

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ



هوش مصنوعی

فصل ۱۸

یادگیری از مثال‌ها

Learning from Examples

کاظم فولادی
دانشکده مهندسی برق و کامپیوتر
دانشگاه تهران

<http://courses.fouladi.ir/ai>

هوش مصنوعی

یادگیری از مثال‌ها



مقدمه

عامل‌های یادگیرنده

یادگیری

LEARNING AGENTS

یادگیری، مکانیزم‌های تصمیم‌گیری عامل را به گونه‌ای 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 به تنهایی ممکن بود اجازه بدهد، شایسته‌تر شود.

ساخت عامل‌های هوش مصنوعی

ایجاد مؤلفه‌ی یادگیرنده و آموزش عامل

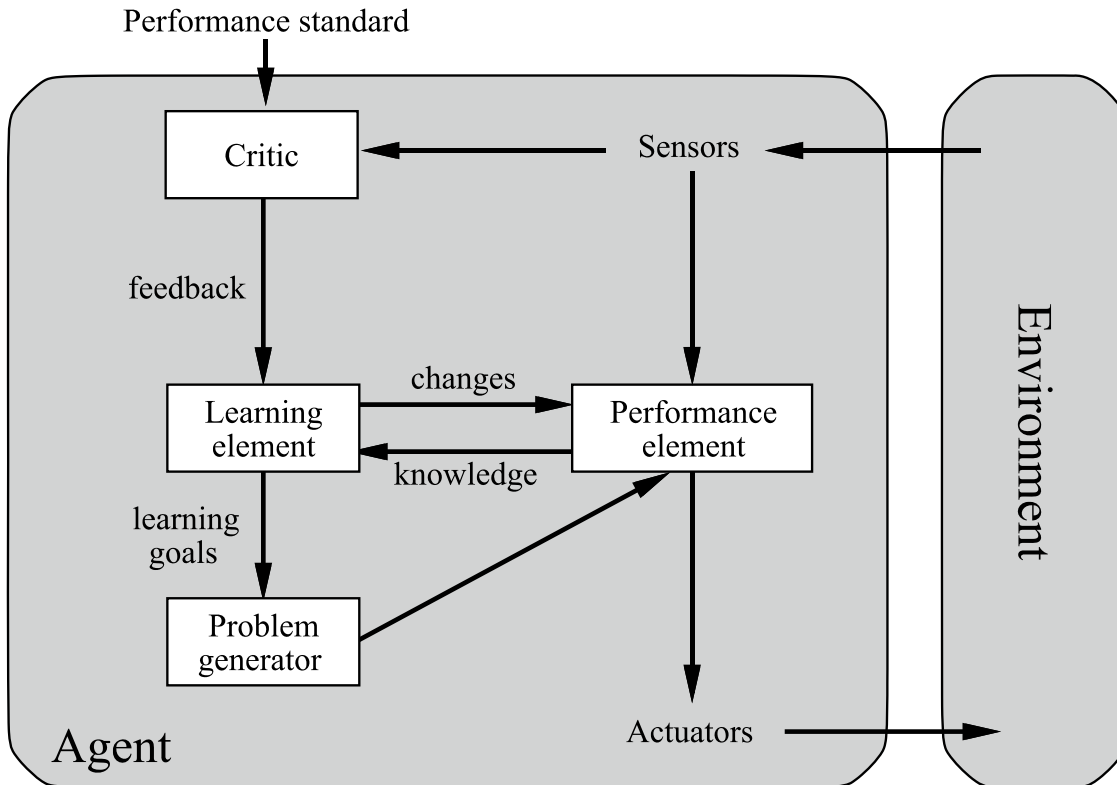
برنامه‌نویسی از صفر

● 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 کارایی استاندارد: Performance standard



منتقد

Critic

به عنصر یادگیرنده
فیدبک می‌دهد:

با توجه به یک معیار
استاندارد کارایی ثابت به
عنصر یادگیرنده می‌گوید
که عامل چه قدر خوب
عمل کرده است.

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

مسئول بهبود
بخشیدن به کارایی:

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

عنصر یادگیرنده

Learning
element

عنصر انجام‌دهنده

Performance
element

مسئول انتخاب کنش بیرونی:

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

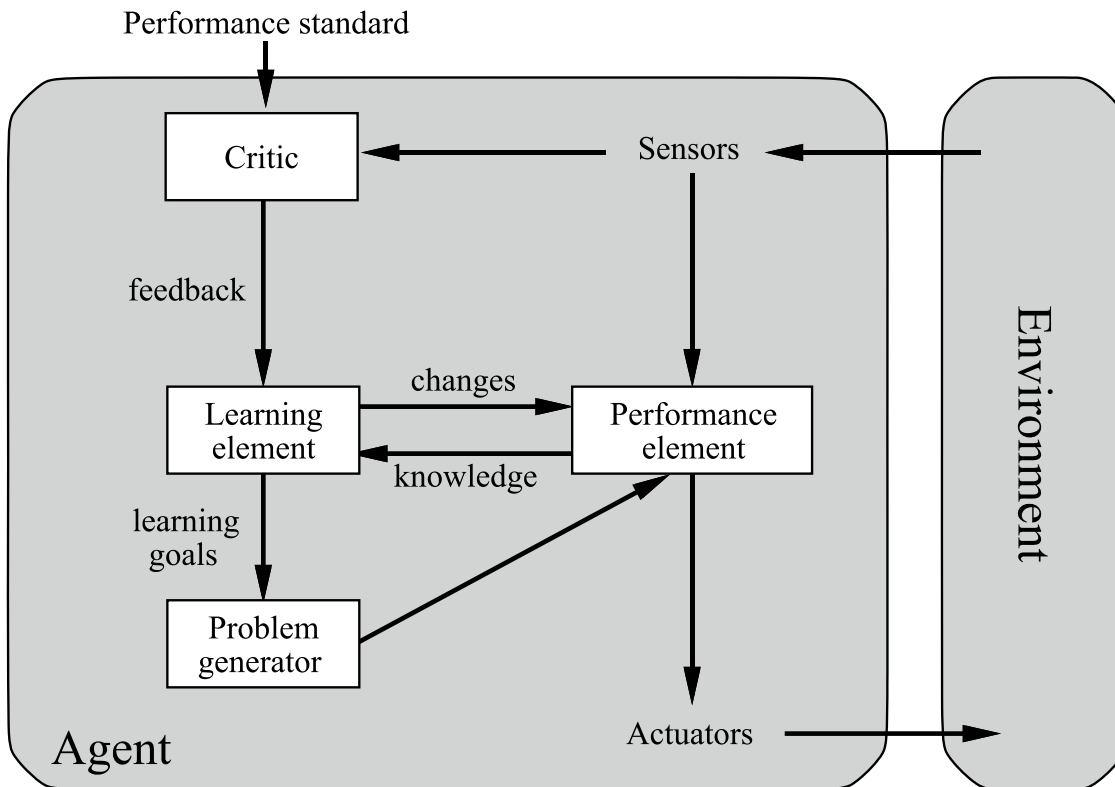
مولد مسئله

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 کارایی کل عامل بهبود یابد.

LEARNING AGENTS

یادگیری از مثال‌ها

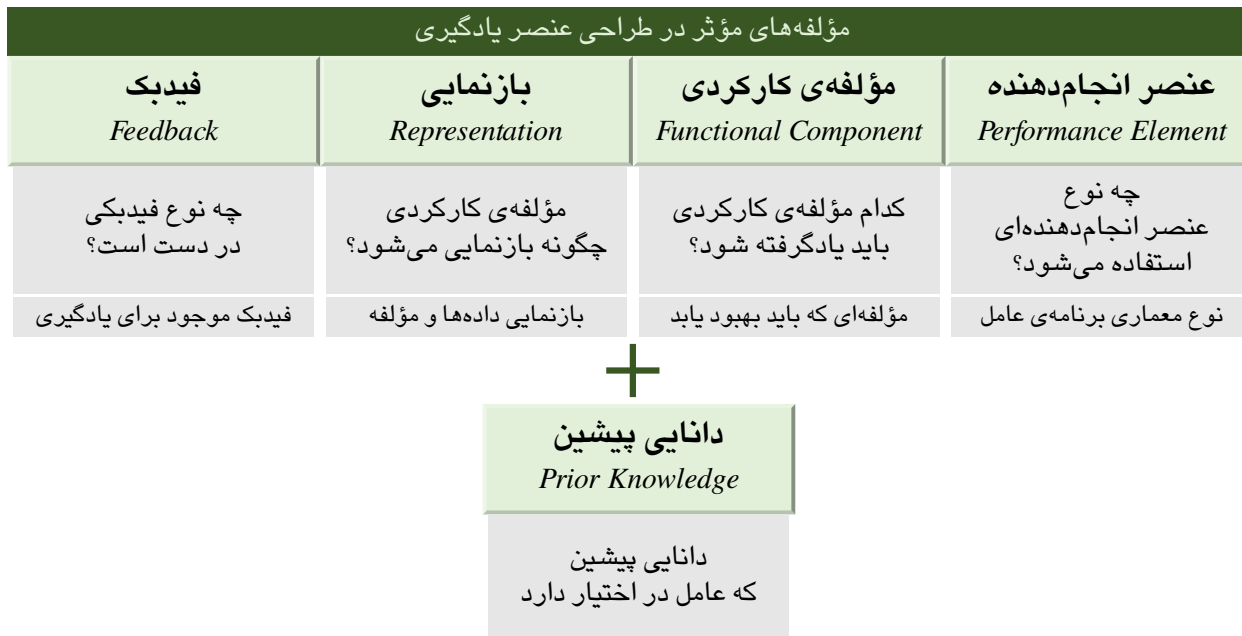
۱

صورت‌های یادگیری

عنصر یادگیری

LEARNING ELEMENT

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



عنصر یادگیری

مثال

LEARNING ELEMENT

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

صورت‌های یادگیری

FORMS OF LEARNING

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

یادگیری استقرائی

آنچه ساینس (science) نامیده می شود

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 - خروجی

یادگیری نیمه نظارتی

Semisupervised

یادگیری تقویتی

Reinforcement

یادگیری بی نظارت

Unsupervised

یادگیری بانظارت

Supervised

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

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:

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

که در آن هر y_j به وسیله‌ی 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 <i>Supervised Learning</i>	
مجموعه‌ی جفت‌های ورودی-خروجی معلوم برای آموزش یادگیرنده	مجموعه‌ی آموزشی <i>Training Set</i>
مجموعه‌ی جفت‌های ورودی-خروجی معلوم برای آزمایش یادگیرنده	مجموعه‌ی آزمایشی <i>Test Set</i>
تابع f که باید یاد گرفته شود	هدف <i>Target</i>
تابع مجهول h که باید تقریب مناسبی برای f باشد	فرضیه <i>Hypothesis</i>
مجموعه‌ی همه‌ی توابع منتخب h برای تقریب f	فضای فرضیه <i>Hypothesis Space</i>

 $\mathcal{H}$

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.,

O	O	X
	X	
X		

, $+1$

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

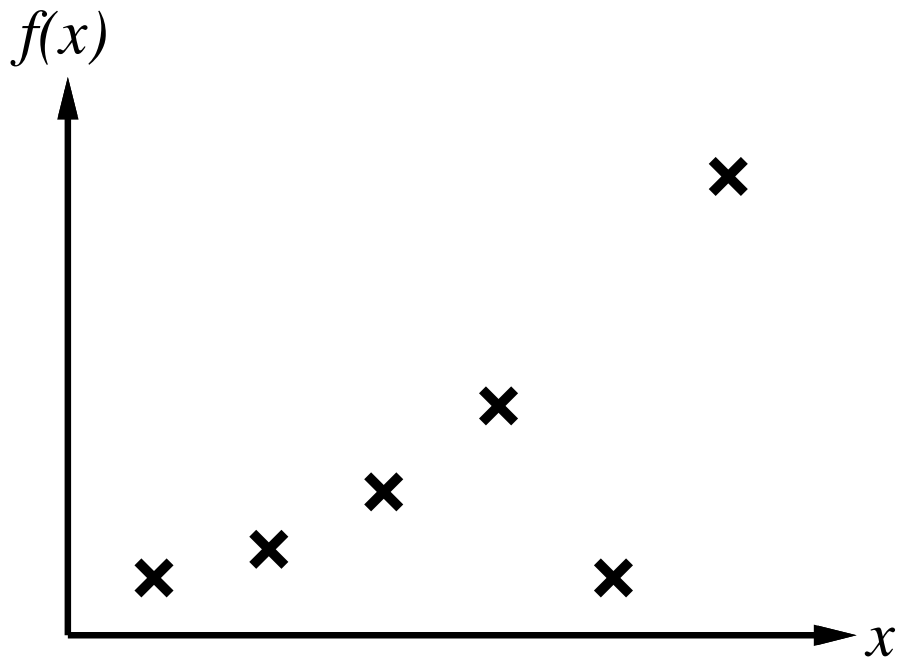
یادگیری بانظارت

طبقه‌بندی و رگرسیون

SUPERVISED LEARNINGیادگیری بانظارت
*Supervised Learning*طبقه‌بندی
*Classification*وقتی خروجی y از یک مجموعه‌ی متناهی مشخص انتخاب شود.رگرسیون
*Regression*وقتی خروجی y یک عدد باشد (یافتن امید شرطی یا متوسط y)

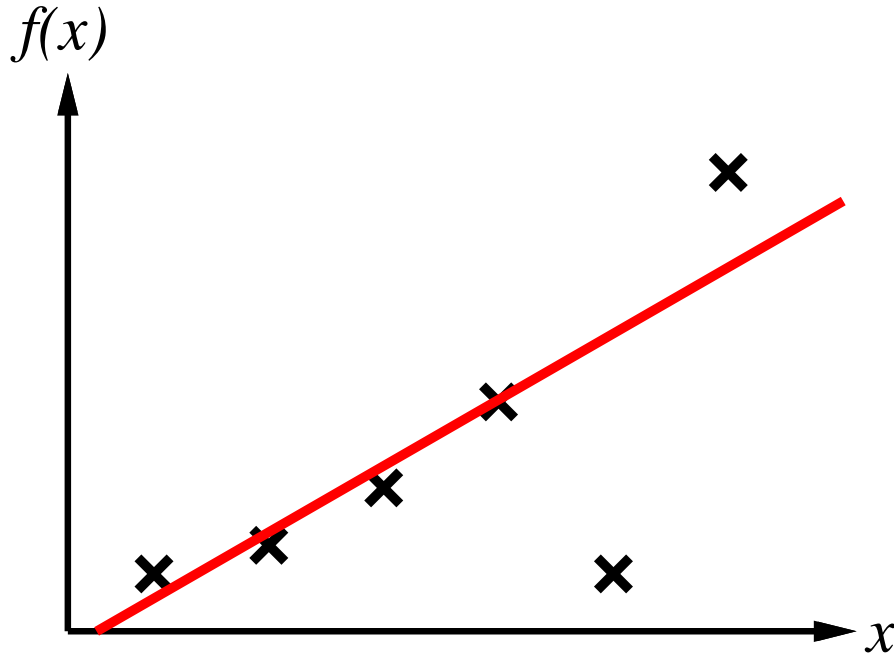
روش یادگیری بانظارت

مثال: برازش منحنی (۱ از ۵)

SUPERVISED LEARNING METHOD

روش یادگیری بانظارت

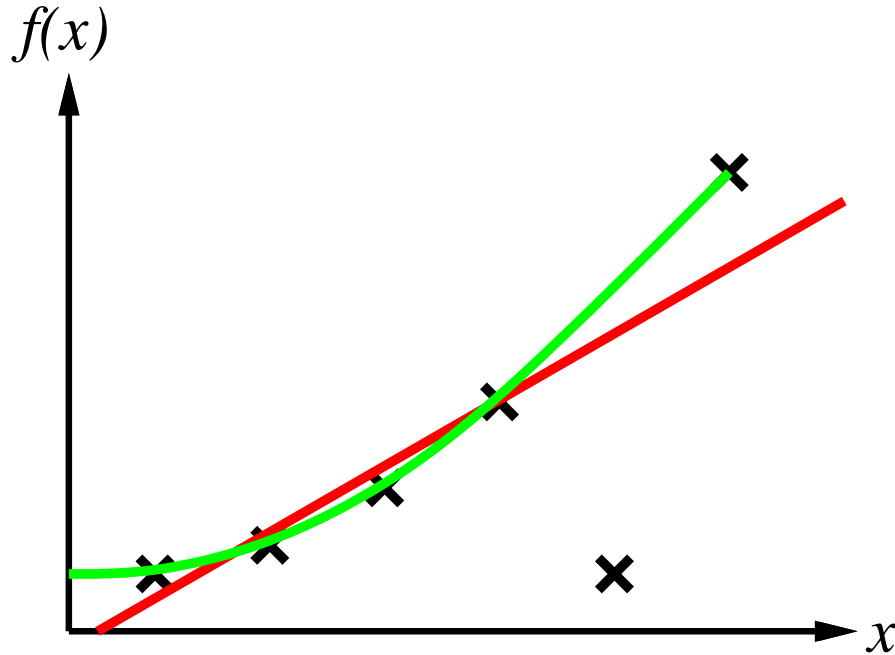
مثال: برازش منحنی (۲ از ۵)

SUPERVISED LEARNING METHOD

فرضیه بسیار ساده (چند جمله ای خطی)، ناسازگار، تعمیم پذیری خوب

روش یادگیری استقرائی

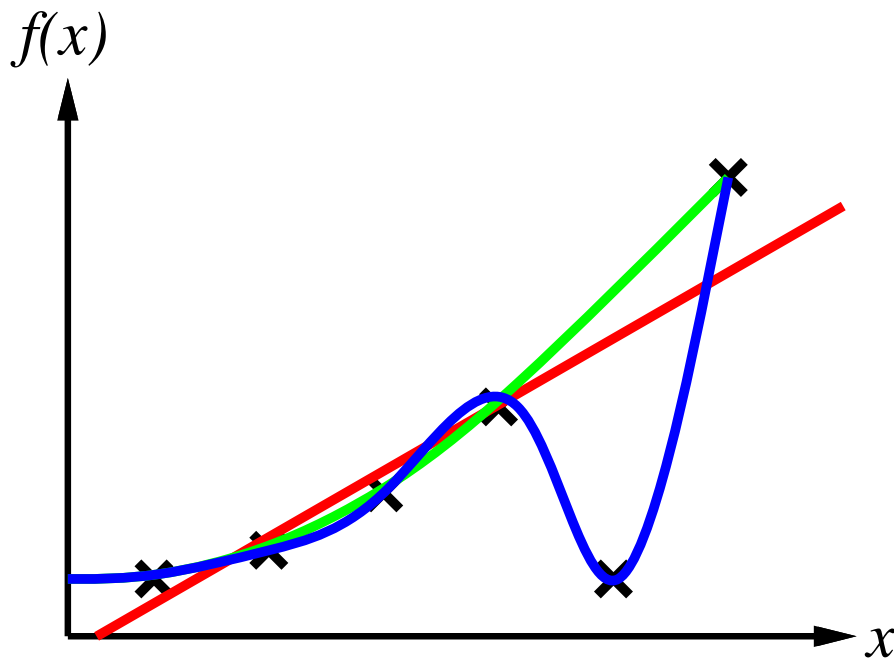
مثال: برازش منحنی (۳ از ۵)

SUPERVISED LEARNING METHOD

فرضیه ساده (چندجمله‌ای درجه دوم)، ناسازگار، تعمیم‌پذیری خوب

روش یادگیری بانظارت

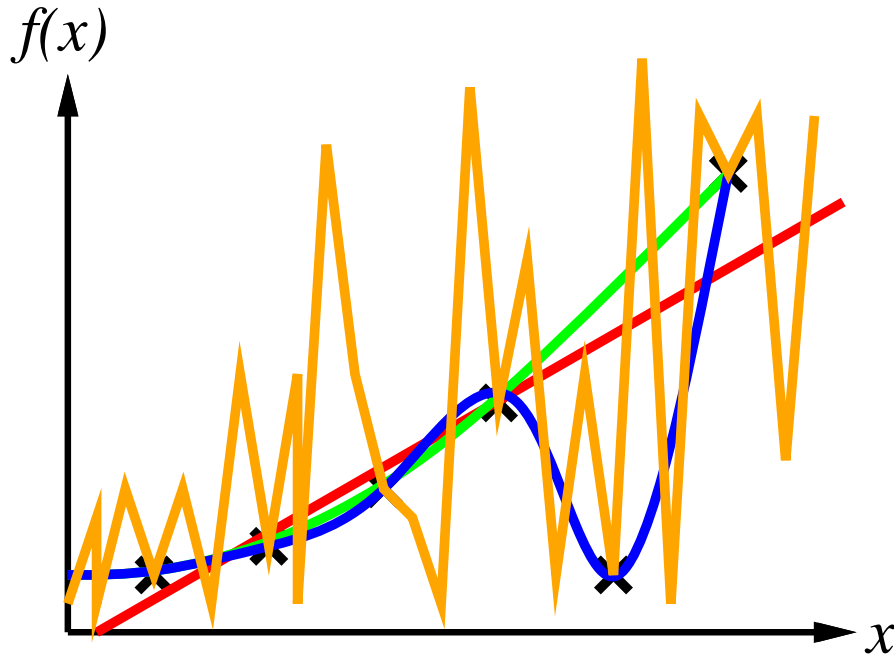
مثال: برازش منحنی (۴ از ۵)

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

تعمیم <i>Generalization</i>	سازگاری <i>Consistency</i>
یک فرضیه تعمیم پذیر است اگر مقادیر خروجی نمونه‌های جدید را به درستی پیش‌بینی کند.	یک فرضیه سازگار است اگر بر روی همه‌ی نمونه‌های آموزشی درست باشد.

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

بده‌بستان میان رسایی - 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.

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

«به نام ویلیام اوخامی فیلسوف انگلیسی قرن 14 میلادی»

تیغهی اوخامی

Ockham's Razor

یادگیری بانظارت

محتمل‌ترین فرضیه برای داده‌های موجود

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

محتمل‌ترین فرضیه با داشتن داده‌ها

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

عبارت معادل از طریق قاعده‌ی بیز

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

یادگیری از مثال‌ها

۳

یادگیری
درخت‌های
تصمیم

بازنمایی‌های مبتنی بر خصیصه

مثال: انتظار برای میز در یک رستوران

ATTRIBUTE-BASED REPRESENTATIONS

مثال

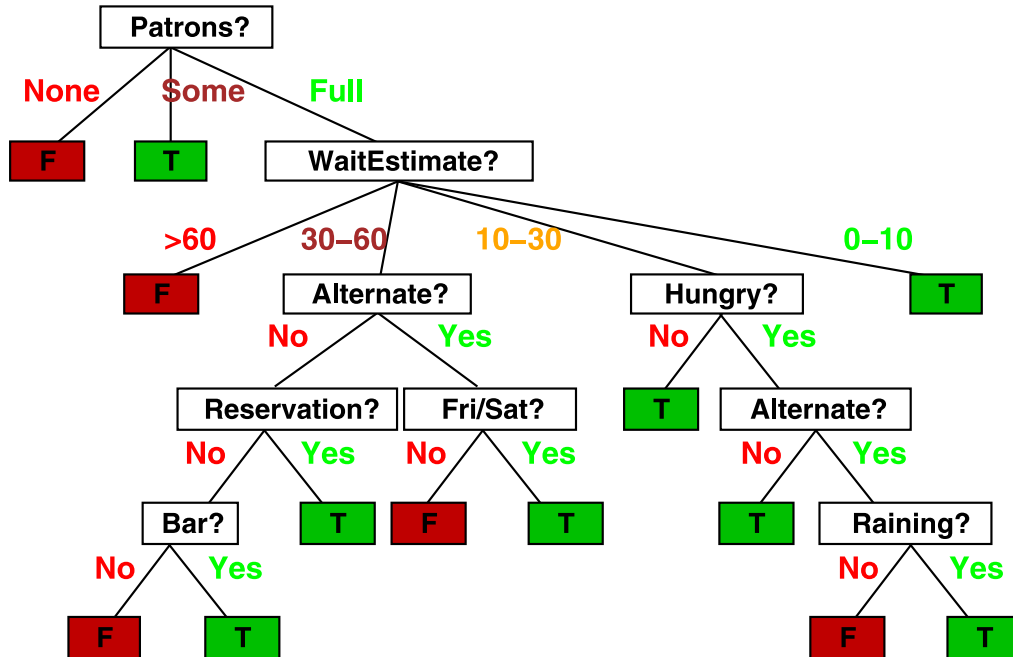
مثال‌هایی از موقعیت‌هایی که در آنها برای یک میز در رستوران صبر می‌کنیم یا خیر:

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

درخت‌های تصمیم

مثال: انتظار برای میز در یک رستوران

DECISION TREES



درخت تصمیم: یک بازنمایی ممکن برای فرضیه‌ها

خصیصه‌های بی‌ربط در تصمیم‌گیری (price و type) 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 $\leftarrow$ یک مسیر از ریشه تا برگ در درخت تصمیم

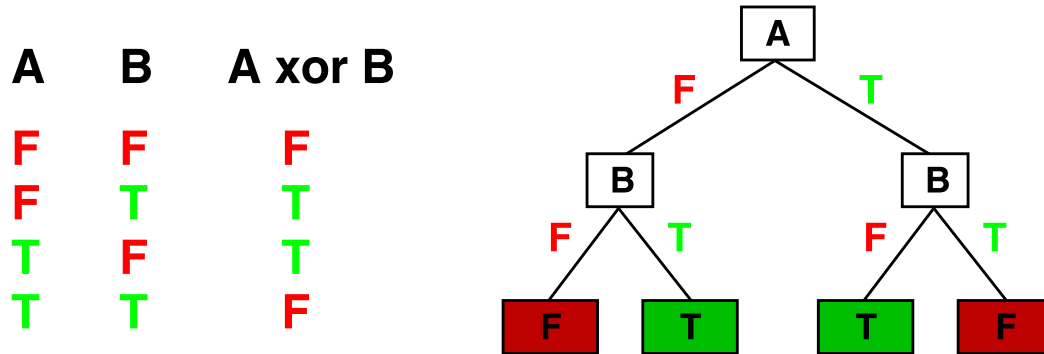
برای هر مجموعه‌ی آموزشی، یک درخت تصمیم سازگار وجود دارد:
با یک مسیر از ریشه تا یک برگ برای هر نمونه
(مگر اینکه f بر حسب x غیرقطعی باشد).
اما احتمال دارد که تعمیم‌پذیری آن برای مثال‌های جدید پایین باشد.

یافتن درخت‌های تصمیم متراکم‌تر ترجیح داده می‌شود.

درخت‌های تصمیم

مثال: تابع یای انحصاری

EXPRESSIVENESS



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

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 به عنوان ریشه‌ی (زیر) درخت انتخاب کنید (به صورت بازگشتی)

```

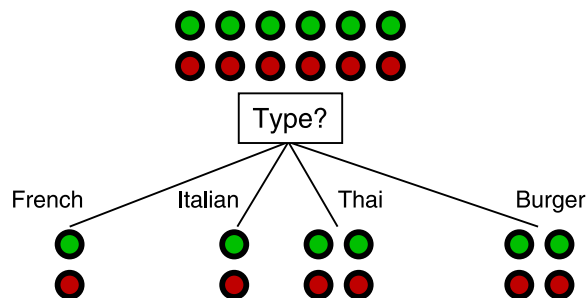
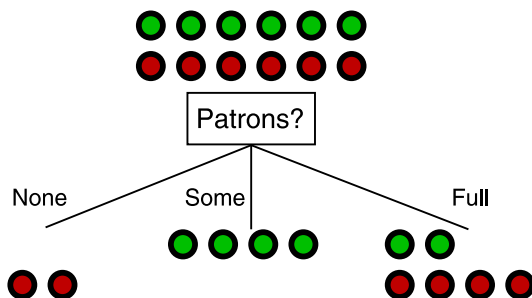
function DTL(examples, attributes, default) returns a decision tree
  if examples is empty then return default
  else if all examples have the same classification then return the classification
  else if attributes is empty then return MODE(examples)
  else
    best ← CHOOSE-ATTRIBUTE(attributes, examples)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

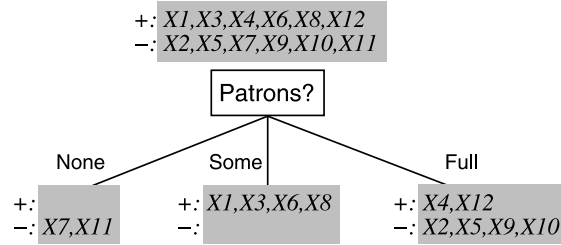
ایده: یک خصیصه‌ی خوب، مثال‌ها را به مجموعه‌هایی که (در حالت ایده‌آل) «همه مثبت» یا «همه منفی» هستند، می‌شکند.



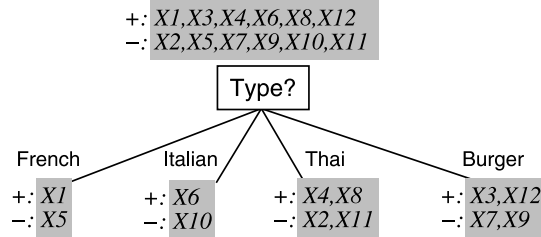
Patrons? گزینه‌ی بهتری است:

در مورد طبقه‌بندی اطلاعات می‌دهد.

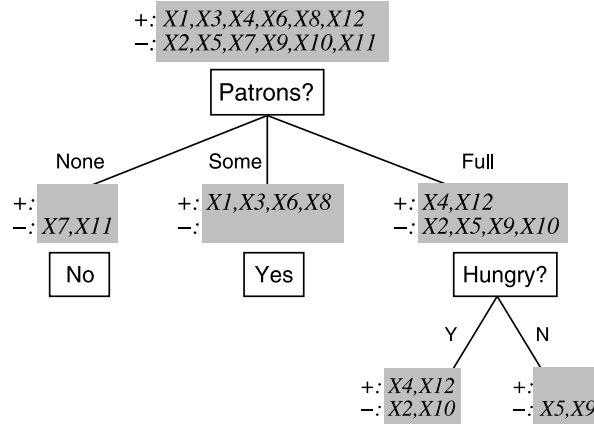
(a)



(b)



(c)



اطلاعات

آنتروپی

INFORMATION

اطلاعات به پرسش‌ها پاسخ می‌دهد

اطلاعات
Information

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

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

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

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

(موسوم به آنتروپی پیشین)

درخت تصمیم

استفاده از اطلاعات برای انتخاب خصیصه

USING INFORMATION

Suppose we have p positive and n negative examples at the root

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

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

An attribute splits the examples E into subsets E_i , each of which (we hope) needs less information to complete the classification

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

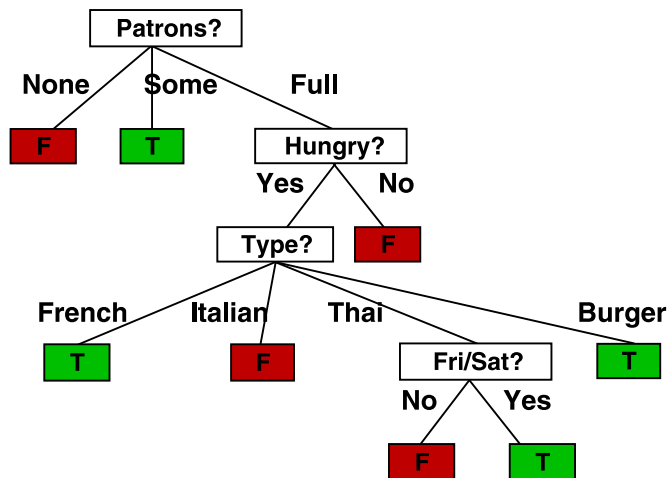
$$\sum_i \frac{p_i + n_i}{p + n} H(\langle p_i/(p_i + n_i), n_i/(p_i + n_i) \rangle)$$

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

$\Rightarrow$ choose the attribute that minimizes the remaining information needed

درخت تصمیم

مثال



درخت تصمیم یاد گرفته شده از روی ۱۲ مثال مسئله‌ی رستوران

ارزیابی کارایی

PERFORMANCE MEASUREMENT

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

۱) استفاده از قضیه‌های نظریه‌ی یادگیری محاسباتی/آماري

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

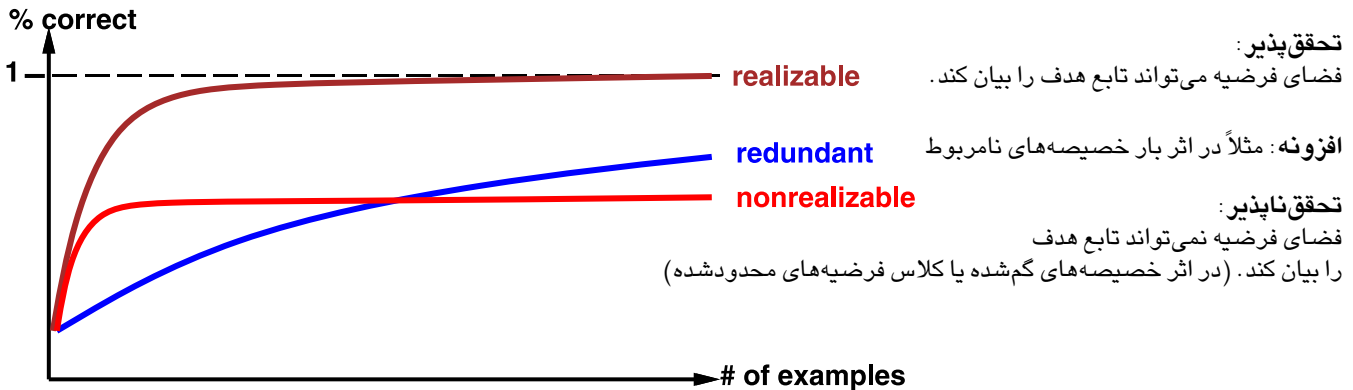
ارزیابی کارایی

منحنی یادگیری

PERFORMANCE MEASUREMENT

% پاسخ‌های درست روی مجموعه‌ی آزمایشی به صورت تابعی از اندازه‌ی مجموعه‌ی آموزشی

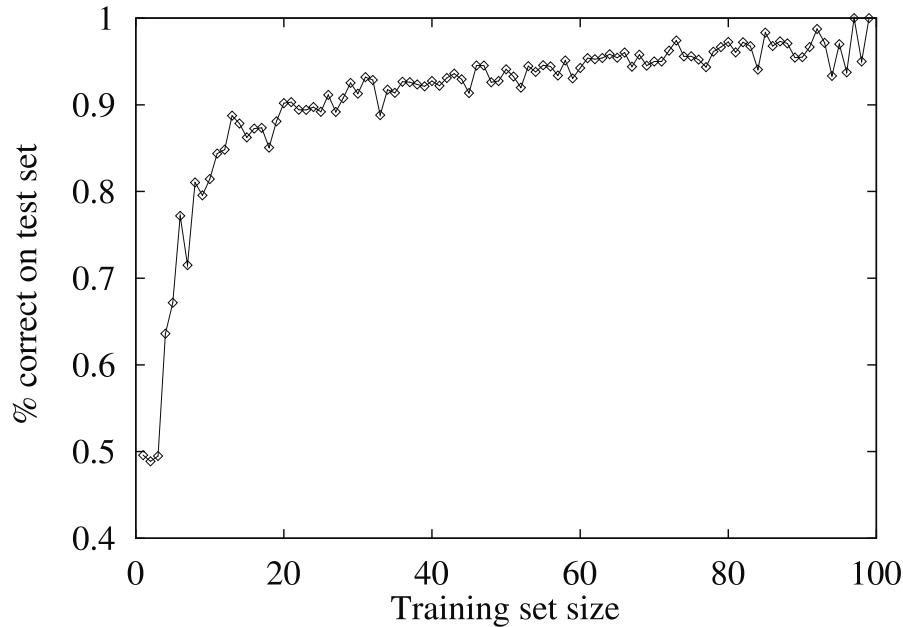
منحنی یادگیری
Learning Curve



ارزیابی کارایی

منحنی یادگیری: مثال

PERFORMANCE MEASUREMENT



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

۴

ارزیابی و انتخاب بهترین فرضیه

یادگیری از مثال‌ها

۵

نظریه‌ی
یادگیری

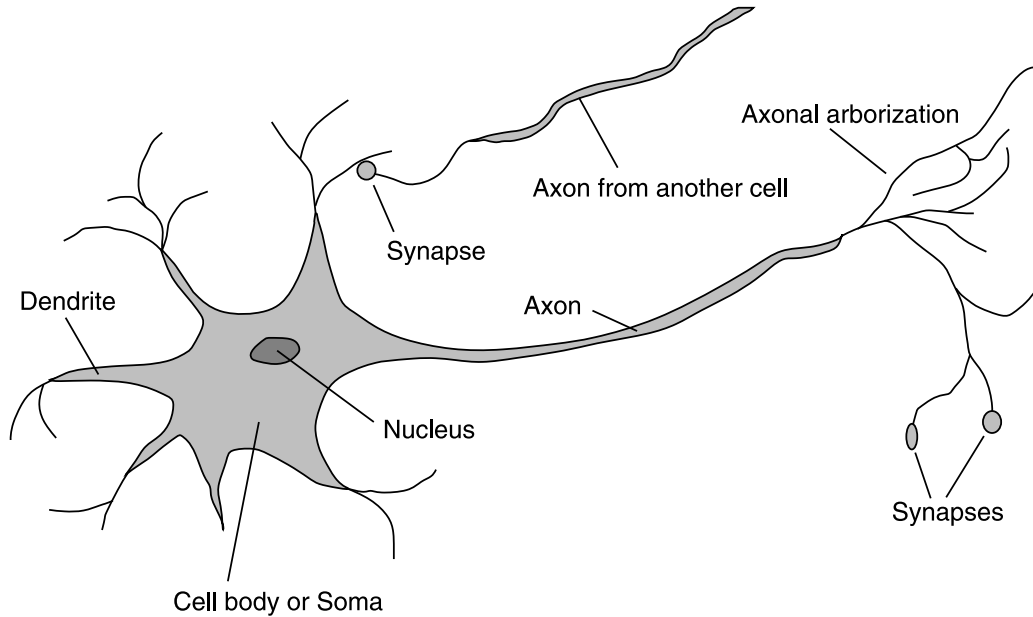
۶

رگرسیون
و
طبقه‌بندی
با
مدل‌های خطی

۷

شبکه‌های عصبی مصنوعی

مغز

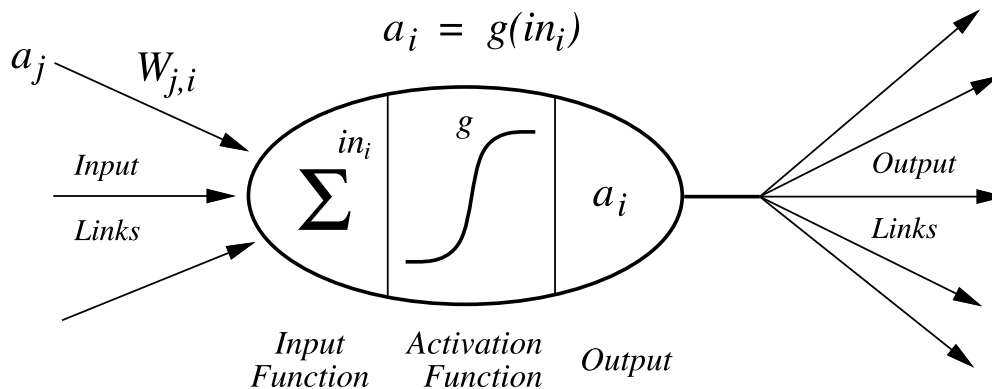
BRAIN

- 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 «مک‌لوچ-پیتز»

McCULLOCH-PITTS “UNIT”

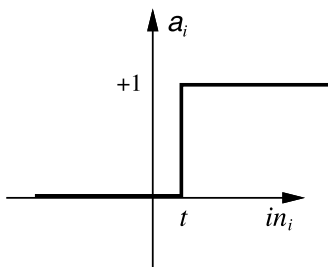


یک مدل بسیار ساده‌شده از نرون واقعی:

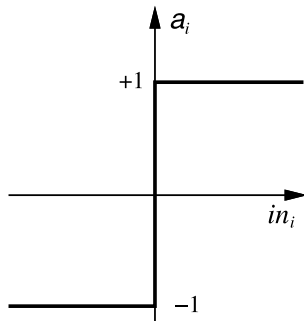
با هدف توسعه‌ی درک اینکه شبکه‌های متشکل از واحدهای ساده چه کاری می‌توانند انجام دهند.

شبکه‌های عصبی مصنوعی

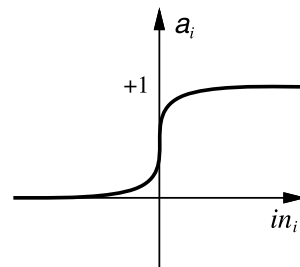
توابع فعال‌سازی

ACTIVATION FUNCTIONS

(a) Step function



(b) Sign function



(c) Sigmoid function

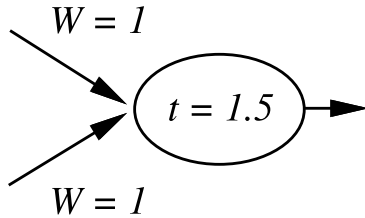
$$1/(1 + e^{-x})$$

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

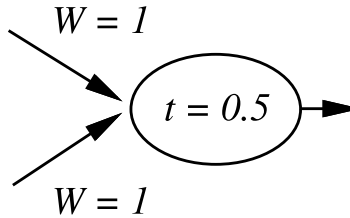
شبکه‌های عصبی مصنوعی

پیاده‌سازی توابع منطقی

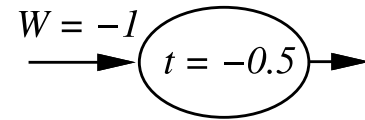
IMPLEMENTING LOGICAL FUNCTIONS



AND



OR



NOT

شبکه‌های عصبی مصنوعی

ساختارهای شبکه

NETWORK STRUCTURES

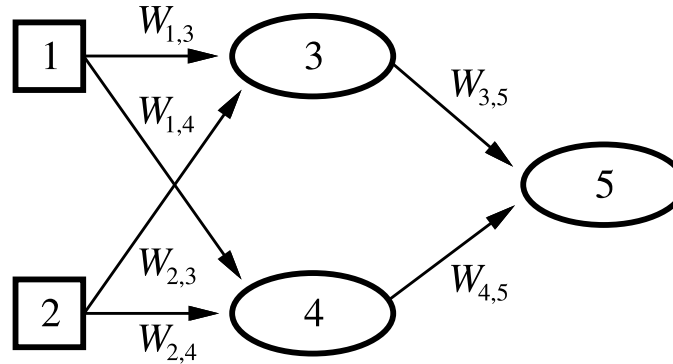
ساختارهای شبکه‌های عصبی مصنوعی			
شبکه‌های بازگشتی <i>Recurrent Networks</i>		شبکه‌های پیش‌خور <i>Feed-forward Networks</i>	
توابع دارای حالت داخلی (مانند فلیپ‌فلاپ) را پیاده‌سازی می‌کنند.		توابع بدون حالت داخلی را پیاده‌سازی می‌کنند.	
با فیدبک / چرخه‌های جهت‌دار با تاخیرها		بدون فیدبک	
ماشین‌های بولتزمن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

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

شبکه‌های عصبی مصنوعی

ساختارهای شبکه: شبکه‌ی پیش‌خور: مثال

FEED-FORWARD EXAMPLE



شبکه‌ی پیش‌خور = یک خانواده‌ی پارامتری شده از توابع غیرخطی

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

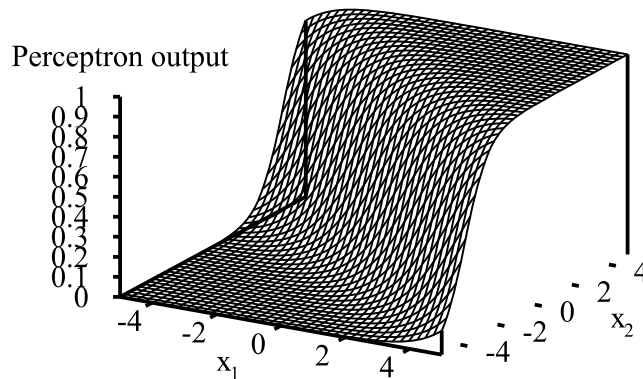
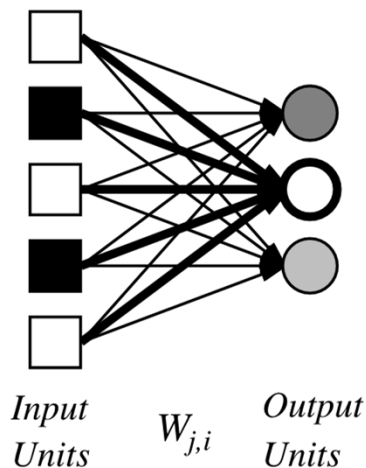
تنظیم وزن‌ها، تابع را تغییر می‌دهد: انجام یادگیری از این طریق

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

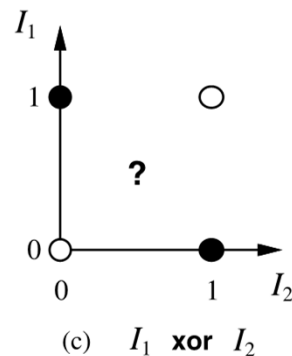
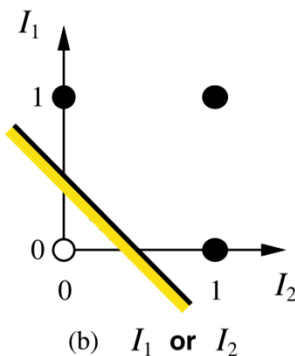
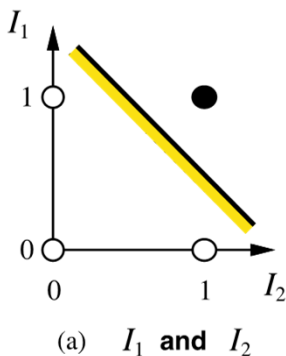
پرسپترون با تابع فعال‌سازی پله $g =$

پرسپترون یک جداساز خطی را در فضای ورودی بازنمایی می‌کند:

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

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

پرسپترون نمی‌تواند تابع XOR 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:

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

قاعده‌ی ساده برای به‌هنگام‌سازی وزن‌ها:

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

برای مثال:

خطای مثبت $\Leftarrow$ افزایش خروجی شبکه

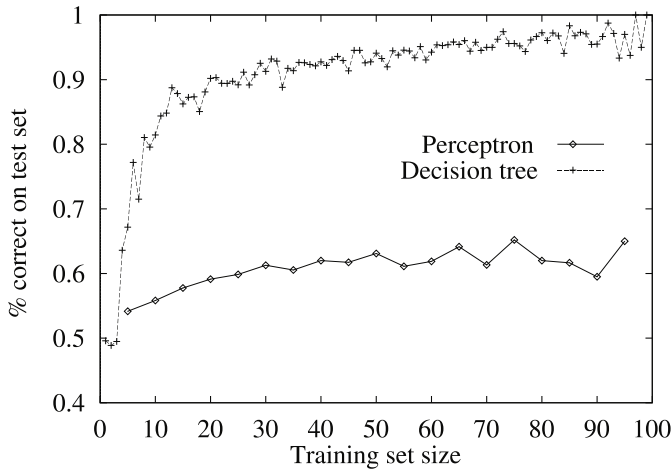
$\Leftarrow$ لزوم افزایش وزن‌ها روی ورودی‌های مثبت، کاهش وزن‌ها روی ورودی‌های منفی

شبکه‌های عصبی مصنوعی

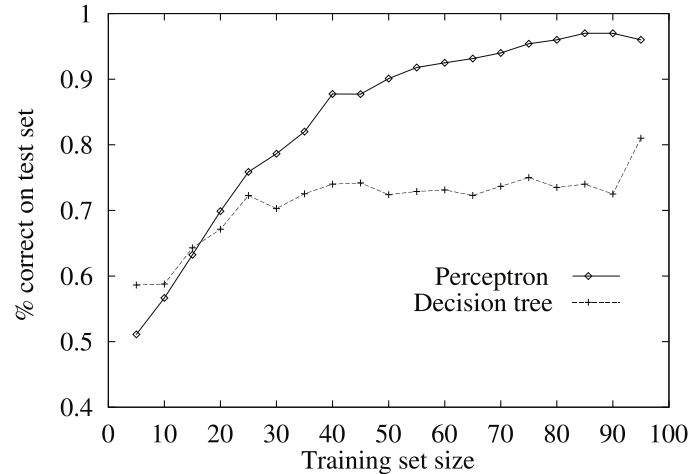
پرسپترون تک‌لایه: یادگیری پرسپترون: ویژگی‌ها

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.

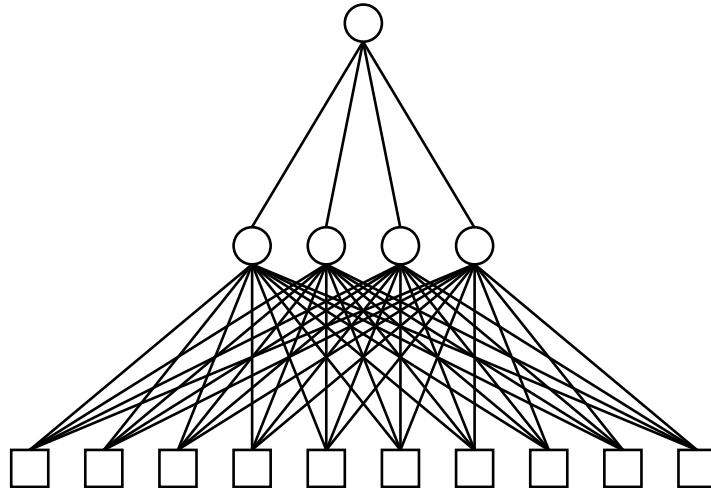
Output units O_i

$W_{j,i}$

Hidden units a_j

$W_{k,j}$

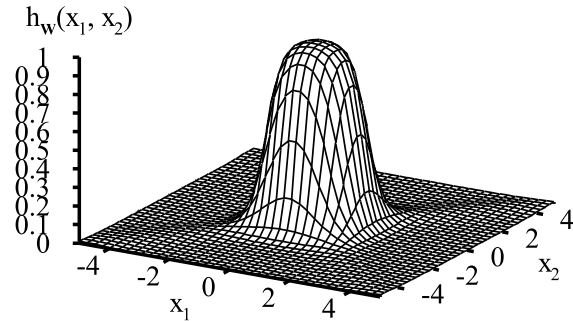
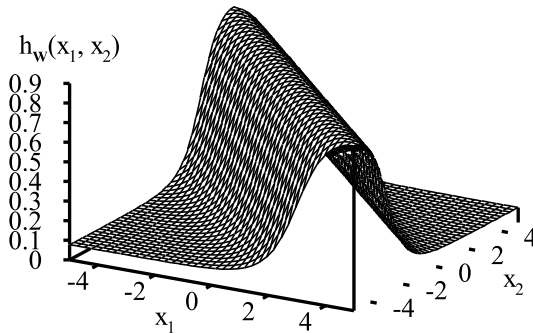
Input units I_k



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 $\Leftarrow$ ایجاد تیغه

ترکیب دو تیغه‌ی متعامد $\Leftarrow$ ایجاد یک برآمدگی

اضافه کردن برآمدگی‌ها با اندازه‌ها و مکان‌های گوناگون برای برآزش 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 نروساینیتیست‌ها 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

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

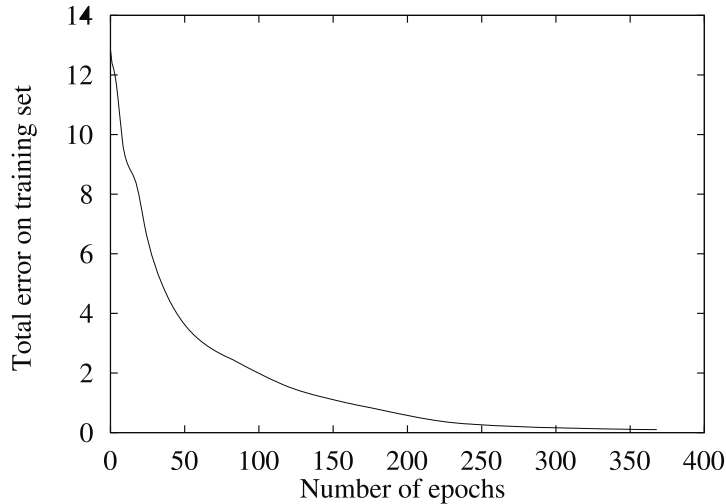
شبکه‌های عصبی مصنوعی

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)



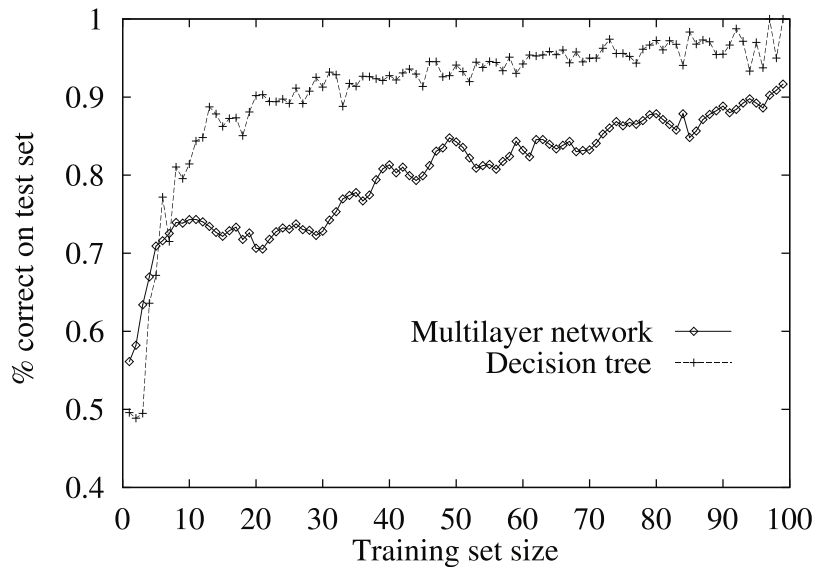
مشکل نوعی: همگرایی کند، می‌نیمم‌های محلی

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.

یادگیری از مثال‌ها



مدل‌های ناپارامتری

۹

ماشین‌های بردار پشتیبان

یادگیری از مثال‌ها



یادگیری جمعی

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

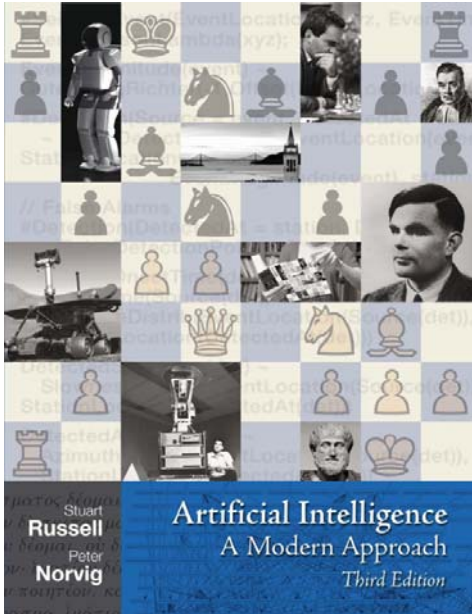
LeNet: 768–192–30–10 unit MLP = 0.9% error

Current best (kernel machines, vision algorithms) $\approx$ 0.6% error

یادگیری از مثال‌ها

۱۲

منابع،
مطالعه،
تکلیف



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.