Computational Learning Theory: A Brief Introduction

Machine Learning
Spring 2021

The slides are based on Vivek Srikumar
This section

1. General framework and key concepts: empirical error and generalization error, goal

2. Define the PAC learnability

3. Make formal connections to the principle of Occam’s razor
Checkpoint: The bigger picture

• Supervised learning: instances, labels, and hypotheses

![Diagram of supervised learning process]

- Labeled data
- Learning algorithm
- Hypothesis/Model h
- New example
- Prediction
Checkpoint: The bigger picture

• Supervised learning: instances, labels, and hypotheses

• Specific learners
  – Decision trees
  – Perceptron
  – LMS
  – ...

Learning algorithm
Labeled data
Hypothesis/Model h
New example
\( h \)
Prediction
Checkpoint: The bigger picture

- Supervised learning: instances, labels, and hypotheses

- Specific learners
  - Decision trees
  - Perceptron
  - LMS
  - ...

- General ML ideas
  - Features as high dimensional vectors
  - Overfitting
  - ...

[Diagram showing the process of learning from labeled data to predictions]

Learning algorithm
Labeled data
Hypothesis/Model h
New example h Prediction
1. Computational learning theory

• A model
  – Train on a fixed training set
  – Then deploy it in the wild

• How well will your learned model perform on future instances?
Supervised learning setup

- **Instance Space**: $X$, the set of examples
Supervised learning setup

- **Instance Space**: $X$, the set of examples
- **Concept Space**: $C$, the type/family of target functions: $f \in C$ is the hidden target function
  - Eg: all Boolean functions; all n-dimensional linear functions, ...

- **Hypothesis Space**: $H$, the set of possible hypotheses
  - This is the set that the learning algorithm explores
- **Training instances**: $S \subseteq \{-1,1\}$: positive and negative examples of the target concept. ($S$ is a finite subset of $X$)
- What we want: A hypothesis $h \in H$ such that $h(x) = f(x)$
  - A hypothesis $h \in H$ such that $h(x) = f(x)$ for all $x \in S$?
  - A hypothesis $h \in H$ such that $h(x) = f(x)$ for all $x \in X$?
Supervised learning setup

- **Instance Space**: $X$, the set of examples
- **Concept Space**: $C$, the type/family of target functions: $f \in C$ is the hidden target function
  - Eg: all Boolean functions; all n-dimensional linear functions, ...
- **Hypothesis Space**: $H$, the set of possible hypotheses
  - This is the set that the learning algorithm explores
Supervised learning setup

- **Instance Space**: $X$, the set of examples
- **Concept Space**: $C$, the type/family of target functions: $f \in C$ is the hidden target function
  - Eg: all Boolean functions; all $n$-dimensional linear functions, ...
- **Hypothesis Space**: $H$, the set of possible hypotheses
  - This is the set that the learning algorithm explores
- **Training instances**: $S \times \{-1,1\}$: positive and negative examples of the target concept. ($S$ is a finite subset of $X$)
  
  $< x_1, f(x_1) >, < x_2, f(x_2) >, ..., < x_n, f(x_n) >$
Supervised learning setup

- **Instance Space**: $X$, the set of examples
- **Concept Space**: $C$, the type/family of target functions: $f \in C$ is the hidden target function
  - E.g.: all Boolean functions; all $n$-dimensional linear functions, ...
- **Hypothesis Space**: $H$, the set of possible hypotheses
  - This is the set that the learning algorithm explores
- **Training instances**: $S \times \{-1, 1\}$: positive and negative examples of the target concept. ($S$ is a finite subset of $X$)
  $$< x_1, f(x_1) >, < x_2, f(x_2) >, ..., < x_n, f(x_n) >$$
- **What we want**: A hypothesis $h \in H$ such that $h(x) = f(x)$
  - A hypothesis $h \in H$ such that $h(x) = f(x)$ for all $x \in S$?
  - A hypothesis $h \in H$ such that $h(x) = f(x)$ for all $x \in X$?
Supervised learning setup

- **Instance Space**: $X$, the set of examples
- **Concept Space**: $C$, the type/family of target functions: $f \in C$ is the hidden target function
  - Eg: all Boolean functions; all $n$-dimensional linear functions, ...
- **Hypothesis Space**: $H$, the set of possible hypotheses (or candidates)
  - This is the set that the learning algorithm explores
- **Training instances**: $S \times \{-1, 1\}$: positive and negative examples of the target concept. ($S$ is a finite subset of $X$)
  - *Training instances are generated by a fixed unknown probability distribution $D$ over $X$*
- **What we want**: A hypothesis $h \in H$ such that $h(x) = f(x)$
  - Evaluate $h$ on subsequent examples $x \in X$ drawn according to $D
Generalization Error of a hypothesis

Definition

Given a distribution $D$ over examples, the error of a hypothesis $h$ with respect to a target concept $f$ is

$$\text{err}_D(h) = \Pr_{x \sim D}[h(x) \neq f(x)]$$
Generalization Error of a hypothesis

**Definition**

Given a distribution $D$ over examples, the *error* of a hypothesis $h$ with respect to a target concept $f$ is

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

Instance space $X$
Generalization Error of a hypothesis

Definition

Given a distribution $D$ over examples, the error of a hypothesis $h$ with respect to a target concept $f$ is

$$\text{err}_D(h) = \Pr_{x \sim D}[h(x) \neq f(x)]$$
Generalization Error of a hypothesis

Definition

Given a distribution $D$ over examples, the \textit{error} of a hypothesis $h$ with respect to a target concept $f$ is

$$\text{err}_D(h) = \Pr_{x \sim D}[h(x) \neq f(x)]$$
Generalization Error of a hypothesis

**Definition**

Given a distribution $D$ over examples, the *error* of a hypothesis $h$ with respect to a target concept $f$ is

$$\text{err}_D(h) = \Pr_{x \sim D}[h(x) \neq f(x)]$$
Empirical error

Contrast true error against the empirical error

For a target concept $f$, the empirical error of a hypothesis $h$ is defined *for a training set $S$ as the fraction of examples $x$ in $S$ for which the functions $f$ and $h$ disagree*. That is, $h(x) \neq f(x)$

Denoted by $\text{err}_S(h)$

**Overfitting**: When the empirical error on the training set $\text{err}_S(h)$ is substantially lower than $\text{err}_D(h)$
Formulating the theory of prediction

In the general case, we have

- $X$: instance space, $Y$: output space = \{+1, -1\}
- $D$: an unknown distribution over $X$
- $f$: an unknown target function $X \to Y$, taken from a concept class $C$
- $h$: a hypothesis function $X \to Y$ that the learning algorithm selects from a hypothesis class $H$
- $S$: a set of $m$ training examples drawn from $D$, labeled with $f$
- $\text{err}_D(h)$: The true error of any hypothesis $h$
- $\text{err}_S(h)$: The empirical error or training error or observed error of $h$
Theoretical questions

• Can we describe or bound the generalization error ($\text{err}_D$) given the empirical error ($\text{err}_S$)?

• Is a concept class C learnable? (what does it mean for “learnable”)?

• How many examples does an algorithm need to guarantee good performance? (sample complexity)

• How much cost does an algorithm take to learn class C? (computational complexity)?
Requirements of Learning

• Cannot expect a learner to learn a concept **exactly**
  – There will generally be multiple concepts consistent with the available data (which represent a small fraction of the available instance space)
  – Unseen examples could *potentially* have any label
  – We “agree” to misclassify *uncommon* examples that do not show up in the training set
Requirements of Learning

• Cannot expect a learner to learn a concept exactly
  – There will generally be multiple concepts consistent with the available data (which represent a small fraction of the available instance space)
  – Unseen examples could potentially have any label
  – We “agree” to misclassify uncommon examples that do not show up in the training set

• Cannot always expect to learn a close approximation to the target concept
Requirements of Learning

• Cannot expect a learner to learn a concept exactly
  – There will generally be multiple concepts consistent with the available data (which represent a small fraction of the available instance space)
  – Unseen examples could potentially have any label
  – We “agree” to misclassify uncommon examples that do not show up in the training set

• Cannot always expect to learn a close approximation to the target concept
  – Sometimes (only in rare learning situations, we hope) the training set will not be representative (will contain uncommon examples)
Requirements of Learning

• Cannot expect a learner to learn a concept exactly
  – There will generally be multiple concepts consistent with the available data (which represent a small fraction of the available instance space)
  – Unseen examples could potentially have any label (data distribution is unknown!)
  – We “agree” to misclassify uncommon examples that do not show up in the training set

• Cannot always expect to learn a close approximation to the target concept
  – Sometimes (only in rare learning situations, we hope) the training set will not be representative (will contain uncommon examples)

• The only realistic expectation of a good learner is that with high probability it will learn a close approximation to the target concept
Probably approximately correctness (PAC)

• The only realistic expectation of a good learner is that with high probability it will learn a close approximation to the target concept.
Probably approximately correctness (PAC)

• The only realistic expectation of a good learner is that with high probability it will learn a close approximation to the target concept

• In Probably Approximately Correct (PAC) learning, one requires that
  – given small parameters $\epsilon$ and $\delta$,  
  – With probability at least $1 - \delta$, a learner produces a hypothesis with error at most $\epsilon$
**Probably approximately correctness (PAC)**

- The only realistic expectation of a good learner is that **with high probability** it will learn a **close approximation** to the target concept.

- In **Probably Approximately Correct (PAC) learning**, one requires that:
  - given small parameters $\epsilon$ and $\delta$,  
  - With probability at least $1 - \delta$, a learner produces a hypothesis with error at most $\epsilon$.

- The only reason we can hope for this is the **consistent distribution assumption**.
2. PAC Learnability

Consider a concept class C defined over an instance space X (containing instances of length n), and a learner L using a hypothesis space H.
PAC Learnability

Consider a concept class $C$ defined over an instance space $X$ (containing instances of length $n$), and a learner $L$ using a hypothesis space $H$

The concept class $C$ is **PAC learnable** by $L$ using $H$ if

$$\Pr_{D} \left[ \text{Err}_{D}(h) \leq \epsilon \right] \geq 1 - \delta$$
PAC Learnability

Consider a concept class \( C \) defined over an instance space \( X \) (containing instances of length \( n \)), and a learner \( L \) using a hypothesis space \( H \).

The concept class \( C \) is **PAC learnable** by \( L \) using \( H \) if

- for all \( f \in C \),
- for all distribution \( D \) over \( X \), and fixed \( 0 < \varepsilon, \delta < 1 \),
PAC Learnability

Consider a concept class $C$ defined over an instance space $X$ (containing instances of length $n$), and a learner $L$ using a hypothesis space $H$.

The concept class $C$ is **PAC learnable** by $L$ using $H$ if for all $f \in C$, for all distribution $D$ over $X$, and fixed $0 < \epsilon, \delta < 1$, given $m$ examples sampled independently according to $D$, the algorithm $L$ produces, with probability at least $(1 - \delta)$, a hypothesis $h \in H$ that has error at most $\epsilon$, where $m$ is *polynomial* in $\frac{1}{\epsilon}, \frac{1}{\delta}, n$ and $\text{size}(H)$.
PAC Learnability

Consider a concept class $C$ defined over an instance space $X$ (containing instances of length $n$), and a learner $L$ using a hypothesis space $H$

The concept class $C$ is **PAC learnable** by $L$ using $H$ if for all $f \in C$, for all distribution $D$ over $X$, and fixed $0 < \varepsilon, \delta < 1$, given $m$ examples sampled independently according to $D$, the algorithm $L$ produces, with probability at least $(1 - \delta)$, a hypothesis $h \in H$ that has error at most $\varepsilon$, where $m$ is *polynomial* in $1/\varepsilon$, $1/\delta$, $n$ and size($H$)

recall that $\text{Err}_D(h) = \Pr_D[f(x) \neq h(x)]$
PAC Learnability

Consider a concept class $C$ defined over an instance space $X$ (containing instances of length $n$), and a learner $L$ using a hypothesis space $H$.

The concept class $C$ is **PAC learnable** by $L$ using $H$ if

for all $f \in C$,

for all distribution $D$ over $X$, and fixed $0 < \varepsilon, \delta < 1$,

given $m$ examples sampled independently according to $D$, the algorithm $L$ produces, with probability at least $(1 - \delta)$, a hypothesis $h \in H$ that has error at most $\varepsilon$,

where $m$ is *polynomial* in $\frac{1}{\varepsilon}, \frac{1}{\delta}, n$ and $\text{size}(H)$

recall that $\text{Err}_D(h) = \Pr_D[f(x) \neq h(x)]$

The concept class $C$ is **efficiently learnable** if $L$ can produce the hypothesis in time that is polynomial in $\frac{1}{\varepsilon}, \frac{1}{\delta}, n$ and $\text{size}(H)$
PAC Learnability

- We impose two requirements
  - Polynomial sample complexity (information theoretic constraint)
    - Is there enough information in the sample to distinguish a hypothesis $h$ that approximate $f$?
  - Polynomial time complexity (computational complexity)
    - Is there an efficient algorithm that can process the sample and produce a good hypothesis $h$?

To be PAC learnable, there must be a hypothesis $h \in H$ with arbitrary small error for every $f \in C$. We assume $H \subseteq C$. (Properly PAC learnable if $H = C$)

Worst Case definition: the algorithm must meet its accuracy
- for every distribution (The distribution free assumption)
- for every target function $f$ in the class $C$
PAC Learnability

• We impose two requirements
• Polynomial \textit{sample complexity} (information theoretic constraint)
  – Is there enough information in the sample to distinguish a hypothesis \( h \) that approximate \( f \)?
PAC Learnability

- We impose two requirements
- Polynomial \textit{sample complexity} (information theoretic constraint)
  - Is there enough information in the sample to distinguish a hypothesis $h$ that approximate $f$?
- Polynomial \textit{time complexity} (computational complexity)
  - Is there an efficient algorithm that can process the sample and produce a good hypothesis $h$?
PAC Learnability

• We impose two requirements
  • Polynomial *sample complexity* (information theoretic constraint)
    – Is there enough information in the sample to distinguish a hypothesis $h$ that approximate $f$?
  • Polynomial *time complexity* (computational complexity)
    – Is there an efficient algorithm that can process the sample and produce a good hypothesis $h$?

To be PAC learnable, there must be a hypothesis $h \in H$ with arbitrary small error for every $f \in C$. We assume $H \supseteq C$. (*Properly* PAC learnable if $H=C$)
PAC Learnability

• We impose two requirements
  • Polynomial *sample complexity* (information theoretic constraint)
    – Is there enough information in the sample to distinguish a hypothesis $h$ that approximate $f$?
  • Polynomial *time complexity* (computational complexity)
    – Is there an efficient algorithm that can process the sample and produce a good hypothesis $h$?

To be PAC learnable, there must be a hypothesis $h \in H$ with arbitrary small error for every $f \in C$. We assume $H \supseteq C$. (*Properly* PAC learnable if $H=C$)

**Worst Case definition**: the algorithm must meet its accuracy
  – for every distribution (The distribution free assumption)
  – for every target function $f$ in the class $C$
3. Occam’s Razor Justification

Named after William of Occam
– AD 1300s

*Prefer simpler explanations over more complex ones*

“Numquam ponenda est pluralitas sine necessitate”

(Never posit plurality without necessity.)

Historically, a widely prevalent idea across different schools of philosophy
Towards formalizing Occam’s Razor

*Claim*: The probability that there is a hypothesis $h \in H$ that:

1. Is **Consistent** with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| \ (1 - \epsilon)^m$
Towards formalizing Occam’s Razor

(Assuming consistency)

Claim: The probability that there is a hypothesis $h \in H$ that:

1. Is Consistent with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$
   is less than $|H| (1 - \epsilon)^m$
Towards formalizing Occam’s Razor

(*Assuming consistency*)

**Claim**: The probability that there is a hypothesis \( h \in H \) that:

1. Is **Consistent** with \( m \) examples, and
2. Has \( \text{err}_D(h) > \epsilon \)

That is, **consistent yet bad**

is less than \( |H| (1 - \epsilon)^m \)}
Towards formalizing Occam’s Razor

(Assuming consistency)

**Claim**: The probability that there is a hypothesis \( h \in H \) that:

1. Is **Consistent** with \( m \) examples, and
2. Has \( err_D(h) > \epsilon \)

is less than \( |H| (1 - \epsilon)^m \)

**Proof**: Let \( h \) be such a bad hypothesis that has an error > \( \epsilon \)
Towards formalizing Occam’s Razor

(Assuming consistency)

Claim: The probability that there is a hypothesis \( h \in H \) that:
1. Is Consistent with \( m \) examples, and
2. Has \( \text{err}_D(h) > \varepsilon \)
   is less than \( |H| (1 - \varepsilon)^m \)

Proof: Let \( h \) be such a bad hypothesis that has an error > \( \varepsilon \)
Probability that \( h \) is consistent with one example is \( \Pr[f(x) = h(x)] < 1 - \varepsilon \)
Towards formalizing Occam’s Razor

( Assuming consistency )

**Claim:** The probability that there is a hypothesis \( h \in H \) that:
1. Is **Consistent** with \( m \) examples, and
2. Has \( \text{err}_D(h) > \epsilon \)
   is less than \( |H| (1 - \epsilon)^m \)

**Proof:** Let \( h \) be such a bad hypothesis that has an error > \( \epsilon \)
Probability that \( h \) is consistent with one example is \( \Pr[f(x) = h(x)] < 1 - \epsilon \)

\[
\Pr[ h \text{ is bad but consist with one example } ] < 1 - \epsilon
\]
Towards formalizing Occam’s Razor

(*Assuming consistency*)

**Claim:** The probability that there is a hypothesis \( h \in H \) that:

1. Is **Consistent** with \( m \) examples, and
2. Has \( \text{err}_D(h) > \epsilon \)

is less than \( |H| \ (1 - \epsilon)^m \)

**Proof:** Let \( h \) be such a bad hypothesis that has an error \( > \epsilon \)

Probability that \( h \) is consistent with one example is \( \Pr[f(x) = h(x)] < 1 - \epsilon \)

\[
\Pr[ \text{h is bad but consist with one example }] < 1 - \epsilon
\]

The training set consists of \( m \) examples drawn independently

So, probability that \( h \) is consistent with \( m \) examples < \( (1 - \epsilon)^m \)
Towards formalizing Occam’s Razor

(Assuming consistency)

Claim: The probability that there is a hypothesis \( h \in H \) that:

1. Is Consistent with \( m \) examples, and
2. Has \( \text{err}_D(h) > \epsilon \)

is less than \(|H| \cdot (1 - \epsilon)^m\)

Proof: Let \( h \) be such a bad hypothesis that has an error \( > \epsilon \)

Probability that \( h \) is consistent with one example is \( \Pr[f(x) = h(x)] < 1 - \epsilon \)

\[
\Pr[\ h \text{ is bad but consist with one example } ] < 1 - \epsilon
\]

The training set consists of \( m \) examples drawn independently

So, probability that \( h \) is consistent with \( m \) examples < \((1 - \epsilon)^m\)

\[
\Pr[\ h \text{ is bad but consist with } m \text{ example } ] < (1 - \epsilon)^m
\]
Towards formalizing Occam’s Razor  
(Assuming consistency)

*Claim*: The probability that there is a hypothesis $h \in H$ that:
1. Is *consistent* with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| \left(1 - \epsilon \right)^m$

*Proof*: Let $h$ be such a bad hypothesis that has an error $> \epsilon$
Probability that $h$ is consistent with one example is $\Pr[f(x) = h(x)] < 1 - \epsilon$

$$\Pr[ \text{h is bad but consist with one example } ] < 1 - \epsilon$$

The training set consists of $m$ examples drawn independently
So, probability that $h$ is consistent with $m$ examples $< (1 - \epsilon)^m$

$$\Pr[ \text{h is bad but consist with m example } ] < (1 - \epsilon)^m$$

*Question*: What is the probability that there exists *some bad hypothesis* in $H$ that is consistent with $m$ examples?
Towards formalizing Occam’s Razor

(Assuming consistency)

**Claim**: The probability that there is a hypothesis $h \in H$ that:

1. Is **Consistent** with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| (1 - \epsilon)^m$

**Proof**: $\Pr[ h \text{ is bad but consistent with } m \text{ example } ] < (1 - \epsilon)^m$

**Event A**: there exists some bad hypothesis in $H$ that is consistent with $m$ examples
Towards formalizing Occam’s Razor

*(Assuming consistency)*

**Claim**: The probability that there is a hypothesis $h \in H$ that:

1. Is **Consistent** with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| \cdot (1 - \epsilon)^m$

**Proof**: $\Pr[\text{ h is bad but consist with } m\text{ example }] < (1 - \epsilon)^m$

**Event** $A$: there exists some bad hypothesis in $H$ that is consistent with $m$ examples

$A = [h_1 \text{ bad} & \text{consistent with } m\text{ example}] \text{ OR } [h_2 \text{ bad} & \text{consistent with } m\text{ example}] \text{ OR } ... \text{ OR } [\text{the last } h \text{ bad} & \text{consistent with } m\text{ examples}]$
Towards formalizing Occam’s Razor

*(Assuming consistency)*

**Claim**: The probability that there is a hypothesis $h \in H$ that:

1. Is **Consistent** with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| (1 - \epsilon)^m$

**Proof**: $\Pr[\text{ h is bad but consist with } m \text{ example }] < (1 - \epsilon)^m$

**Event** $A$: there exists some **bad hypothesis** in $H$ that is consistent with $m$ examples

$$A = [h_1 \text{ bad\&consistent with } m \text{ example}] \text{ OR } [h_2 \text{ bad\&consistent with } m \text{ example}] \text{ OR } \ldots \text{ OR } [\text{the last } h \text{ bad\&consistent with } m \text{ examples}]$$

$$A = B_1 \cup B_2 \cup B_3 \cup \ldots \cup B_{|H|}$$
Towards formalizing Occam’s Razor  

(*Assuming consistency*)

**Claim**: The probability that there is a hypothesis \( h \in H \) that:

1. Is **Consistent** with \( m \) examples, and
2. Has \( \text{err}_D(h) > \epsilon \)

is less than \( |H| (1 - \epsilon)^m \)

**Proof**: \( \Pr[ \text{h is bad but consist with } m \text{ example } ] < (1 - \epsilon)^m \)

**Event** \( A \): there exists **some bad hypothesis** in \( H \) that is consistent with \( m \) examples

\[
A = [h_1 \text{ bad}&\text{consistent with m example}] \text{ OR [h}_2 \text{ bad}&\text{consistent with m example}] \text{ OR ... OR [the last h bad}&\text{consistent with m examples]}
\]

\[
A = B_1 \cup B_2 \cup B_3 \cup ... \cup B_{|H|}
\]

\[
p(A) \leq p(B_1) + p(B_2) + p(B_3) + ... + p(B_{|H|}) \leq |H|(1 - \epsilon)^m
\]
Occam’s Razor

The probability that there is a hypothesis $h \in H$ that is

1. **Consistent** with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| (1 - \epsilon)^m$
Occam’s Razor

The probability that there is a hypothesis \( h \in H \) that is

1. Consistent with \( m \) examples, and
2. Has \( \text{err}_D(h) > \epsilon \)

is less than \( |H| (1 - \epsilon)^m \)

Just like before, we want to make this probability small, say smaller than \( \delta \)
Occam’s Razor

The probability that there is a hypothesis $h \in H$ that is

1. Consistent with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| (1 - \epsilon)^m$

Just like before, we want to make this probability small, say smaller than $\delta$

$|H| (1 - \epsilon)^m < \delta$
Occam’s Razor

The probability that there is a hypothesis $h \in H$ that is

1. **Consistent** with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| (1 - \epsilon)^m$

Just like before, we want to make this probability small, say smaller than $\delta$

$$|H| (1 - \epsilon)^m < \delta$$

$$\ln(|H|) + m \ln(1 - \epsilon) < \ln \delta$$
Occam’s Razor

The probability that there is a hypothesis $h \in H$ that is

1. Consistent with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| (1 - \epsilon)^m$

Just like before, we want to make this probability small, say smaller than $\delta$

$$|H| (1 - \epsilon)^m < \delta$$
$$\ln(|H|) + m \ln(1 - \epsilon) < \ln \delta$$

We know that $e^{-x} = 1 - x + \frac{x^2}{2} - \frac{x^3}{6} \cdots > 1 - x$

Let’s use $\ln(1 - \epsilon) < -\epsilon$ to get a safer $\delta$
Occam’s Razor

The probability that there is a hypothesis $h \in H$ that is

1. Consistent with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| (1 - \epsilon)^m$

Just like before, we want to make this probability small, say smaller than $\delta$

$$|H| (1 - \epsilon)^m < \delta$$

$$\ln(|H|) + m \ln(1 - \epsilon) < \ln \delta$$

We know that $e^{-x} = 1 - x + \frac{x^2}{2} - \frac{x^3}{6} \cdots > 1 - x$

Let’s use $\ln(1 - \epsilon) < -\epsilon$ to get a safer $\delta$

$$\ln(|H|) + m \ln(1 - \epsilon) < \ln(|H|) + m (-\epsilon) < \ln \delta$$

$$\ln(|H|) - \ln \delta < m \epsilon$$

$$m \epsilon > \ln(|H|) + \ln 1/\delta$$
Occam’s Razor

The probability that there is a hypothesis $h \in H$ that is

1. Consistent with $m$ examples, and
2. Has $\text{err}_D(h) > \epsilon$

is less than $|H| (1 - \epsilon)^m$

Just like before, we want to make this probability small, say smaller than $\delta$

$$|H| (1 - \epsilon)^m < \delta$$
$$\ln(|H|) + m \ln(1 - \epsilon) < \ln \delta$$

We know that $e^{-x} = 1 - x + \frac{x^2}{2} - \frac{x^3}{6} \cdots > 1 - x$

Let’s use $\ln(1 - \epsilon) < -\epsilon$ to get a safer $\delta$

That is, if $m > \frac{1}{\epsilon} \left( \ln(|H|) + \ln \frac{1}{\delta} \right)$ then, the probability of getting a bad hypothesis is small
Occam’s Razor

Let $H$ be any hypothesis space.

With probability at least $1 - \delta$, a hypothesis $h \in H$ that is consistent with a training set of size $m$ will have true error $< \epsilon$ if

$$m > \frac{1}{\epsilon} \left( \ln(|H|) + \ln \frac{1}{\delta} \right)$$
Occam’s Razor

Let H be any hypothesis space. With probability at least $1 - \delta$, a hypothesis $h \in H$ that is consistent with a training set of size $m$ will have true error $< \epsilon$ if

$$m > \frac{1}{\epsilon} \left( \ln(|H|) + \ln \frac{1}{\delta} \right)$$

1. Expecting lower error increases sample complexity (i.e. more examples needed for the guarantee)
Occam’s Razor

Let $H$ be any hypothesis space.

With probability at least $1 - \delta$, a hypothesis $h \in H$ that is consistent with a training set of size $m$ will have true error $< \varepsilon$ if

$$m > \frac{1}{\varepsilon} \left( \ln(|H|) + \ln \frac{1}{\delta} \right)$$

1. Expecting lower error increases sample complexity (i.e. more examples needed for the guarantee)

2. If we have a larger hypothesis space, then we will make learning harder (i.e. higher sample complexity)
Occam’s Razor

Let \( H \) be any hypothesis space. With probability at least \( 1 - \delta \), a hypothesis \( h \in H \) that is consistent with a training set of size \( m \) will have true error \( < \epsilon \) if

\[
m > \frac{1}{\epsilon} \left( \ln(|H|) + \ln \frac{1}{\delta} \right)
\]

1. Expecting lower error increases sample complexity (i.e. more examples needed for the guarantee)
2. If we have a larger hypothesis space, then we will make learning harder (i.e. higher sample complexity)
3. If we want a higher confidence in the classifier we will produce, sample complexity will be higher.
Occam’s Razor

Let $H$ be any hypothesis space.

With probability at least $1 - \delta$, a hypothesis $h \in H$ that is consistent with a training set of size $m$ will have true error $< \epsilon$ if

$$m > \frac{1}{\epsilon} \left( \ln(|H|) + \ln \frac{1}{\delta} \right)$$
Occam’s Razor

Let \( H \) be any hypothesis space.

With probability at least \( 1 - \delta \), a hypothesis \( h \in H \) that is consistent with a training set of size \( m \) will have true error < \( \epsilon \) if

\[
m > \frac{1}{\epsilon} \left( \ln(|H|) + \ln \frac{1}{\delta} \right)
\]

This reflects Occam’s Razor because it expresses a preference towards smaller hypothesis spaces.
Occam’s Razor

Let $H$ be any hypothesis space.

With probability at least $1 - \delta$, a hypothesis $h \in H$ that is consistent with a training set of size $m$ will have true error $< \varepsilon$ if

$$m > \frac{1}{\varepsilon} \left( \ln(|H|) + \ln \frac{1}{\delta} \right)$$

This reflects Occam’s Razor because it expresses a preference towards smaller hypothesis spaces.

Shows when a $m$-consistent hypothesis generalizes well (i.e. error $< \varepsilon$).
Occam’s Razor

Let $H$ be any hypothesis space.

With probability at least $1 - \delta$, a hypothesis $h \in H$ that is consistent with a training set of size $m$ will have true error $< \epsilon$ if

$$m > \frac{1}{\epsilon} \left( \ln(|H|) + \ln \frac{1}{\delta} \right)$$

This reflects Occam’s Razor because it expresses a preference towards smaller hypothesis spaces.

Shows when a $m$-consistent hypothesis generalizes well (i.e. error $< \epsilon$).

Complicated/larger hypothesis spaces are not necessarily bad. But simpler ones are unlikely to fool us by being consistent with many examples!
Consistent Learners and Occam’s Razor

From the definition, we get the following general scheme for PAC learning:

Given a sample of $m$ examples:
- Find some $h \in H$ that is consistent with all $m$ examples:
  - If $m$ is large enough, a consistent hypothesis must be close enough to $f$
  - Check that $m$ does not have to be too large (i.e., polynomial in the relevant parameters): we showed that the “closeness” guarantee requires that
    $$m > \frac{1}{\epsilon} (\ln |H| + \ln \frac{1}{\delta})$$
- Show that the consistent hypothesis $h \in H$ can be computed efficiently
Consistent Learners and Occam’s Razor

From the definition, we get the following general scheme for PAC learning

Given a sample of $m$ examples

- Find some $h \in H$ that is consistent with all $m$ examples
  - If $m$ is large enough, a consistent hypothesis must be close enough to $f$

  - Check that $m$ does not have to be too large (i.e. polynomial in the relevant parameters): we showed that the “closeness” guarantee requires that
    $$m > \frac{1}{\varepsilon} (\ln |H| + \ln \frac{1}{\delta})$$

- Show that the consistent hypothesis $h \in H$ can be computed efficiently

Question: what if $H$ is infinite? We will discuss it later