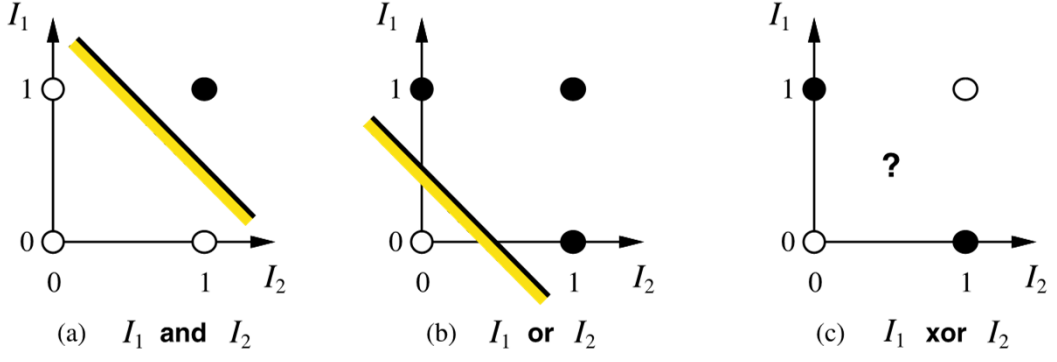
پرسپترون با تابع فعال‌سازی پله = g

پرسپترون یک جداساز خطی را در فضای ورودی بازنمایی می‌کند:

$$\Sigma_j W_j x_j > 0 \quad \text{or} \quad \mathbf{W} \cdot \mathbf{x} > 0$$

پرسپترون نمی‌تواند تابع XOR را بازنمایی کند.

پرسپترون می‌تواند توابع AND، OR، NOT و Majority و ... را بازنمایی کند.



شبکه‌های عصبی مصنوعی

پرسپترون تک‌لایه : یادگیری پرسپترون

PERCEPTRON LEARNING

یادگیری با تنظیم وزن‌ها برای کاهش خطا بر روی مجموعه‌ی آموزشی

مجذور خطا برای یک مثال با ورودی $\mathbf{x}$ و خروجی واقعی y :

$$E = \frac{1}{2}Err^2 \equiv \frac{1}{2}(y - h_{\mathbf{W}}(\mathbf{x}))^2 \ ,$$

انجام جستجوی بهینه‌سازی با کاهش گرادیان :

$$\begin{aligned}\frac{\partial E}{\partial W_j} &= Err \times \frac{\partial Err}{\partial W_j} = Err \times \frac{\partial}{\partial W_j}\left(y - g(\Sigma_{j=0}^{n} W_j x_j)\right) \\ &= -Err \times g'(in) \times x_j\end{aligned}$$

قاعده‌ی ساده برای به‌هنگام‌سازی وزن‌ها :

$$W_j \leftarrow W_j + \alpha \times Err \times g'(in) \times x_j$$

برای مثال:

خطای مثبت ⇐ افزایش خروجی شبکه

⇐ لزوم افزایش وزن‌ها روی ورودی‌های مثبت، کاهش وزن‌ها روی ورودی‌های منفی

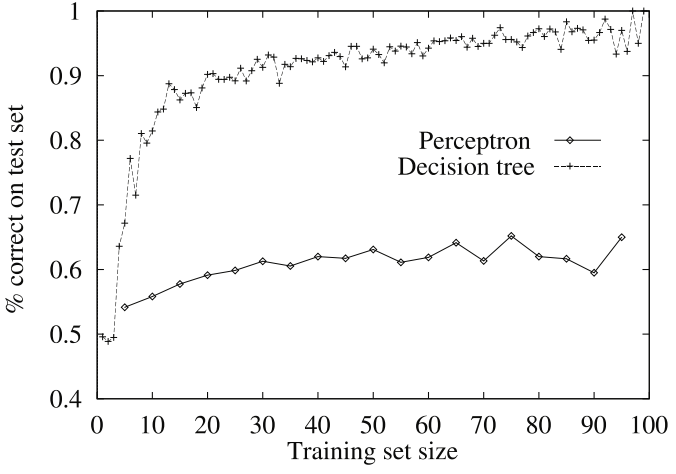
شبکه‌های عصبی مصنوعی

پرسپترون تک‌لایه: یادگیری پرسپترون: ویژگی‌ها

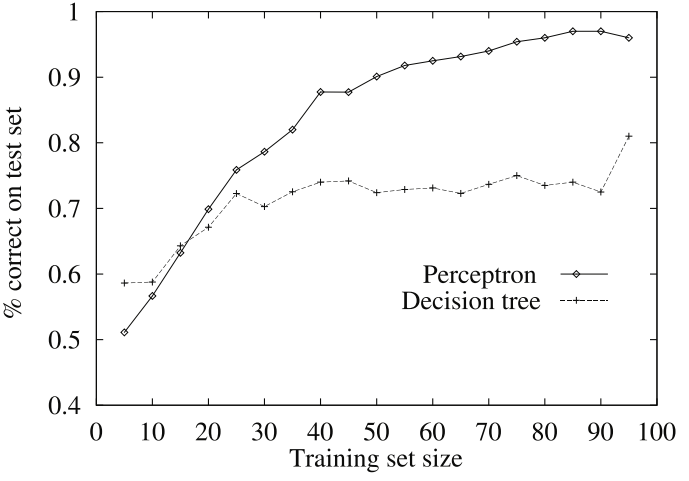
PERCEPTRON LEARNING

قاعده‌ی یادگیری پرسپترون، به یک تابع سازگار همگرا می‌شود

برای هر مجموعه داده‌ی جدایی‌پذیر خطی



تابع مسئله‌ی رستوران



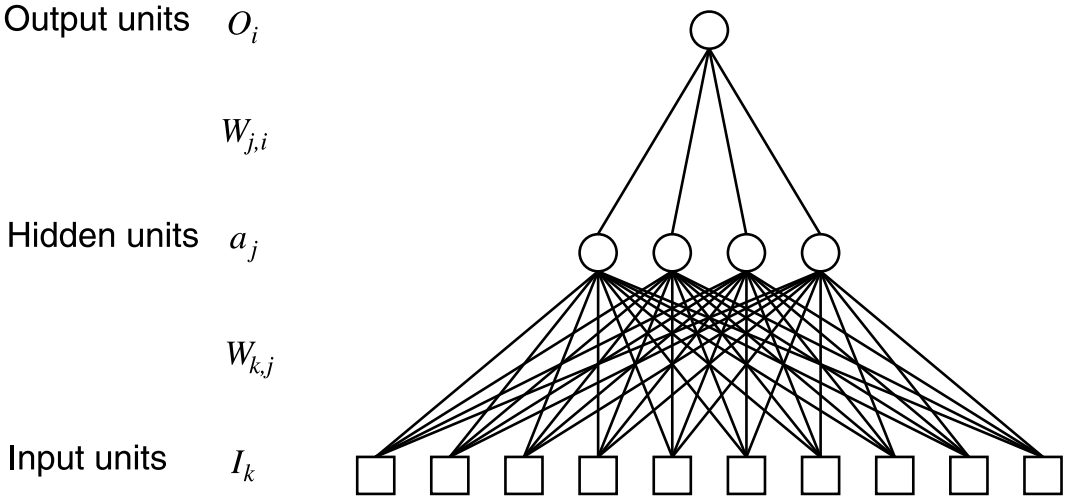
تابع اکثریت *Majority*

شبکه‌های عصبی مصنوعی

پرسپترون‌های چندلایه

MULTILAYER PERCEPTRONS

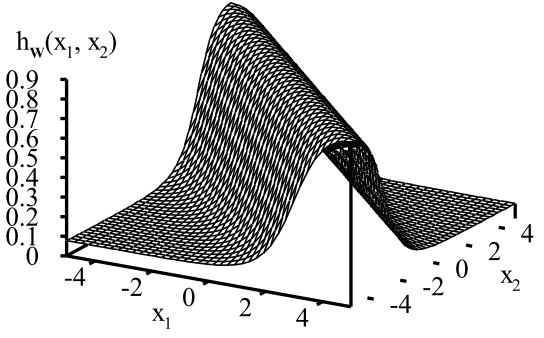
لایه‌ها معمولاً به طور کامل به هم متصل هستند.
تعداد واحدهای لایه‌ی پنهان معمولاً دستی تعیین می‌شود.

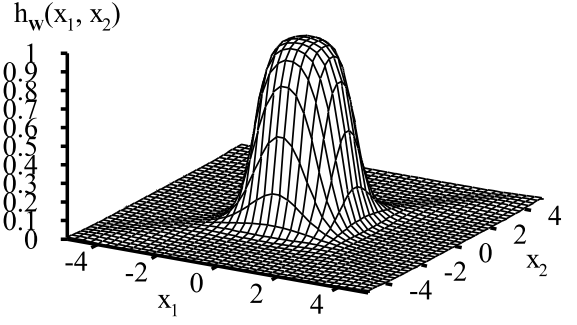


شبکه‌های عصبی مصنوعی

پرسپترون‌های چندلایه: رسائی

EXPRESSIVENESS OF MLPS





همه‌ی توابع پیوسته با دو لایه

همه‌ی توابع با سه لایه

ترکیب دو تابع آستانه با وجوه مخالف ⇐ ایجاد تیغه

ترکیب دو تیغه‌ی متعامد ⇐ ایجاد یک برآمدگی

اضافه کردن برآمدگی‌ها با اندازه‌ها و مکان‌های گوناگون برای برازش هر رویه‌ی دلخواه

(نیاز به تعداد نمایی واحد پنهان)

شبکه‌های عصبی مصنوعی

پرسپترون‌های چندلایه : یادگیری پس‌انتشار

BACK-PROPAGATION LEARNING

لایه‌ی خروجی: مشابه لایه‌ی پرسپترون تک‌لایه

$$W_{j,i} \leftarrow W_{j,i} + \alpha \times a_j \times \Delta_i$$

$$\Delta_i = Err_i \times g'(in_i)$$

لایه‌ی پنهان: خطا از لایه‌ی خروجی به پشت منتشر می‌شود **(back-propagate)** :

$$\Delta_j = g'(in_j) \sum_i W_{j,i} \Delta_i$$

قاعده‌ی به‌هنگام‌سازی برای وزن‌ها در لایه‌ی پنهان:

$$W_{k,j} \leftarrow W_{k,j} + \alpha \times a_k \times \Delta_j$$

(بسیاری از نروساینتیست‌ها انکار می‌کنند که در مغز پس‌انتشار رخ می‌دهد!)

شبکه‌های عصبی مصنوعی

پرسپترون‌های چندلایه: یادگیری پس‌انتشار: استخراج فرمول (۱)

BACK-PROPAGATION LEARNING

مجذور خطا بر روی یک مثال واحد به صورت زیر تعریف می‌شود:

$$E = \frac{1}{2}\sum_i (y_i - a_i)^2 \ ,$$

که در آن مجموع روی همه‌ی گره‌ها در لایه‌ی خروجی محاسبه می‌شود.

$$\begin{aligned}
\frac{\partial E}{\partial W_{j,i}} &= -(y_i - a_i)\frac{\partial a_i}{\partial W_{j,i}} = -(y_i - a_i)\frac{\partial g(in_i)}{\partial W_{j,i}} \\
&= -(y_i - a_i)g'(in_i)\frac{\partial in_i}{\partial W_{j,i}} = -(y_i - a_i)g'(in_i)\frac{\partial}{\partial W_{j,i}}\left(\sum_j W_{j,i}a_j\right) \\
&= -(y_i - a_i)g'(in_i)a_j = -a_j\Delta_i
\end{aligned}$$

شبکه‌های عصبی مصنوعی

پرسپترون‌های چندلایه: یادگیری پس‌انتشار: استخراج فرمول (۲)

BACK-PROPAGATION LEARNING

$$
\begin{aligned}
\frac{\partial E}{\partial W_{k,j}} &= -\sum_i (y_i - a_i)\frac{\partial a_i}{\partial W_{k,j}} = -\sum_i (y_i - a_i)\frac{\partial g(in_i)}{\partial W_{k,j}} \\
&= -\sum_i (y_i - a_i) g'(in_i)\frac{\partial in_i}{\partial W_{k,j}} = -\sum_i \Delta_i \frac{\partial}{\partial W_{k,j}}\left(\sum_j W_{j,i} a_j\right) \\
&= -\sum_i \Delta_i W_{j,i}\frac{\partial a_j}{\partial W_{k,j}} = -\sum_i \Delta_i W_{j,i}\frac{\partial g(in_j)}{\partial W_{k,j}} \\
&= -\sum_i \Delta_i W_{j,i} g'(in_j)\frac{\partial in_j}{\partial W_{k,j}} \\
&= -\sum_i \Delta_i W_{j,i} g'(in_j)\frac{\partial}{\partial W_{k,j}}\left(\sum_k W_{k,j} a_k\right) \\
&= -\sum_i \Delta_i W_{j,i} g'(in_j) a_k = -a_k \Delta_j
\end{aligned}
$$

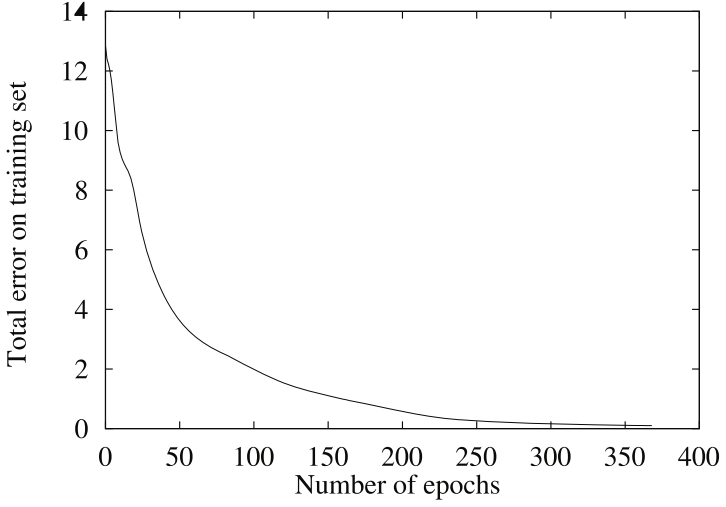
شبکه‌های عصبی مصنوعی

پرسپترون‌های چندلایه: یادگیری پس‌انتشار: ویژگی‌ها

BACK-PROPAGATION LEARNING

در هر اپک (epoch)، بهنگام‌سازی‌های گرادیان را برای همه‌ی مثال‌ها جمع بزنید و اعمال کنید.

منحنی آموزش **(training curve)**



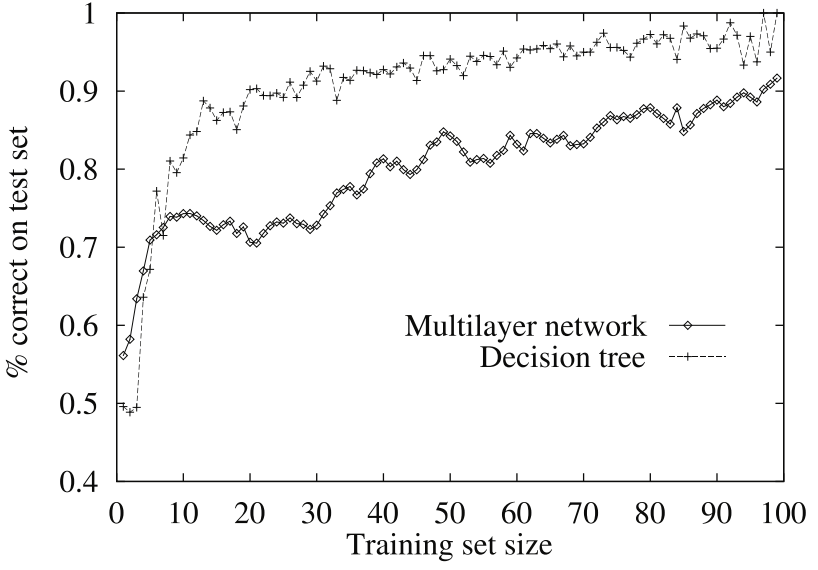
مشکل نوعی: همگرایی کند، می‌نیمم‌های محلی

شبکه‌های عصبی مصنوعی

پرسپترون‌های چندلایه: یادگیری پس‌انتشار: ویژگی‌ها

BACK-PROPAGATION LEARNING

منحنی یادگیری برای MLP با ۴ واحد پنهان



MLPها برای وظایف بازشناسی الگوی پیچیده بسیار خوب هستند،
اما فرضیه‌های حاصل نمی‌توانند به‌سادگی فهمیده شوند.

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۸

مدل‌های ناپارامتری

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۹

ماشین‌های بردار پشتیبان

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۱۰

یادگیری جمعی

هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۱۱

یادگیری ماشینی عملی

بازشناسی اعداد دست‌نویس

مثالی از یادگیری بانظارت

HANDWRITTEN DIGIT RECOGNITION

3-nearest-neighbor = 2.4% error
400–300–10 unit MLP = 1.6% error
LeNet: 768–192–30–10 unit MLP = 0.9% error

Current best (kernel machines, vision algorithms) ≈ 0.6% error

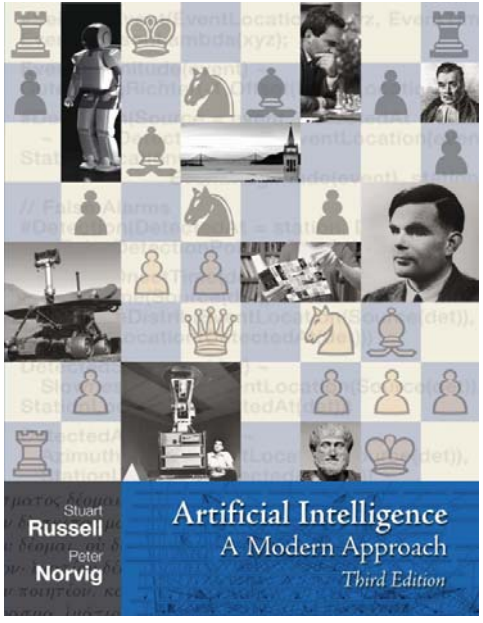
هوش مصنوعی پیشرفته

یادگیری از مثال‌ها

۱۲

منابع، مطالعه، تکلیف

منبع اصلی



Stuart Russell and Peter Norvig,
Artificial Intelligence: A Modern Approach,
3rd Edition, Prentice Hall, 2010.

Chapter 18

18 LEARNING FROM EXAMPLES

In which we describe agents that can improve their behavior through diligent study of their own experiences.

LEARNING

An agent is **learning** if it improves its performance on future tasks after making observations about the world. Learning can range from the trivial, as exhibited by jotting down a phone number, to the profound, as exhibited by Albert Einstein, who inferred a new theory of the universe. In this chapter we will concentrate on one class of learning problem, which seems restricted but actually has vast applicability: from a collection of input–output pairs, learn a function that predicts the output for new inputs.

Why would we want an agent to learn? If the design of the agent can be improved, why wouldn't the designers just program in that improvement to begin with? There are three main reasons. First, the designers cannot anticipate all possible situations that the agent might find itself in. For example, a robot designed to navigate mazes must learn the layout of each new maze it encounters. Second, the designers cannot anticipate all changes over time; a program designed to predict tomorrow's stock market prices must learn to adapt when conditions change from boom to bust. Third, sometimes human programmers have no idea how to program a solution themselves. For example, most people are good at recognizing the faces of family members, but even the best programmers are unable to program a computer to accomplish that task, except by using learning algorithms. This chapter first gives an overview of the various forms of learning, then describes one popular approach, decision-tree learning, in Section 18.3, followed by a theoretical analysis of learning in Sections 18.4 and 18.5. We look at various learning systems used in practice: linear models, nonlinear models (in particular, neural networks), nonparametric models, and support vector machines. Finally we show how ensembles of models can outperform a single model.

18.1 FORMS OF LEARNING

Any component of an agent can be improved by learning from data. The improvements, and the techniques used to make them, depend on four major factors:

- Which *component* is to be improved.

693