Computational Learning Theory

James Worrell
What is Machine Learning?

Best algorithm for predicting user ratings, based on previous ratings, without any other information about the users or films.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Team Name</th>
<th>Best Score</th>
<th>% Improvement</th>
<th>Last Submit Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BellKor's Pragmatic Chaos</td>
<td>0.8558</td>
<td>10.05</td>
<td>2009-06-26 18:42:37</td>
</tr>
<tr>
<td>2</td>
<td>PragmaticTheory</td>
<td>0.8582</td>
<td>9.80</td>
<td>2009-06-25 22:15:51</td>
</tr>
<tr>
<td>3</td>
<td>BellKor in BigChaos</td>
<td>0.8590</td>
<td>9.71</td>
<td>2009-05-13 08:14:09</td>
</tr>
<tr>
<td>4</td>
<td>Grand Prize Team</td>
<td>0.8593</td>
<td>9.68</td>
<td>2009-06-12 08:20:24</td>
</tr>
<tr>
<td>5</td>
<td>Dace</td>
<td>0.8604</td>
<td>9.56</td>
<td>2009-04-22 05:57:03</td>
</tr>
<tr>
<td>6</td>
<td>BigChaos</td>
<td>0.8613</td>
<td>9.47</td>
<td>2009-06-23 23:06:52</td>
</tr>
</tbody>
</table>
What is Machine Learning?

Neural network trained to IM level using database of chess positions
Machine Learning Problems

- **Classification**: assign a category to each data item, e.g., classify newspaper articles according to the topics *current events, sports, politics, entertainment, business*, and *other*. 
Machine Learning Problems

- **Classification**: assign a category to each data item, e.g., classify newspaper articles according to the topics *current events*, *sports*, *politics*, *entertainment*, *business*, and *other*;

- **Regression**: predict a numerical value for each data item, e.g., predict the value of a stock given various data about a company;
Machine Learning Problems

- **Classification**: assign a category to each data item, e.g., classify newspaper articles according to the topics *current events*, *sports*, *politics*, *entertainment*, *business*, and *other*;

- **Regression**: predict a numerical value for each data item, e.g., predict the value of a stock given various data about a company;

- **Clustering**: partition a set of items into classes, e.g., identify communities among users of a social network;
Machine Learning Problems

- **Classification**: assign a category to each data item, e.g., classify newspaper articles according to the topics *current events, sports, politics, entertainment, business, and other*;

- **Regression**: predict a numerical value for each data item, e.g., predict the value of a stock given various data about a company;

- **Clustering**: partition a set of items into classes, e.g., identify communities among users of a social network;

- **Ranking**: order items according to several criteria, e.g., order the results of a search query by relevance to a particular user.
What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand . . .
What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand . . .

. . . what concepts we can hope to learn efficiently, and how much data is necessary to learn them
What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand...

...what concepts we can hope to learn efficiently, and how much data is necessary to learn them

...what types of guarantees we might hope to achieve (error bounds, complexity bounds)
What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand . . .

. . . what concepts we can hope to learn efficiently, and how much data is necessary to learn them

. . . what types of guarantees we might hope to achieve (error bounds, complexity bounds)

. . . why particular algorithms may or may not perform well under various conditions
What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand...

...what concepts we can hope to learn efficiently, and how much data is necessary to learn them

...what types of guarantees we might hope to achieve (error bounds, complexity bounds)

...why particular algorithms may or may not perform well under various conditions

This generates useful intuition for algorithm design!
A classification problem in the consistency model comprises:

- **Domain**: a collection of sets $\mathcal{X} = \{X_n : n \in \mathbb{N}\}$ indexed by the dimension $n$. 
A classification problem in the consistency model comprises:

- **Domain**: a collection of sets $\mathcal{X} = \{X_n : n \in \mathbb{N}\}$ indexed by the dimension $n$.

- **Set of labels** $\mathcal{Y}$. In all our examples $\mathcal{Y}$ will either be $\{0, 1\}$ or $\{-1, +1\}$. A **labelled example** is a pair $(\tilde{x}, y)$ with input $\tilde{x} \in X_n$ and label $y \in \mathcal{Y}$.
Consistency Model

A classification problem in the consistency model comprises:

- **Domain**: a collection of sets $\mathcal{X} = \{X_n : n \in \mathbb{N}\}$ indexed by the dimension $n$.

- **Set of labels** $\mathcal{Y}$. In all our examples $\mathcal{Y}$ will either be $\{0, 1\}$ or $\{-1, +1\}$. A **labelled example** is a pair $(\bar{x}, y)$ with input $\bar{x} \in X_n$ and label $y \in \mathcal{Y}$.

- **Hypothesis class**: an indexed collection $\mathcal{H} = \{H_n : n \in \mathbb{N}\}$, with $H_n$ a set of functions $X_n \to \mathcal{Y}$. A hypothesis $h \in H_n$ is **consistent** with a set $S$ of labelled examples if $h(\bar{x}) = y$ for each pair $(\bar{x}, y) \in S$. 
Example - Spam Filtering

Date: Mon, 12 Oct 2015 19:36:45
From: FSTTCS 2015
Subject: Paper Submission 25
Not spam

Date: 13 Oct 2015 13:39:14
From: HITACHI Personal Finance
Your Latest Money Saving Tips
Spam

Date: Thu, 17 Sep 2015 05:00:12
From: registration@it.ox.ac.uk
Subject: Nexus mailbox expiry 01/10/15
???
Example - Spam Filtering

Represent training data as set of **labelled feature vectors**: 

<table>
<thead>
<tr>
<th>Bad Spelling</th>
<th>Attachment?</th>
<th>“Ben”</th>
<th>“Viagra”</th>
<th>Spam?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Example - Spam Filtering

Represent training data as set of **labelled feature vectors**:

<table>
<thead>
<tr>
<th>Bad Spelling</th>
<th>Attachment?</th>
<th>“Ben”</th>
<th>“Viagra”</th>
<th>Spam?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Find accurate **prediction rule**:

\[ 2 \cdot \text{BadSpell} - 3 \cdot \text{Ben} + 1 \cdot \text{Viagra} > 2 \]
Learnability in the Consistency Model

**Definition**

We say that algorithm $\mathcal{A}$ learns a hypothesis class $\mathcal{H}$ in the consistency model if given $n \in \mathbb{N}$ and a set of labelled examples $S \subseteq X_n \times Y$, $\mathcal{A}$ produces a hypothesis $h \in H_n$ consistent with $S$ if one exists and states that none exists otherwise. We moreover say that $\mathcal{A}$ **efficiently learns** $\mathcal{H}$ if it runs it time polynomial in the dimension $n$, the size of the example set $S$, and the size of the smallest representation of a hypothesis in $H_n$ consistent with $S$, assuming one exists.
Learning Boolean Conjunctions

<table>
<thead>
<tr>
<th>Guitar</th>
<th>Fast beat</th>
<th>Male singer</th>
<th>Acoustic</th>
<th>90s</th>
<th>Liked</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Learning Boolean Conjunctions

<table>
<thead>
<tr>
<th>Guitar</th>
<th>Fast beat</th>
<th>Male singer</th>
<th>Acoustic</th>
<th>90s</th>
<th>Liked</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ \neg \text{Guitar} \land \text{Fast beat} \land \text{Acoustic} \]
Algorithm to Learn Boolean Conjunctions

**Input:** Labelled sample $S \subseteq \{0, 1\}^n \times \{0, 1\}$

1. Let the initial hypothesis $\varphi$ be $p_1 \land \neg p_1 \land \ldots \land p_n \land \neg p_n$. 
Algorithm to Learn Boolean Conjunctions

**Input:** Labelled sample $S \subseteq \{0, 1\}^n \times \{0, 1\}$

1. Let the initial hypothesis $\varphi$ be $p_1 \land \neg p_1 \land \ldots \land p_n \land \neg p_n$.

2. For each positive example $\vec{x} = (x_1, \ldots, x_n)$, delete from $\varphi$ any literal not satisfied by $\vec{x}$. Do nothing for negative examples.
Algorithm to Learn Boolean Conjunctions

**Input:** Labelled sample $S \subseteq \{0, 1\}^n \times \{0, 1\}$

1. Let the initial hypothesis $\varphi$ be $p_1 \land \neg p_1 \land \ldots \land p_n \land \neg p_n$.

2. For each positive example $\vec{x} = (x_1, \ldots, x_n)$, delete from $\varphi$ any literal not satisfied by $\vec{x}$. Do nothing for negative examples.

3. Output the resulting formula $\varphi$ if it maps all the negative examples in $S$ to 0. Otherwise report that no consistent concept exists.
A linear classifier is a function $f : \mathbb{R}^n \rightarrow \{-1, 1\}$ given by

$$f(\mathbf{x}) = \begin{cases} +1 & \text{if } \mathbf{a} \cdot \mathbf{x} \geq b \\ -1 & \text{otherwise} \end{cases}$$
A linear classifier is a function $f : \mathbb{R}^n \to \{-1, 1\}$ given by

$$f(\vec{x}) = \begin{cases} 
+1 & \text{if } \vec{a} \cdot \vec{x} \geq b \\
-1 & \text{otherwise}
\end{cases}$$

Consistency with $S = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_k, y_k)\} \subseteq \mathbb{R}^n \times \{-1, 1\}$ formulated as a linear program:

$$\text{find } \vec{a} \in \mathbb{R}^n, b \in \mathbb{R} \text{ s.t.}$$

$$y_i(\vec{a} \cdot \vec{x}_i - b) \geq 1, i = 1, \ldots, k.$$
A linear classifier is a function $f : \mathbb{R}^n \to \{-1, 1\}$ given by

$$f(\vec{x}) = \begin{cases} +1 & \text{if } \vec{a} \cdot \vec{x} \geq b \\ -1 & \text{otherwise} \end{cases}$$

Consistency with $S = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_k, y_k)\} \subseteq \mathbb{R}^n \times \{-1, 1\}$ formulated as a linear program:

$$\text{find } \vec{a} \in \mathbb{R}^n, b \in \mathbb{R} \text{ s.t. } y_i(\vec{a} \cdot \vec{x}_i - b) \geq 1, i = 1, \ldots, k.$$ 

If a consistent linear classifier exists then we say that $S$ is linearly separable.
3-Term DNF

Positive examples
- (100000, 1)
- (010000, 1)
- (001000, 1)
- (000100, 1)
- (000010, 1)
- (000001, 1)

Negative examples
- (110000, 0)
- (100100, 0)
- (100010, 0)
- (011000, 0)
- (010001, 0)
- (001001, 0)
- (000011, 0)
3-Term DNF

Positive examples
(100000, 1)
(010000, 1)
(001000, 1)
(000100, 1)
(000010, 1)
(000001, 1)

Negative examples
(110000, 0)
(100100, 0)
(100010, 0)
(011000, 0)
(010001, 0)
(001001, 0)
(000011, 0)
Learnability in the Consistency Model

Definition
We say that algorithm $\mathcal{A}$ learns a hypothesis class $\mathcal{H}$ in the consistency model if given $n \in \mathbb{N}$ and a set of labelled examples $S \subseteq X_n \times Y$, $\mathcal{A}$ produces a hypothesis $h \in H_n$ consistent with $S$ if one exists and states that none exists otherwise. We moreover say that $\mathcal{A}$ efficiently learns $\mathcal{H}$ if it runs in time polynomial in the size of the set $S$. 
What’s Wrong With This Model?

<table>
<thead>
<tr>
<th>Guitar</th>
<th>Fast beat</th>
<th>Male singer</th>
<th>Acoustic</th>
<th>90s</th>
<th>Liked</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
What’s Wrong With This Model?

<table>
<thead>
<tr>
<th>Guitar</th>
<th>Fast beat</th>
<th>Male singer</th>
<th>Acoustic</th>
<th>90s</th>
<th>Liked</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Suppose we try to learn these data using the hypothesis class of DNF formulas . . .
What’s Wrong With This Model?

<table>
<thead>
<tr>
<th>Guitar</th>
<th>Fast beat</th>
<th>Male singer</th>
<th>Acoustic</th>
<th>90s</th>
<th>Liked</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Suppose we try to learn these data using the hypothesis class of DNF formulas . . .

The obvious algorithm yields hypothesis

\[(G \land F \land M \land \neg A \land 90s) \lor (G \land \neg F \land M \land A \land \neg 90s)\]
The PAC Model

- **PAC**: Probably Approximately Correct

Addresses two shortcomings of the consistency model:

- Measures the predictive power of hypotheses
- Handles noisy data
The PAC Model

- **PAC**: Probably Approximately Correct

- Addresses two shortcomings of the consistency model:
  - Measures the **predictive power** of hypotheses
  - Handles **noisy** data
A probability space is a triple \((\Omega, \mathcal{F}, \Pr)\) comprising:

- **Sample space** \(\Omega\): the set of outcomes so far and random experiment, e.g., for casting a die \(\Omega = \{1, 2, \ldots, 6\}\).
- **Events set** \(\mathcal{F}\): a collection of subsets of \(\Omega\) containing \(\Omega\), and their complement and union. For example, “the die lands on an odd number.”
- **Probability distribution** \(\Pr\): a mapping from \(\mathcal{F}\) to \([0, 1]\), giving the probability of an event, such that for any finite or countable family of disjoint events \(\{A_i: i \in I\}\), \(\Pr\left(\bigcup_{i \in I} A_i\right) = \sum_{i \in I} \Pr(A_i)\).
A probability space is a triple $(\Omega, \mathcal{F}, \Pr)$ comprising:

- **Sample space** $\Omega$: the set of all outcomes of a random experiment, e.g., for casting a die $\Omega = \{1, 2, \ldots, 6\}$. 
A probability space is a triple \((\Omega, \mathcal{F}, \Pr)\) comprising:

- **Sample space** \(\Omega\): the set of all outcomes of a random experiment, e.g., for casting a die \(\Omega = \{1, 2, \ldots, 6\}\).

- **Events set** \(\mathcal{F}\): a collection of subsets of \(\Omega\) containing \(\emptyset\) and \(\Omega\), and closed under complementation and countable union. For example, “the die lands on an odd number”.

A probability space is a triple \((\Omega, \mathcal{F}, \Pr)\) comprising:

- **Sample space** \(\Omega\): the set of all outcomes of a random experiment, e.g., for casting a die \(\Omega = \{1, 2, \ldots, 6\}\).

- **Events set** \(\mathcal{F}\): a collection of subsets of \(\Omega\) containing \(\emptyset\) and \(\Omega\), and closed under complementation and countable union. For example, “the die lands on an odd number”.

- **Probability distribution** \(\Pr\): A mapping from \(\mathcal{F}\) to \([0, 1]\), giving the probability of an event, such that for any finite or countable family of disjoint events \(\{A_i : i \in I\}\),

\[
\Pr \left( \bigcup_{i \in I} A_i \right) = \sum_{i \in I} \Pr(A_i)
\]
In this Course

(1) $\Omega = \{0, 1\}^n$, $\mathcal{F}$ = all subsets of $\Omega$, $Pr$ assigns a probability to each outcome.
(1) $\Omega = \{0, 1\}^n$, $\mathcal{F} =$ all subsets of $\Omega$, $\Pr$ assigns a probability to each outcome.

(2) $\Omega = \mathbb{R}^n$, $\mathcal{F}$ contains all reasonable subsets of $\mathbb{R}$, $\Pr$ defined by a density function, e.g., $\Omega = \mathbb{R}$ and

$$\Pr([a, b]) = \int_{a}^{b} f(x)dx$$
Conditional Probability and Independence

The conditional probability of event $A$ given event $B$ is defined to be

$$
Pr(A \mid B) = \frac{Pr(A \cap B)}{Pr(B)}
$$

if $Pr(B) > 0$.

Two events $A$ and $B$ are said to be independent if

$$
Pr(A \cap B) = Pr(A) Pr(B)
$$
A farmer believes that a certain species of apple tree will only bear fruit if the temperature and rainfall lie between certain upper and lower bounds. The farmer would like to determine these bounds by observing his orchard in successive growing seasons. How many seasons before the farmer can make an accurate estimation?
Hypothesis Rectangle
Border Regions

\[
\begin{align*}
Pr(E_1) &= Pr(E_2) = Pr(E_3) = Pr(E_4) = \varepsilon/4
\end{align*}
\]
Error Estimation

Given $\epsilon > 0$ and $\delta > 0$, how many samples are needed such that $\text{err}(R_S) \leq \epsilon$ with probability at least $1 - \delta$?
Given $\epsilon > 0$ and $\delta > 0$, how many samples are needed such that $\text{err}(R_S) \leq \epsilon$ with probability at least $1 - \delta$?

If the sample hits all four “border regions”, then:

$$
\text{err}(R_S) = \Pr(R \setminus R_S) \\
\leq \Pr(E_1 \cup E_2 \cup E_3 \cup E_4) \\
\leq \sum_{i=1}^{4} \Pr(E_i) \leq \epsilon,
$$
Given $\epsilon > 0$ and $\delta > 0$, how many samples are needed such that $\text{err}(R_S) \leq \epsilon$ with probability at least $1 - \delta$?

If the sample hits all four “border regions”, then:

$$\text{err}(R_S) = \Pr(R \setminus R_S) \leq \Pr(E_1 \cup E_2 \cup E_3 \cup E_4) \leq \sum_{i=1}^{4} \Pr(E_i) \leq \epsilon,$$

How many samples are need to hit all border regions with probability at least $1 - \delta$?
A Sample Bound

\[ \Pr(\text{all } m \text{ samples miss } E) = 1 - \frac{e}{m}. \]

\[ \Pr(\text{some } E_i \text{ missed by all } m \text{ samples}) \leq \frac{e}{m}. \]

But \( \frac{e}{m} \leq \frac{1}{(4/\epsilon)^{\log(4/\epsilon)}}. \)

Sample complexity is polynomial in \( 1/\epsilon \) and \( 1/\epsilon^2 \).
A Sample Bound

- $\Pr(\text{all } m \text{ samples miss } E_1) = \left(1 - \frac{\varepsilon}{4}\right)^m \leq e^{-\varepsilon m/4}$. 
A Sample Bound

- $\Pr(\text{all } m \text{ samples miss } E_1) = \left(1 - \frac{\varepsilon}{4}\right)^m \leq e^{-\varepsilon m/4}$.

- $\Pr(\text{some } E_i \text{ missed by all } m \text{ samples}) \leq 4e^{-\varepsilon m/4}$.
A Sample Bound

- \( \Pr(\text{all } m \text{ samples miss } E_1) = \left(1 - \frac{\varepsilon}{4}\right)^m \leq e^{-\varepsilon m/4}. \)

- \( \Pr(\text{some } E_i \text{ missed by all } m \text{ samples}) \leq 4e^{-\varepsilon m/4}. \)

- But \( 4e^{-\varepsilon m/4} \leq \delta \) iff \( m \geq \left(\frac{4}{\varepsilon}\right) \log(4/\delta). \)
A Sample Bound

- \( \Pr(\text{all } m \text{ samples miss } E_1) = (1 - \frac{\varepsilon}{4})^m \leq e^{-\varepsilon m/4}. \)

- \( \Pr(\text{some } E_i \text{ missed by all } m \text{ samples}) \leq 4e^{-\varepsilon m/4}. \)

- But \( 4e^{-\varepsilon m/4} \leq \delta \) iff \( m \geq (4/\varepsilon) \log(4/\delta). \)

Sample complexity is polynomial in \( 1/\varepsilon \) and \( 1/\delta. \)
Definition

A hypothesis class $\mathcal{H}$ is PAC learnable if there exists an algorithm $A$ with the following property: for every $n \in \mathbb{N}$, target concept $c \in H_n$, distribution $D$ on $X_n$, error parameter $\varepsilon > 0$, and confidence parameter $\delta > 0$, given $\text{poly}(n, 1/\varepsilon, 1/\delta)$ examples, $A$ outputs a hypothesis $h \in H_n$ such that $\text{err}(h) \leq \varepsilon$ with probability at least $1 - \delta$, where

$$\text{err}(h) = \Pr_{x \sim D} (h(x) \neq c(x))$$
Formal Definition (Realisable Case)

Definition  
A hypothesis class $\mathcal{H}$ is PAC learnable if there exists an algorithm $\mathcal{A}$ with the following property: for every $n \in \mathbb{N}$, target concept $c \in H_n$, distribution $D$ on $X_n$, error parameter $\varepsilon > 0$, and confidence parameter $\delta > 0$, given $\text{poly}(n, 1/\varepsilon, 1/\delta)$ examples, $\mathcal{A}$ outputs a hypothesis $h \in H_n$ such that $\text{err}(h) \leq \varepsilon$ with probability at least $1 - \delta$, where

$$\text{err}(h) = \Pr_{x \sim D} ( h(x) \neq c(x) )$$

Note that the number of examples does not depend on the distribution $D$. We say the model is distribution free or distribution independent.
The definition of PAC learning talks about sample complexity as a function of $n, \varepsilon, \delta$. We’re also interested in computational complexity.
The definition of PAC learning talks about **sample complexity** as a function of $n$, $\varepsilon$, $\delta$. We’re also interested in **computational complexity**.

**Definition**

A hypothesis class $\mathcal{H}$ is efficiently PAC-learnable if there exists a PAC-learning algorithm $A$ for $\mathcal{H}$ that runs in time $\text{poly}(n, 1/\varepsilon, 1/\delta)$. 

**Efficient PAC Learnability**
Boolean Conjunctions

- Fix a target concept $c$ on input set $\{0, 1\}^n$ and a distribution $D$ on $\{0, 1\}^n$. 
Boolean Conjunctions

- Fix a target concept $c$ on input set $\{0, 1\}^n$ and a distribution $D$ on $\{0, 1\}^n$.

- Given a sample $S$ chosen according to $D$ and labelled according to $c$, we can compute in time polynomial in $S$ “strongest” hypothesis $h$ that is consistent with $S$. 
Boolean Conjunctions

- Fix a target concept $c$ on input set ${0, 1}^n$ and a distribution $D$ on ${0, 1}^n$.

- Given a sample $S$ chosen according to $D$ and labelled according to $c$, we can compute in time polynomial in $S$ “strongest” hypothesis $h$ that is consistent with $S$.

- We show that if $S$ is sufficiently large then $h$ is probably approximately correct.
Boolean Conjunctions

- Fix a target concept $c$ on input set $\{0, 1\}^n$ and a distribution $D$ on $\{0, 1\}^n$.

- Given a sample $S$ chosen according to $D$ and labelled according to $c$, we can compute in time polynomial in $S$ “strongest” hypothesis $h$ that is consistent with $S$.

- We show that if $S$ is sufficiently large then $h$ is probably approximately correct.

- The hypothesis class of Boolean conjunctions is efficiently PAC learnable.
Sample Bounds in the Finite Case

Lemma
Fix an input set \( X \). Let \( H \) be a finite set of functions \( X \rightarrow \{0, 1\} \) and let \( c : X \rightarrow \{0, 1\} \). Given \( \varepsilon, \delta > 0 \), let

\[
m \geq \frac{1}{\varepsilon} \left( \ln |H| + \ln \left( \frac{1}{\delta} \right) \right).
\]

Then for any distribution \( D \) on \( X \), the probability that a sample of \( m \) points drawn from \( X \) according to \( D \) and labelled by \( c \) is consistent with some hypothesis \( h \in H \) that has error rate

\[
\Pr_{x \sim D} [h(x) \neq c(x)] \geq \varepsilon
\]

is at most \( \delta \).
3-Term DNF and 3-CNF

- Is the hypothesis class of 3-Term DNF PAC learnable?

By De Morgan distributivity, every 3-term DNF $\mathcal{I}_1 \land \mathcal{I}_2 \land \mathcal{I}_3$ has an equivalent 3-CNF:

$$\mathcal{I}_1 \land \mathcal{I}_2 \land \mathcal{I}_3 \land (\neg \mathcal{I}_1, \neg \mathcal{I}_2, \neg \mathcal{I}_3).$$

Boolean conjunctions

- 3-Term DNF

Now 3-CNF are sufficiently learnable in the consistency model!

Initial hypothesis := conjunction of all 3-literal clauses. Now delete any clause not satisfied by some positive example.

What is the running time?
3-Term DNF and 3-CNF

- Is the hypothesis class of 3-Term DNF PAC learnable?

- By De Morgan distributivity, every 3-term DNF $\tau_1 \lor \tau_2 \lor \tau_3$ has an equivalent 3-CNF:

$$\tau_1 \lor \tau_2 \lor \tau_3 \equiv \bigwedge_{\ell_1 \in \tau_1, \ell_2 \in \tau_2, \ell_3 \in \tau_3} (\ell_1 \lor \ell_2 \lor \ell_3).$$
3-Term DNF and 3-CNF

- Is the hypothesis class of 3-Term DNF PAC learnable?

- By De Morgan distributivity, every 3-term DNF $\tau_1 \lor \tau_2 \lor \tau_3$ has an equivalent 3-CNF:

$$
\tau_1 \lor \tau_2 \lor \tau_3 \equiv \bigwedge_{l_1 \in \tau_1, l_2 \in \tau_2, l_3 \in \tau_3} (l_1 \lor l_2 \lor l_3).
$$

Boolean conjunctions $\subseteq$ 3-Term DNF $\subseteq$ 3-CNF
3-Term DNF and 3-CNF

- Is the hypothesis class of 3-Term DNF PAC learnable?

- By De Morgan distributivity, every 3-term DNF $\tau_1 \lor \tau_2 \lor \tau_3$ has an equivalent 3-CNF:

$$\tau_1 \lor \tau_2 \lor \tau_3 \equiv \bigwedge_{\ell_1 \in \tau_1, \ell_2 \in \tau_2, \ell_3 \in \tau_3} (\ell_1 \lor \ell_2 \lor \ell_3).$$

- Boolean conjunctions $\subseteq$ 3-Term DNF $\subseteq$ 3-CNF

- Now 3-CNF are efficiently learnable in the consistency model!
3-Term DNF and 3-CNF

- Is the hypothesis class of 3-Term DNF PAC learnable?

- By De Morgan distributivity, every 3-term DNF $\tau_1 \lor \tau_2 \lor \tau_3$ has an equivalent 3-CNF:

$$\tau_1 \lor \tau_2 \lor \tau_3 \equiv \bigwedge_{\ell_1 \in \tau_1, \ell_2 \in \tau_2, \ell_3 \in \tau_3} (\ell_1 \lor \ell_2 \lor \ell_3).$$

Boolean conjunctions $\subseteq$ 3-Term DNF $\subseteq$ 3-CNF

- Now 3-CNF *are* efficiently learnable in the consistency model!

  - Initial hypothesis $\coloneqq$ conjunction of all 3-literal clauses. Now delete any clause not satisfied by some positive example.
3-Term DNF and 3-CNF

- Is the hypothesis class of 3-Term DNF PAC learnable?

- By De Morgan distributivity, every 3-term DNF $\tau_1 \lor \tau_2 \lor \tau_3$ has an equivalent 3-CNF:

$$\tau_1 \lor \tau_2 \lor \tau_3 \equiv \bigwedge_{\ell_1 \in \tau_1, \ell_2 \in \tau_2, \ell_3 \in \tau_3} (\ell_1 \lor \ell_2 \lor \ell_3).$$

Boolean conjunctions $\subseteq 3$-Term DNF $\subseteq 3$-CNF

- Now 3-CNF are efficiently learnable in the consistency model!

- Initial hypothesis := conjunction of all 3-literal clauses. Now delete any clause not satisfied by some positive example.

- What is the running time?
PAC-Learnability of 3-CNF

The hypothesis class of 3-CNF formulas is efficiently learnable in the consistency model . . .
PAC-Learnability of 3-CNF

The hypothesis class of 3-CNF formulas is efficiently learnable in the consistency model . . .

. . . now apply sample bound to deduce PAC learnability.

\[ m \geq \frac{1}{\varepsilon} \left( \ln |H| + \ln \left( \frac{1}{\delta} \right) \right) . \]
What about DNF Formulas?

Recall that there was a trivial algorithm to learn DNF in the consistency model.
What about DNF Formulas?

Recall that there was a trivial algorithm to learn DNF in the consistency model.

But what happens when we try to apply the sample bound to obtain PAC learnability?

\[ m \geq \frac{1}{\varepsilon} \left( \ln |H| + \ln \left( \frac{1}{\delta} \right) \right). \]
Agnostic Setting

- What if there’s no target function (e.g., features are weight and height; labels are “male” and “female”)? Or what if the data are noisy – labels get flipped, etc?
Agnostic Setting

- What if there’s no target function (e.g., features are weight and height; labels are “male” and “female”)? Or what if the data are noisy – labels get flipped, etc?

- We’re in the agnostic (or unrealisable) setting: labels produced by distribution $D$ on pairs $(\tilde{x}, y)$ of inputs and labels.
Agnostic Setting

What if there’s no target function (e.g., features are weight and height; labels are “male” and “female”)? Or what if the data are noisy – labels get flipped, etc?

We’re in the agnostic (or unrealisable) setting: labels produced by distribution $D$ on pairs $(\tilde{x}, y)$ of inputs and labels.

Now define

$$err(h) := \Pr_{(\tilde{x}, y) \sim D} (h(\tilde{x}) \neq y).$$

Aim to get as close as possible to $\min_{h \in H} err(h)$. 
Agnostic Setting

- What if there’s no target function (e.g., features are weight and height; labels are “male” and “female”)? Or what if the data are noisy – labels get flipped, etc?

- We’re in the agnostic (or unrealisable) setting: labels produced by distribution $D$ on pairs $(\tilde{x}, y)$ of inputs and labels.

- Now define
  \[
  \text{err}(h) := \Pr_{(\tilde{x}, y) \sim D} (h(\tilde{x}) \neq y).
  \]
  Aim to get as close as possible to $\min_{h \in \mathcal{H}} \text{err}(h)$.

- Strategy: given sample $\{(\tilde{x}_1, y_1), \ldots, (\tilde{x}_m, y_m)\}$, minimise the empirical error
  \[
  \hat{\text{err}}(h) = \frac{1}{m} |\{i : h(\tilde{x}_i) \neq y_i\}|.
  \]
PAC Learnability (Agnostic Setting)

**Definition**
The hypothesis class $\mathcal{H}$ is PAC learnable if there exists an algorithm $A$ with the following property: for every $n \in \mathbb{N}$ and distribution $D$ on $X_n \times Y$, error parameter $\varepsilon > 0$, and confidence parameter $\delta > 0$, given access to $\text{poly}(n, 1/\varepsilon, 1/\delta)$ examples $A$ outputs a hypothesis $h_S \in H_n$ such that

$$\text{err}(h_S) - \min_{h \in H} \text{err}(h) \leq \varepsilon$$

with probability at least $1 - \delta$. 

Hoeffding’s Inequality

Theorem

(Hoeffding’s inequality) Let $Z_1, Z_2, \ldots, Z_m$ be independent random variables such that $Z_i \in [0, 1]$ for all $i$. Let $\hat{p}$ be the empirical mean of these random variables,

$$\hat{p} = \frac{1}{m} \sum_{i=1}^{m} Z_i$$

and let $p$ be its expected value,

$$p = E[\hat{p}] = \frac{1}{m} \sum_{i=1}^{m} E[Z_i] .$$

Then for any $\varepsilon > 0$,

$$\Pr (|p - \hat{p}| \geq \varepsilon) \leq 2e^{-2\varepsilon^2 m} .$$
Theorem
For any sample \( S \subseteq X \times Y \), let \( h_S \in H \) denote a function with minimal empirical error over \( S \). Then for any distribution \( D \) over \( X \times Y \), for any \( \varepsilon, \delta > 0 \), if a sample \( S \) of cardinality

\[
m \geq \frac{2}{\varepsilon^2} \left( \log |H| + \log \left( \frac{2}{\delta} \right) \right)
\]

(1)

is drawn independently from \( D \), the function \( h_S \) satisfies

\[
\text{err}(h_S) - \min_{h \in H} \text{err}(h) \leq \varepsilon
\]

(2)

with probability at least \( 1 - \delta \).
Proof

Suppose the sample $S$ satisfies **uniform convergence**:

$$\forall h \in H, \ \left| \text{err}(h) - \widehat{\text{err}}(h) \right| \leq \frac{\varepsilon}{2}.$$
Proof

Suppose the sample $S$ satisfies \textit{uniform convergence}:

$$\forall h \in H, |\text{err}(h) - \hat{\text{err}}(h)| \leq \frac{\varepsilon}{2}.$$  

Then for any $h \in H$,

$$\text{err}(h_S) \leq \hat{\text{err}}(h_S) + \frac{\varepsilon}{2} \quad \text{(uniform convergence)}$$

$$\leq \hat{\text{err}}(h) + \frac{\varepsilon}{2} \quad \text{(since } \hat{\text{err}}(h_S) \leq \hat{\text{err}}(h)\text{)}$$

$$\leq \text{err}(h) + \varepsilon \quad \text{(uniform convergence)}$$
Proof

Suppose the sample $S$ satisfies *uniform convergence*:

$$\forall h \in H, |\text{err}(h) - \hat{\text{err}}(h)| \leq \frac{\varepsilon}{2}.$$ 

Then for any $h \in H$,

$$\text{err}(h_S) \leq \hat{\text{err}}(h_S) + \frac{\varepsilon}{2} \quad \text{(uniform convergence)}$$

$$\leq \hat{\text{err}}(h) + \frac{\varepsilon}{2} \quad \text{(since } \hat{\text{err}}(h_S) \leq \hat{\text{err}}(h))$$

$$\leq \text{err}(h) + \varepsilon \quad \text{(uniform convergence)}$$

Since the derivation above holds for any $h \in H$, we have that

$$\text{err}(h_S) \leq \min_{h \in H} \text{err}(h) + \varepsilon.$$
Proof

Fix $h \in H$ and define

$$Z_i = \begin{cases} 
1 & \text{if } h(x_i) \neq y_i \\
0 & \text{otherwise.} 
\end{cases}$$
Proof

Fix $h \in H$ and define

$$Z_i = \begin{cases} 1 & \text{if } h(x_i) \neq y_i \\ 0 & \text{otherwise.} \end{cases}$$

Then $\hat{\text{err}}(h) = (1/m) \sum_{i=1}^{m} Z_i$ and $\text{err}(h) = E[\hat{\text{err}}(h)]$, Thus,

$$\Pr \left( |\text{err}(h) - \hat{\text{err}}(h)| \geq \frac{\varepsilon}{2} \right) \leq 2e^{-\varepsilon^2 m/2}.$$
Proof

Fix \( h \in H \) and define

\[
Z_i = \begin{cases} 
1 & \text{if } h(x_i) \neq y_i \\
0 & \text{otherwise.}
\end{cases}
\]

Then \( \hat{\text{err}}(h) = (1/m) \sum_{i=1}^{m} Z_i \) and \( \text{err}(h) = E[\hat{\text{err}}(h)] \), Thus,

\[
\Pr \left( |\text{err}(h) - \hat{\text{err}}(h)| \geq \frac{\varepsilon}{2} \right) \leq 2e^{-\varepsilon^2 m/2}.
\]

Applying the union bound, we get

\[
\Pr \left( \exists h \in H : |\text{err}(h) - \hat{\text{err}}(h)| \geq \frac{\varepsilon}{2} \right) \leq 2|H|e^{-\varepsilon^2 m/2}.
\]
The Bias-Variance Tradeoff

\[
\text{err}(h_S) \leq \min_{h \in H} \text{err}(h) + \sqrt{\frac{2 \log |H| + 2 \log(2/\delta)}{m}}.
\]
Our End Goal

Theorem

Let $H$ be finite. With probability $1 - \delta$ over samples of size $m$, for all $h \in H$ we have

$$\text{err}(h) \leq \widehat{\text{err}}(h) + \sqrt{\frac{2 \log |H| + 2 \log (2/\delta)}{m}}.$$
Our End Goal

**Theorem**

*Let $H$ be finite. With probability $1 - \delta$ over samples of size $m$, for all $h \in H$ we have*

$$\text{err}(h) \leq \hat{\text{err}}(h) + \sqrt{\frac{2 \log |H| + 2 \log (2/\delta)}{m}}.$$  

**Theorem**

*Let $H$ have VC dimension $d$. With probability $1 - \delta$ over samples of size $m$, for all $h \in H$ we have*

$$\text{err}(h) \leq \hat{\text{err}}(h) + O \left( \sqrt{\frac{\log (m/d)}{m/d}} \right) + O \left( \sqrt{\frac{\log 1/\delta}{m}} \right).$$
Our End Goal

Theorem

Let $H$ be finite. With probability $1 - \delta$ over samples of size $m$, for all $h \in H$ we have

$$\text{err}(h) \leq \hat{\text{err}}(h) + \sqrt{\frac{2 \log |H| + 2 \log (2/\delta)}{m}}.$$  

Theorem

Let $H$ have VC dimension $d$. With probability $1 - \delta$ over samples of size $m$, for all $h \in H$ we have

$$\text{err}(h) \leq \hat{\text{err}}(h) + O \left( \sqrt{\frac{\log(m/d)}{m/d}} \right) + O \left( \sqrt{\frac{\log 1/\delta}{m}} \right).$$

For finite $H$, what is an upper bound on the VC dimension of $H$?
Loss Functions

Given a hypothesis $h : X \to Y$, we define an associated 0-1 loss function $g : X \times Y \to \{0, 1\}$ by

$$g(x, y) = \begin{cases} 1 & \text{if } h(x) \neq y \\ 0 & \text{otherwise.} \end{cases}$$
Hinge Loss

Consider a linear classifier $h(\vec{x}) = \text{sign}(f(\vec{x}))$ for some linear function $f(\vec{x}) = \vec{w} \cdot \vec{x} + b$. The hinge loss function associated to such a classifier is defined by

$$g(\vec{x}, y) = \max(0, 1 - yf(\vec{x})).$$
Rademacher Complexity

Let $G$ be a family of loss functions mapping a set $Z$ into $\mathbb{R}$ and let $D$ be a distribution over $Z$.

Definition

Let $\sigma = (\sigma_1, \ldots, \sigma_m)$ be a list of independent random variables taking values $-1$ and $+1$ with equal probability. Then the empirical Rademacher complexity of $G$ with respect to $S \subseteq Z$ is defined to be

$$R_S(G) = \mathbb{E}_\sigma \left[ \sup_{g \in G} \frac{1}{m} \sum_{i=1}^{m} \sigma_i g(z_i) \right]. \quad (3)$$

For any integer $m \geq 1$ the Rademacher complexity of $G$ with respect to samples of size $m$ drawn according to $D$ is

$$R_m(G) = \mathbb{E}_{S \sim D^m} [R_S(G)].$$
Theorem (McDiarmid’s Inequality)

Let $f : V^m \rightarrow \mathbb{R}$ be a function such that for some $c > 0$ and all $x_1, \ldots, x_m, x'_i \in V$,

$$|f(x_1, \ldots, x_i, \ldots, x_m) - f(x_1, \ldots, x'_i, \ldots, x_m)| \leq c.$$ 

Let $X_1, \ldots, X_m$ be independent random variables taking values in $V$. Then for all $\varepsilon > 0$,

$$\Pr(f(X_1, \ldots, X_m) \geq E[f(X_1, \ldots, X_m)] + \varepsilon) \leq e^{-2\varepsilon^2/mc^2}.$$
Uniform Convergence

Given a loss function $g : Z \rightarrow \mathbb{R}$, define the **expected loss** of $g$ w.r.t. distribution $D$ on $Z$ to be $E[g] := E_{z \sim D}[g(z)]$.

Given $S = \{z_1, \ldots, z_m\} \subseteq Z$, define the **empirical loss** of $g$ on $S$ to be $\hat{E}_S[g] := \frac{1}{m} \sum_{i=1}^{m} g(z_i)$.

**Theorem (Uniform Convergence)**

Let $G$ be a family of functions mapping a set $Z$ to the unit interval $[0, 1]$. Suppose that a sample $S$ of size $m$ is drawn according to distribution $D$ on $Z$. Then for any $\delta > 0$, with probability at least $1 - \delta$ the following holds for all functions $g \in G$:

$$E[g] \leq \hat{E}_S[g] + 2R_m(G) + O\left(\sqrt{\frac{\log \frac{1}{\delta}}{m}}\right).$$
VC Bounds on Rademacher Complexity

Lemma (Massart’s Lemma)

Let $A \subseteq \mathbb{R}^m$ be a finite set of vectors with $\|a\| \leq 1$ for all $a \in A$. Then

$$E_{\sigma} \left[ \max_{a \in A} \sum_{i=1}^{m} \sigma_i a_i \right] \leq \sqrt{2 \log |A|},$$

where the $\sigma_i$ are independent random variables uniform over $\{-1, +1\}$ and $a_1, \ldots, a_m$ are the components of vector $a$.

Corollary

Let $H$ be a family of functions on a domain $X$, taking values in $\{-1, +1\}$, with VC dimension $d$. Let $S = \{x_1, \ldots, x_m\} \subseteq X$, then

$$R_S(H) = O \left( \sqrt{\frac{\log(m/d)}{m/d}} \right)$$