Dan Roth
University of Illinois, Urbana-Champaign
danr@cs.uiuc.edu
http://L2R.cs.uiuc.edu/~danr
Context Sensitive Text Correction

Illinois’ bored of education. board

We took a walk it the park two. in, too

We fill it need no be this way feel, not

The amount of chairs in the room is… number

I’d like a peace of cake for desert piece, dessert
Disambiguation Problems

Middle Eastern ____ are known for their sweetness.

Task: Decide which of \{ deserts, desserts \} is more likely in the given context.

Ambiguity: modeled as confusion sets (class labels C)

C=\{ deserts, desserts \}
C=\{ Noun, Adj., Verb \}
C=\{ topic=Finance, topic=Computing \}
C=\{ NE=Person, NE=location \}
Disambiguation Problems

- Archetypical disambiguation problem

- Data is available (?)

- In principle, a solved problem
  Golding&Roth, Mangu&Brill,…

- But
  Many issues are involved in making an “in principle” solution a realistic one
Learning to Disambiguate

- **Given**
  - a confusion set $C = \{\text{deserts, desserts}\}$
  - sentence(s)
    Middle Eastern ____ are known for their sweetness

- **Map** into a feature based representation

- **Learn** a function $F_C$ that determines which of $C = \{\text{deserts, desserts}\}$ more likely in a given context.

- **Evaluate** the function on future $C$ sentences
Learning Approach: Representation

S= I don’t know whether to laugh or cry

[x          x                    x     x]

Consider words, pos tags, relative location in window
Generate binary features representing presence of:

a word/pos within window around target word
conjunctions of size 2, within window of size 3
words:
don’t within +/-3     know within +/-3     Verb at -1
to within +/-3      laugh within +/-3      Verb at +1

pos+words: Verb__to;     ____to Verb
Learning Approach: Representation

\[ S = I \text{ don’t know whether to laugh or cry} \]
Is represented as a set of its active features
\[ S = (\text{don’t at } -2, \text{ know within } +/-3, \ldots \text{ ____ to Verb, } \ldots) \]
Label= the confusion set element that occurs in the text

Hope: \[ S = I \text{ don’t care whether to laugh or cry} \]
has almost the same representation

This representation can be used by any propositional learning algorithm. (features, examples)

Previous works: TBL (Decision Lists) NB, SNoW, DT, ...
Notes on Representation

- There is a huge number of potential features (~$10^5$).
- Out of these – only a small number is actually active in each example.

- The representation can be significantly smaller if we list only features that are active in each example.

- Some algorithms can take this into account. Some cannot. (Later).
Notes on Representation (2)

- Formally:
  A feature is a characteristic function over sentences
  \[ \chi : S \rightarrow \{0,1\} \]

- When the number of features is fixed, the collection of all examples is
  \[ \{(\chi_1, \chi_2, \ldots, \chi_n)\} \equiv \{0,1\}^n \]

- When we do not want to fix the number of features (very large number, on-line algorithms, …) can work in the infinite attribute domain
  \[ \{(\chi_1, \chi_2, \ldots, \chi_n, \ldots)\} \equiv \{0,1\}^\infty \]
An Algorithm

Consider all training data \( S: \{(l, f, f, \ldots)\} \)

Represent as:

\[
S = \{(f, \#(l=0), \#(l=1))\} \text{ for all features}
\]

1. Choose best feature \( f^* \) (and the label it suggests)
2. \( S \leftarrow S \setminus \{\text{Examples labeled in (1)}\} \)
3. GoTo 1
An Algorithm: Hypothesis

If \( f_1 \) then label
Else, if \( f_2 \) then label
Else...
Else default label

A decision list

Issues: How well will this do?
We train on the training data, what about new data?
Generalization

I saw the girl in the park
The from needs to be completed
I maybe there tomorrow

- New sentences you have not seen before. Can you recognize and correct it this time?

- Intuitively, there are some regularities in the language, "identified" from previous examples, which can be utilized on future examples.

- Two technical ways to formalize this intuition
1: Direct Learning

- Model the problem of text correction as a problem of learning from examples.
- Goal: learn directly how to make predictions.

**PARADIGM**

- Look at many examples.
- Discover some regularities in the data.
- Use these to construct a prediction policy.
- A policy (a function, a predictor) needs to be specific.
  
  [it/in] rule: if *the* occurs after the target ⇒ *in*
  
  (in most cases, it won’t be that simple, though)
2: Generative Model

- Model the problem of text correction as that of *generating* correct sentences.
- Goal: learn a model of the language; use it to predict.

**PARADIGM**

- Learn a probability distribution over all sentences
  - In practice: make assumptions on the distribution’s type
- Use it to estimate which sentence is more likely.
  \[ \Pr(\text{I saw the girl it the park}) \not\sim \Pr(\text{I saw the girl in the park}) \]
- [In the same paradigm we sometimes learn a conditional probability distribution]
  - In practice: a decision policy depends on the assumptions
Example: Model of Language

- **Model 1:** There are 5 characters, A, B, C, D, E
- At any point can generate any of them, according to:
  
  \[
  P(A) = 0.3; \quad P(B) = 0.1; \quad P(C) = 0.2; \quad P(D) = 0.2; \quad P(E) = 0.1 \quad P(END) = 0.1
  \]

- Graphical representation: A sunflower model
- A sentence in the language: AAACCCDEABB.
- A less likely sentence: DEEEEEBBBBBBBBBBBB

- Given the model, can compute the probability of a sentence, and decide which is more likely.
Example: Model of Language

- **Model 2**: A probabilistic finite state model.

<table>
<thead>
<tr>
<th>Start:</th>
<th>$P_s(A) = 0.4; P_s(B) = 0.4; P_s(C) = 0.2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>From A:</td>
<td>$P_A(A) = 0.5; P_A(B) = 0.3; P_A(C) = 0.1; P_A(S) = 0.1$</td>
</tr>
<tr>
<td>From B:</td>
<td>$P_B(A) = 0.1; P_B(B) = 0.4; P_B(C) = 0.4; P_B(S) = 0.1$</td>
</tr>
<tr>
<td>From C:</td>
<td>$P_C(A) = 0.3; P_C(B) = 0.4; P_C(C) = 0.2; P_C(S) = 0.1$</td>
</tr>
</tbody>
</table>

- **Practical issues:**
  - What is the space over which we define the model? Characters? Words? Ideas?
  - How do we acquire the model? Estimation; Smoothing
Learning Paradigms: Comments

- The difference is not along probabilistic/deterministic or statistical/symbolic lines. Both paradigms can do both.

- The difference is in the basic assumptions underlying the paradigms, and why they work.
  - 1st: Distribution Free: uncover regularities in the past; hope they will be there in the future.
  - 2nd: Know the (type of) probabilistic model of the language (target phenomenon). Use it.

Direct Learning: Formalism

- **Goal:** discover some regularities from examples and generalize to previously unseen data.
  
  What are the examples we learn from?

- **Instance Space** $X$: The space of all examples

  $X = \{0,1\}^n$ or $\{0,1\}^\infty$

  How do we represent our hypothesis?

- **Hypothesis Space** $H$: Space of potential functions

  $h : X \rightarrow \{0,1\}$

- **Goal:** given training data $S \subseteq X$, find a good $h \in H$
Why Does Learning Work?

- Learning is impossible, unless....
- Outcome of Learning cannot be trusted, unless,....
- How can we quantify the expected generalization?
- Assume $h$ is good on the training data; what can be said on $h$'s performance on previously unseen data?

- These are some of the topics studied in Computational Learning Theory (COLT)

- notice: mode of interaction is also important
- More on all of these in CS346 (CS440 now?)
Learning is impossible, unless…

$y = f(x_1, x_2, x_3, x_4)$

**Given:**
- Training examples $(x, f(x))$
- of unknown function $f$

**Find:**
- A good approximation to $f$

<table>
<thead>
<tr>
<th>Example</th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Why Does Learning Work (2)?

- **Complete Ignorance**: There are $2^{16} = 56536$ possible functions over four input features.

- We can’t figure out which one is correct until we’ve seen every possible input-output pair.

- Even after seven examples we still have $2^9$ possibilities for $f$

- Is Learning Possible?

<table>
<thead>
<tr>
<th>Example</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>?</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>?</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>?</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
Hypothesis Space

- **Simple Rules**: There are only 16 simple conjunctive rules of the form
  \[ y = x_i \land x_j \land x_k \]

- Try to learn a function of this form that explains the data. (try it: there isn’t).

- **m-of-n rules**: There are 29 possible rules of the form
  "y = 1 if and only if at least m of the following n variables are 1"

  (try it, there is).
Bias

- Learning requires guessing a good, small hypothesis class.
- We can start with a very small class and enlarge it until it contains an hypothesis that fits the data.
  - (model selection)
- We could be wrong!
Can We Trust the Hypothesis?

- There is a hidden conjunction the learner is to learn:
  \[ f = x_2 \land x_3 \land x_4 \land x_5 \land x_{100} \]
- How many examples are needed to learn it? How?

- **Protocol:**
  - Some random source (e.g., Nature) provides training examples;
  - Teacher (Nature) provides the labels \((f(x))\)
- Not the only possible protocol (membership query; teaching)

\[
\begin{align*}
\langle 1,1,1,1,1,1,\ldots,1,1,1 \rangle & \quad \langle 1,1,1,1,0,0,0,\ldots,0,0,0 \rangle \\
\langle 1,1,1,1,1,0,\ldots,0,1,1 \rangle & \quad \langle 1,0,1,1,1,0,\ldots,0,1,1,1 \rangle \\
\langle 1,1,1,1,1,0,\ldots,0,0,1 \rangle & \quad \langle 1,0,1,0,0,0,\ldots,0,1,1 \rangle \\
\langle 1,1,1,1,1,1,\ldots,0,1 \rangle & \quad \langle 0,1,0,1,0,0,\ldots,0,1,1 \rangle \\
\langle 1,1,1,1,1,1,\ldots,0,1 \rangle & \quad \langle 0,1,0,1,0,0,\ldots,0,1,1 \rangle
\end{align*}
\]
Learning Conjunction

- **Algorithm**: Elimination
- Start with the set of all literals as candidates
- Eliminate a literal if not active (0) in a positive example

\[
\begin{align*}
<(1,1,1,1,1,1,\ldots,1,1), 1> &\quad f = x_1 \land x_2 \land x_3 \land x_4 \land x_5 \land x_{100} \\
<(1,1,1,0,0,0,\ldots,0,0), 0> &\quad \text{learned nothing} \\
<(1,1,1,1,1,0,\ldots,0,1,1), 1> &\quad f = x_1 \land x_2 \land x_3 \land x_4 \land x_5 \land x_{99} \land x_{100} \\
<(1,0,1,1,0,0,\ldots,0,0,1), 0> &\quad \text{learned nothing} \\
<(1,1,1,1,1,0,\ldots,0,0,1), 1> &\quad f = x_1 \land x_2 \land x_3 \land x_4 \land x_5 \land x_{100} \\
<(1,0,1,0,0,0,\ldots,0,1,1), 0> &\quad \text{learned nothing} \\
<(1,1,1,1,1,1,\ldots,0,1,1), 1> &\quad \text{Final: } f = x_1 \land x_2 \land x_3 \land x_4 \land x_5 \land x_{100} \\
<(0,1,0,1,0,0,\ldots,0,1,1), 0> &
\end{align*}
\]
Prototypical Learning Scenario

- Instance Space: $X$
- Hypothesis Space: $H$ (set of possible hypotheses)
- Training instances $S$:
  - positive and negative examples of the target $f$
  - $S$: sampled according to a fixed, unknown, probability distribution $D$ over $X$
- Determine: A hypothesis $h \in H$ such that
  
  \[
  h(x) = f(x) \quad \text{for all } x \in S
  \]
  
  \[
  h(x) = f(x) \quad \text{for all } x \in X
  \]

- Evaluated on future instances sampled according to $D$
  
  \[
  f = x_1 \land x_2 \land x_3 \land x_4 \land x_5 \land x_{100}
  \]
PAC Learning: Intuition

- Have seen many examples (drawn according to $D$)
- Since in all the positive examples $x_1$ was active, it is likely to be active in future positive examples
- If not, in any case, in $D$, $x_1$ is active only in relatively few examples, so our error will be small.

$$\text{Error}_D = \Pr_{x \in D} [f(x) \neq h(x)]$$

Error can be bounded via Chernoff bounds

A distribution free notion!
Claim: The probability that there exists a hypothesis $h \in H$ that:
1. is consistent with $m$ examples and
2. satisfies $\text{err}(h) > \varepsilon$
is less than $|H|(1 - \varepsilon)^m$.

Equivalently:

- For any distribution $D$ governing the IID generation of training and test instances, for all $h \in H$, for all $0 < \varepsilon, \delta < 1$, if
  $$m > \frac{\ln(|H|) + \ln(1/\delta)}{\varepsilon}$$
- Then, with probability at least $1 - \delta$ (over the choice of the training set of size $m$),
  $$\text{err}(h) < \varepsilon$$
Generalization for Consistent Learners

Claim: The probability that there exists a hypothesis $h \in H$ that:
   (1) is consistent with $m$ examples and
   (2) satisfies $\text{err}(h) > \varepsilon$
   is less than $|H|(1 - \varepsilon)^m$

Proof: Let $h$ be such a bad hypothesis.
   - The probability that $h$ is consistent with one example of $f$ is
     $P_{x \in D} [f(x) = h(x)] < (1 - \varepsilon)$
   - Since the $m$ examples are drawn independently of each other, the probability that $h$ is consistent with $m$ examples is less than $(1 - \varepsilon)^m$
   - The probability that some hypothesis in $H$ is consistent with $m$ examples is less than $|H|(1 - \varepsilon)^m$
Generalization for Consistent Learners

- We want this probability to be smaller than \( \delta \), that is:
  \[ |H|(1- \varepsilon)^m < \delta \]
- And with \((1- x < e^{-x})\)
  \[ \ln(|H|) - m \varepsilon < \ln(\delta) \]

What kind of hypothesis spaces do we want? Large? Small?
To guarantee consistency we need \( H \supseteq C \). But do we want the smallest \( H \) possible?

- For any distribution \( D \) governing the IID generation of training and test instances, for all \( h \in H \), for all \( 0 < \varepsilon, \delta < 1 \), if
  \[ m > \{ \ln(|H|) + \ln(1/ \delta) \}/\varepsilon \]
- Then, with probability at least \( 1-\delta \) (over the choice of the training set of size \( m \)),
  \[ \text{err}(h) < \varepsilon \]
Generalization (Agnostic Learners)

- In general: we try to learn a concept \( f \) using hypotheses in \( H \), but \( f \notin H \)

- Our goal should be to find a hypothesis \( h \in H \), with a small training error:

\[
\text{Err}_{\text{TR}}(h) = P_{x \in S}[f(x) \neq h(x)]
\]

- We want a guarantee that a hypothesis with a small training error will have a good accuracy on unseen examples

\[
\text{Err}_{\text{D}}(h) = P_{x \in D}[f(x) \neq h(x)]
\]

- Hoeffding bounds characterize the deviation between the true probability of an event and its observed frequency over \( m \) independent trials.

\[
\Pr(p > E(p) + \varepsilon) < \exp\{-2m \varepsilon^2\}
\]

\((p \text{ is the underlying probability of the binary variable being 1})\)
Generalization (Agnostic Learners)

- Therefore, the probability that an element $h \in H$ will have training error which is off by more than $\varepsilon$ can be bounded as follows:

$$\Pr(\text{Err}_D(h) > \text{Err}_{TR}(h) + \varepsilon) < \exp\{-2m \varepsilon^2\}$$

- As in the consistent case: use union bound to get a uniform bound on all $H$; to get $|H| \exp\{-2m \varepsilon^2\} < \delta$ we have the following generalization bound: a bound on how much will the true error deviate from the observed error.

- For any distribution $D$ generating training and test instance, with probability at least $1 - \delta$ over the choice of the training set of size $m$, (drawn IID), for all $h \in H$

$$\text{Err}_D(h) < \text{Err}_{TR}(h) + \sqrt{\frac{\log|H| + \log(1/\delta)}{2m}}$$
Summary: Generalization

- Learnability depends on the size of the hypothesis space.

- In the case of a finite hypothesis space:

  \[
  \text{Err}_D(h) < \text{Err}_{TR}(h) + \sqrt{\frac{\log|H| + \log(1/\delta)}{2m}}
  \]

- In the case of an infinite hypothesis space

  \[
  \text{Err}_D(h) < \text{Err}_{TR}(h) + \sqrt{\frac{k\text{VC}|H| + \log(1/\delta)}{2m}}
  \]

- Where \(\text{VC}(H)\) is the Vapnik-Chernvonenkis of the hypothesis class, a combinatorial measure of its complexity.
Learning Theory: Summary (1)

- Labeled observations
  \[ S = \{ (x, l) \}_{i=1}^{m} \]
  sampled according to a distribution \( D \) on \( X \times \{0,1\} \)

- Goal: to compute a hypothesis \( h \in H \) that performs well on future, unseen observations.

- Assumption: test examples are also sampled according to \( D \) (label is not observed)
Learning Theory: Summary (2) [Why does it work?]

- Look for \( h \in H \) that minimizes the true error

\[
\text{Err}_D (h) = \Pr_{(x, l) \in D} [h(x) \neq l]
\]

- All we get to see is the empirical error

\[
\text{Err}_S (h) = \frac{|\{x \in S \mid h(x) \neq l\}|}{|S|}
\]

- Basic theorem: With probability at least \((1-\delta)\)

\[
\text{Err}_D (h) < \text{Err}_S (h) + \sqrt{\frac{\text{VC}(H) + \ln(1/\delta)}{m}}
\]
Practical Lesson

- Use Hypothesis Space with small expressivity
- E.g. prefer to use a function that is linear in the feature space, over higher order functions
  \[ f(x) = \sum_i c_i \chi_i \]

- VC dimension of a linear function of dimension N: is N+1

- Sparsity: If there are a maximum of k active in each example then VC dimension is k+1

- Algorithmic issues: There are good algorithms for linear function; learning higher order functions is computationally hard.
Advances in Theory of Generalization

- VC dimension based bounds are unrealistic.
- The value is mostly in providing quantitative understanding of “why learning works” and what are the important complexity parameters.

- In recent years, this understanding has helped both to
  - drive new algorithms
  - Develop new methods that can actually provide somewhat realistic generalization bounds.

- PAC-Bayes Methods (McAlister, McAlister&Langford)
- Random Projection Methods (Garg, Har-Peled, Roth)
- This method can be shown to have some algorithmic implications.
2: Generative Model

- Model the problem of text correction as that of generating correct sentences.
- Goal: learn a model of the language; use it to predict.

**PARADIGM**

- Learn a probability distribution over all sentences
  - In practice: make assumptions on the distribution’s type
- Use it to estimate which sentence is more likely.
  \[ \Pr(\text{I saw the girl it the park}) \gg \Pr(\text{I saw the girl in the park}) \]
  [In the same paradigm we sometimes learn a conditional probability distribution]
  - In practice: a decision policy depends on the assumptions
Before: Error Driven Learning

- Consider a distribution \( D \) over space \( X \times Y \)
- \( X \) - the instance space; \( Y \) - set of labels. (e.g. +/-1)

- Given a sample \( \{(x,y)\}_{1}^{m} \), and a loss function \( L(x,y) \)
  Find \( h \in H \) that minimizes
  \[
  \Sigma_{i=1,m} L(h(x_{i}),y_{i})
  \]

- \( L \) can be:
  - \( L(a,b)=1, \ a \neq b, \ o/w \ L(a,b) = 0 \) \ (0-1 loss)
  - \( L(a,b)=(a-b)^2 \) \ (L₂)
  - \( L(a,b)=\exp\{-y_{i}h(x_{i})\} \)

- Find an algorithm that minimizes average loss; then, we know that things will be okay (as a function of \( H \)).
Basics of Bayesian Learning

- **Goal:** find the best hypothesis from some space $H$ of hypotheses, given the observed data $D$.

- Define **best** to be: most **probable hypothesis** in $H$.

- In order to do that, we need to assume a probability distribution over the class $H$.

- In addition, we need to know something about the relation between the data observed and the hypotheses (E.g., a coin problem.)

  - As we will see, we will be Bayesian about other things, e.g., the parameters of the model.
Basics of Bayesian Learning

- $P(h)$ - the prior probability of a hypothesis $h$
  Reflects background knowledge; before data is observed. If no information - uniform distribution.

- $P(D)$ - The probability that *this sample* of the Data is observed. (No knowledge of the hypothesis)

- $P(D|h)$: The probability of observing the sample $D$, given that the hypothesis $h$ holds

- $P(h|D)$: The posterior probability of $h$. The probability $h$ holds, given that $D$ has been observed.
Bayes Theorem

\[ P(h \mid D) = \frac{P(D \mid h) P(h)}{P(D)} \]

- \( P(h \mid D) \) increases with \( P(h) \) and with \( P(D \mid h) \)
- \( P(h \mid D) \) decreases with \( P(D) \)
Learning Scenario

\[ P(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)} \]

- The learner considers a set of candidate hypotheses \( H \) (models), and attempts to find the most probable one \( h \in H \), given the observed data.

- Such maximally probable hypothesis is called maximum a posteriori hypothesis (MAP); Bayes theorem is used to compute it:

\[
\begin{align*}
    h_{\text{MAP}} &= \arg\max_{h \in H} P(h \mid D) = \arg\max_{h \in H} P(D \mid h)P(h) \\
                     &\quad \frac{P(h)}{P(D)} \\
    &= \arg\max_{h \in H} P(D \mid h)P(h)
\end{align*}
\]
Learning Scenario (2)

\[ h_{\text{MAP}} = \arg\max_{h \in H} P(h \mid D) = \arg\max_{h \in H} P(D \mid h)P(h) \]

- We may assume that a priori, hypotheses are equally probable
  \[ P(h_i) = P(h_j), \forall h_i, h_j \in H \]
- We get the Maximum Likelihood hypothesis:
  \[ h_{\text{ML}} = \arg\max_{h \in H} P(D \mid h) \]
- Here we just look for the hypothesis that best explains the data
Bayes Optimal Classifier

- How should we use the general formalism?
- What should \( H \) be?

- \( H \) can be a collection of functions. Given the training data, choose an optimal function. Then, given new data, evaluate the selected function on it.
- \( H \) can be a collection of possible predictions. Given the data, try to directly choose the optimal prediction.
- \( H \) can be a collection of (conditional) probability distributions.

- Could be different!