Recap & Today

- So far:
  1. Online decision making and online learning
  2. (Randomized) Weighted Majority for tracking the best expert(s)
  3. Decision making: Kelly criterion

- Going forward:
  1. Batch (offline) learning: setting and definitions
  2. Probably Approximately Correct (PAC) model
  3. PAC learning in noiseless and noisy setting
  4. Efficient PAC learning

- After PAC – back to learning algorithms:
  1. Convex analysis
  2. Efficient learning algorithms, this time for real
Let’s talk about ants
Predicting Ant’s Next Move

- Predict next move direction of an ant $\rightarrow \leftarrow \uparrow \downarrow$
- Build elaborate model of sense antennas and olfactory system

$\Rightarrow$ Potentially good understanding albeit poor predictability

- Record many traces & features (dampness, season, ...)

$\Rightarrow$ Potentially good predictability albeit poor understanding
Chomsky vs. Jelinek

- Noam Chomsky: modern linguist, *universal grammar* theory, *generative grammar* theory, the Chomsky hierarchy

- At Brains, Minds, and Machines symposium Chomsky derided machine learning research for using statistical learning methods to produce behavior that mimics something in the world, but who don’t try to understand the meaning of that behavior.

- Fred Jelinek: EE training from MIT, head of (famous) IBM team for research in speech recognition and machine translation

- “Every time I fire a linguist, the performance of the speech recognizer goes up.”
High-Low game

- Goal: predict whether NFL player had > 2 concussions based on his level of Chronic Traumatic Encephalopathy (CTE)

- Input: pairs of (CTE-level, #concussions > 2)
  \[(0.07, -)(0.9, +)(0.4, +)(0.73, +)(0.2, -)\]

- Output: prediction rule sign[cte − θ]

- “Learning algorithm”: 
  - Set
    \[\hat{\theta} = \min\{x^i : y^i = +1\}\]
  - Set
    \[\hat{\theta} = \max\{x^i : y^i = -1\}\]
  - Set
    \[\hat{\theta} = \frac{1}{2} (\min\{x^i : y^i = +1\} + \max\{x^i : y^i = -1\})\]
Error Analysis - I

- Range of CTE level is $[0, 1]$
- Distribution of CTE levels $\mathcal{D}$ is uniform over $[0, 1]$
- $\exists \theta^* \text{ s.t. } y^i = \text{sign}(x^i - \theta^*)$
- Use the first rule $\hat{\theta} = \min\{x^i : y^i = +1\}$
- Denote $\tilde{\theta} = \max\{x^i : y^i = -1\}$
- “Noman land” $(\tilde{\theta}, \hat{\theta})$
- Samples are identically distributed and independent
Example
Error Analysis - II

• How likely a prediction using \( \hat{\theta} \) would wrong w.p. \( > \varepsilon \) on unseen samples?

• A single sample falls outside no man’s land w.p.

\[
1 - (\hat{\theta} - \tilde{\theta}) \leq 1 - \varepsilon
\]

• Set of \( m \) i.i.d samples all fall outside no man’s land w.p.

\[
(1 - \varepsilon)^m \leq e^{-\varepsilon m}
\]

• Suppose we want that \( \varepsilon \leq 0.001 \) and be at least 99% sure about that we found an accurate threshold

• Then, \( e^{-\varepsilon m} \leq \delta \) where \( \varepsilon = 0.001 \) and \( \delta = 0.01 \)

• Number of samples suffices to be

\[
m \geq \frac{1}{\varepsilon} \log \left( \frac{1}{\delta} \right) = 1000 \times \log(100) \approx 4605
\]
Adversarial High-Low Game

- Online setting $x^t \rightsquigarrow \hat{y}^t = \text{sign}(x^t - \theta^t) \rightsquigarrow y^t \rightsquigarrow \theta^{t+1}$
- Instead of i.i.d. samples, assume $\exists$ adversity:
  - Remembers past examples
    \[
    a = \max_{y^i = -1} x^i ; \quad b = \min_{y^i = +1} x^i
    \]
  - Picks next $x^t = \frac{1}{2}(a + b)$
  - Gets $\hat{y}^t$ and sets $y^t = -\hat{y}^t$
Batch (Offline) Learning

- Instance domain: $\mathcal{X}$ (e.g. $\mathcal{X} = \mathbb{R}^d$)
- Target (label) domain: $\mathcal{Y}$ (e.g. $\mathcal{Y} = \{-1, +1\}$)
- Learner’s input – training data:

  $$S = \{(x^1, y^1), (x^2, y^2), \ldots, (x^m, y^m)\} \in (\mathcal{X} \times \mathcal{Y})^m$$

- Learner’s output – predictor / classifier:

  $$h : \mathcal{X} \rightarrow \mathcal{Y} \quad [\hat{y} = h(x)]$$
Batch (Offline) Learning

• Instance domain: $\mathcal{X}$ (e.g. $\mathcal{X} = \mathbb{R}^d$)
• Target (label) domain: $\mathcal{Y}$ (e.g. $\mathcal{Y} = \{-1, +1\}$)
• Learner’s input – training data:

$$S = \{(x^1, y^1), (x^2, y^2), \ldots, (x^m, y^m)\} \in (\mathcal{X} \times \mathcal{Y})^m$$

• Learner’s output – predictor / classifier:

$$h : \mathcal{X} \rightarrow \mathcal{Y} \quad [\hat{y} = h(x)]$$

• What should be the learning goal?
• Find $h$ with (mostly) correct predictions on unseen examples
Data Model Assumptions

- Must assume relation between training and test (unseen) data
- Assumptions need to be realistic & enable efficient learning
- Statistical assumption, not adversarial, not “time dependent”
- Examples are Independently Identically Distributed (i.i.d.):

\[ x^i \sim \mathcal{D} \Rightarrow \mathcal{D}(x^1, x^2, \ldots, x^m) = \prod_{i=1}^{m} \mathcal{D}(x^i) \]

- For instance:
  
  Dice \( \mathcal{D}(x^1 = i^1, \ldots, x^m = i^m) = (1/6)^{-m} \)
  
  Hypercube \( \mathcal{X} = \{-1, +1\}^n \) \( \mathcal{D}[\mathbf{b}] = 2^{-n} \)
  
  Normal \( \mathcal{X} = \mathbb{R}^n \) \( \mathcal{D}(\mathbf{x}) = (2\pi)^{-n/2} \exp(-1/2\|\mathbf{x}\|^2) \)
Realizability

- If \( x \) is R.V. what about \( y \)?
- General case, uncertainty in \( Y = y \) even when we observe \( X = x \)
  
  \[ D(X, Y) \]

- Realizable case:
  
  \[ D(Y = +1 | X = x) = 1 \quad \text{or} \quad D(Y = -1 | X = x) = 1 \]

- We can define
  
  \[ h^*(x) = \text{sign} \left( D(Y = +1 | x) - \frac{1}{2} \right) \]
Unseen Examples

- Let $h^*$ be the (unknown) correct classifier
- We should find $h$ s.t. $h \approx h^*$
- Define error of $h$ w.r.t. $h^*$ to be
  \[
  \mathcal{L}_D(h) = \mathbb{P}_{x \sim \mathcal{D}}[h(x) \neq h^*(x)]
  \]
  where $\mathcal{D}$ is some (unknown) probability measure over $\mathcal{X}$
- Is it possible to find $h$ s.t. $\mathcal{L}_D(h)$ is arbitrarily small?
Only Approximately

- **Claim:** We cannot hope to find \( h \) s.t. \( \mathcal{L}_D(h) = 0 \)

- **Proof:** for every \( \varepsilon \in (0, 1) \) set

\[
\mathcal{X} = \{0, 1\} \quad \mathcal{D}(0) = 1 - \varepsilon \quad \mathcal{D}(1) = \varepsilon
\]

- Probability not to see \( x = 1 \) among \( m \) i.i.d. samples is

\[
(1 - \varepsilon)^m \approx e^{-\varepsilon m}
\]

- If \( m < 2/\varepsilon \) we won’t observe \( x^i = 1 \) w.p. of \( e^{-2} > 10\% \)

- We can only guess the label when \( x = 1 \)

- **Relaxed Goal:** find \( h \) s.t. \( \mathcal{L}_D(h) \leq \varepsilon \) for a pre-specified \( \varepsilon \)
Only Probably

- Examples are generated according to an i.i.d. process
- We may obtain the same (or very similar) sample over and over
- No algorithm can guarantee $\mathcal{L}_D(h) \leq \varepsilon$ for sure
- Relaxed goal: learning algorithm may fail (completely) w.p. $\delta$
  (Probability is over random choices of examples)
**Probably Approximately Correct (PAC)**

- Learning algorithm (learner) does not know $\mathcal{D}$ and $h^*$
- Learner receives accuracy, $\varepsilon$, and confidence, $\delta$, parameters
- Learner obtains training data, $S$, of $m(\varepsilon, \delta)$ examples
- Number of examples may depend on $\varepsilon$ and $\delta$
- Number of examples can *neither* depend on $\mathcal{D}$ nor on $h^*$
- Learner finds an hypothesis $h$ s.t.

$$\mathcal{L}_{\mathcal{D}}(h) \leq \varepsilon \text{ w.p. } 1 - \delta$$

- Predictor $h$ is:
  - *Probably* (w.p $\geq 1 - \delta$)
  - *Approximately* (within accuracy of $\varepsilon$)
  - *Correct*
Historical Perspective

Probably Approximately Correct Learning (PAC)

Leslie G. Valiant. 
A Theory of the Learnable. 
No Free Lunch

- Suppose that $|X| = \infty$
- For any finite $C \subset X$ take $\mathcal{D}$ to be uniform distribution over $C$
- If number of training examples is $m \leq |C|/2$ the learner has no knowledge on at least half the elements in $C$
- Formalizing the above, it can be shown that:
No Free Lunch

• Suppose that $|\mathcal{X}| = \infty$
• For any finite $C \subset \mathcal{X}$ take $\mathcal{D}$ to be uniform distribution over $C$
• If number of training examples is $m \leq |C|/2$ the learner has no knowledge on at least half the elements in $C$
• Formalizing the above, it can be shown that:

Theorem (No Free Lunch)

Fix $\delta \in (0, 1), \varepsilon < 1/2$. For every learner $A$ and training set size $m$, there exists $\mathcal{D}, f$ such that with probability of at least $\delta$ over the generation of a training data, $S$, of $m$ examples, it holds that $\mathcal{L}_{\mathcal{D}, f}(A(S)) \geq \varepsilon$. 

Remark: $\mathcal{L}_{\mathcal{D}, f}(\text{random guess}) = 1/2$, so the theorem states that you can't be better than a random guess.
No Free Lunch

• Suppose that $|\mathcal{X}| = \infty$
• For any finite $C \subset \mathcal{X}$ take $D$ to be uniform distribution over $C$
• If number of training examples is $m \leq |C|/2$ the learner has no knowledge on at least half the elements in $C$
• Formalizing the above, it can be shown that:

Theorem (No Free Lunch)

Fix $\delta \in (0, 1)$, $\varepsilon < 1/2$. For every learner $A$ and training set size $m$, there exists $D, f$ such that with probability of at least $\delta$ over the generation of a training data, $S$, of $m$ examples, it holds that $\mathcal{L}_{D,f}(A(S)) \geq \varepsilon$.

Remark: $\mathcal{L}_{D,f}$(random guess) = 1/2, so the theorem states that you can’t be better than a random guess
Prior Knowledge

- Give more knowledge to the learner: the target $f$ comes from some hypothesis class, $\mathcal{H} \subset \mathcal{Y}^X$
- The learner knows $\mathcal{H}$
- Is it possible to PAC learn now?
- Of course, the answer depends on $\mathcal{H}$ since the No Free Lunch theorem tells us that for $\mathcal{H} = \mathcal{Y}^X$ the problem is not solvable ...
Learning Finite Classes

- Assume that $\mathcal{H}$ is a finite hypothesis class
  - E.g.: $\mathcal{H}$ is all the functions from $\mathcal{X}$ to $\mathcal{Y}$ that can be implemented using a Python program of length at most $b$
- Use the **Consistent** learning rule:
  - Input: $\mathcal{H}$ and $S = (x_1, y_1), \ldots, (x_m, y_m)$
  - Output: any $h \in \mathcal{H}$ s.t. $\forall i, y_i = h(x_i)$
- This is also called **Empirical Risk Minimization (ERM)**
Learning Finite Classes

- Assume that $\mathcal{H}$ is a finite hypothesis class
  - E.g.: $\mathcal{H}$ is all the functions from $\mathcal{X}$ to $\mathcal{Y}$ that can be implemented using a Python program of length at most $b$

- Use the **Consistent** learning rule:
  - Input: $\mathcal{H}$ and $S = (x_1, y_1), \ldots, (x_m, y_m)$
  - Output: any $h \in \mathcal{H}$ s.t. $\forall i, y_i = h(x_i)$

- This is also called **Empirical Risk Minimization (ERM)**

$\text{ERM}_\mathcal{H}(S)$

- Input: training set $S = (x_1, y_1), \ldots, (x_m, y_m)$
- Define the empirical risk: $\mathcal{L}_S(h) = \frac{1}{m}\{|i : h(x_i) \neq y_i|\}$
- Output: any $h \in \mathcal{H}$ that minimizes $\mathcal{L}_S(h)$
Learning Finite Classes

**Theorem**

Fix $\epsilon, \delta$. If $m \geq \frac{\log(|\mathcal{H}|/\delta)}{\epsilon}$ then for every $\mathcal{D}, f$, with probability of at least $1 - \delta$ (over the choice of $S$ of size $m$),

$$\mathcal{L}_{\mathcal{D}, f}(\text{ERM}_{\mathcal{H}}(S)) \leq \epsilon.$$
Proof

• Let $S|_x = (x_1, \ldots, x_m)$ be the instances of the training set
• We would like to prove:

$$\mathcal{D}^m(\{S|_x : \mathcal{L}_{(\mathcal{D}, f)}(\text{ERM}_\mathcal{H}(S)) > \varepsilon\}) \leq \delta$$

• Let $\mathcal{H}_B$ be the set of “bad” hypotheses,

$$\mathcal{H}_B = \{h \in \mathcal{H} : \mathcal{L}_{(\mathcal{D}, f)}(h) > \varepsilon\}$$

• Let $M$ be the set of “misleading” samples,

$$M = \{S|_x : \exists h \in \mathcal{H}_B, \mathcal{L}_S(h) = 0\}$$

• Observe:

$$\{S|_x : \mathcal{L}_{(\mathcal{D}, f)}(\text{ERM}_\mathcal{H}(S)) > \varepsilon\} \subseteq M = \bigcup_{h \in \mathcal{H}_B} \{S|_x : \mathcal{L}_S(h) = 0\}$$
Proof (Cont.)

Lemma (Union bound)

For any two sets $A, B$ and a distribution $\mathcal{D}$ we have

$$
\mathcal{D}(A \cup B) \leq \mathcal{D}(A) + \mathcal{D}(B).
$$
Proof (Cont.)

Lemma (Union bound)

For any two sets $A, B$ and a distribution $\mathcal{D}$ we have

$$\mathcal{D}(A \cup B) \leq \mathcal{D}(A) + \mathcal{D}(B) .$$

• We have shown:

$$\{S|_x : \mathcal{L}(\mathcal{D}, f)(\text{ERM}_\mathcal{H}(S)) > \varepsilon \} \subseteq \bigcup_{h \in \mathcal{H}_B} \{S|_x : \mathcal{L}_S(h) = 0\}$$

• Therefore, using the union bound

$$\mathcal{D}^m(\{S|_x : \mathcal{L}(\mathcal{D}, f)(\text{ERM}_\mathcal{H}(S)) > \varepsilon \})$$

$$\leq \sum_{h \in \mathcal{H}_B} \mathcal{D}^m(\{S|_x : \mathcal{L}_S(h) = 0\})$$

$$\leq |\mathcal{H}_B| \max_{h \in \mathcal{H}_B} \mathcal{D}^m(\{S|_x : \mathcal{L}_S(h) = 0\})$$
Proof (Cont.)

• Observe:

\[ \mathcal{D}^m(\{S \mid x : \mathcal{L}_S(h) = 0\}) = (1 - \mathcal{L}_{\mathcal{D},f}(h))^m \]

• If \( h \in \mathcal{H}_B \) then \( \mathcal{L}_{\mathcal{D},f}(h) > \varepsilon \) and therefore

\[ \mathcal{D}^m(\{S \mid x : \mathcal{L}_S(h) = 0\}) < (1 - \varepsilon)^m \]

• We have shown:

\[ \mathcal{D}^m(\{S \mid x : \mathcal{L}_{(\mathcal{D},f)}(\text{ERM}_{\mathcal{H}}(S)) > \varepsilon\}) < |\mathcal{H}_B| (1 - \varepsilon)^m \]

• Finally, using \( 1 - \varepsilon \leq e^{-\varepsilon} \) and \( |\mathcal{H}_B| \leq |\mathcal{H}| \) we conclude:

\[ \mathcal{D}^m(\{S \mid x : \mathcal{L}_{(\mathcal{D},f)}(\text{ERM}_{\mathcal{H}}(S)) > \varepsilon\}) < |\mathcal{H}| e^{-\varepsilon m} \]

• The right-hand side would be \( \leq \delta \) if \( m \geq \frac{\log(|\mathcal{H}|/\delta)}{\varepsilon} \). \( \square \)
Illustrating the use of the union bound

- Each point is a possible sample $S|_x$. Each colored oval represents misleading samples for some $h \in \mathcal{H}_B$. The probability mass of each such oval is at most $(1 - \varepsilon)^m$. But, the algorithm might err if it samples $S|_x$ from any of these ovals.
PAC learning

Definition (PAC learnability)

A hypothesis class \( \mathcal{H} \) is PAC learnable if there exists a function \( m_{\mathcal{H}} : (0, 1)^2 \rightarrow \mathcal{N} \) and a learning algorithm with the following property:

- for every \( \varepsilon, \delta \in (0, 1) \)
- for every distribution \( \mathcal{D} \) over \( \mathcal{X} \), and for every labeling function \( f : \mathcal{X} \rightarrow \{0, 1\} \)

when running the learning algorithm on \( m \geq m_{\mathcal{H}}(\varepsilon, \delta) \) i.i.d. examples generated by \( \mathcal{D} \) and labeled by \( f \), the algorithm returns a hypothesis \( h \) such that, with probability of at least \( 1 - \delta \) (over the choice of the examples), \( \mathcal{L}_{(\mathcal{D}, f)}(h) \leq \varepsilon \).
PAC learning

Definition (PAC learnability)

A hypothesis class $\mathcal{H}$ is PAC learnable if there exists a function $m_\mathcal{H} : (0, 1)^2 \to \mathcal{N}$ and a learning algorithm with the following property:

- for every $\epsilon, \delta \in (0, 1)$
- for every distribution $D$ over $\mathcal{X}$, and for every labeling function $f : \mathcal{X} \to \{0, 1\}$

when running the learning algorithm on $m \geq m_\mathcal{H}(\epsilon, \delta)$ i.i.d. examples generated by $D$ and labeled by $f$, the algorithm returns a hypothesis $h$ such that, with probability of at least $1 - \delta$ (over the choice of the examples), $\mathcal{L}(D, f)(h) \leq \epsilon$.

$m_\mathcal{H}$ is called the sample complexity of learning $\mathcal{H}$
PAC learning

Leslie Valiant, Turing award 2010

*For transformative contributions to the theory of computation, including the theory of probably approximately correct (PAC) learning, the complexity of enumeration and of algebraic computation, and the theory of parallel and distributed computing.*
What is learnable and how to learn?

- We have shown:

**Corollary**

Let $\mathcal{H}$ be a finite hypothesis class.

- $\mathcal{H}$ is PAC learnable with sample complexity $m_\mathcal{H}(\varepsilon, \delta) \leq \frac{\log(|\mathcal{H}|/\delta)}{\varepsilon}$

- This sample complexity is obtained by using the $\text{ERM}_\mathcal{H}$ learning rule
What is learnable and how to learn?

• We have shown:

Corollary

Let $\mathcal{H}$ be a finite hypothesis class.

• $\mathcal{H}$ is PAC learnable with sample complexity $m_{\mathcal{H}}(\epsilon, \delta) \leq \frac{\log(|\mathcal{H}|/\delta)}{\epsilon}$

• This sample complexity is obtained by using the ERM$_{\mathcal{H}}$ learning rule

• What about infinite hypothesis classes?

• What is the sample complexity of a given class?

• Is there a generic learning algorithm that achieves the optimal sample complexity?
What is learnable and how to learn?

Chervonenkis
The fundamental theorem of statistical learning:
• The sample complexity is characterized by the VC dimension
• The ERM learning rule is a generic (near) optimal learner

Vapnik