CSC 580 Principles of Machine Learning

# 17 Learning theory

**Chicheng Zhang**

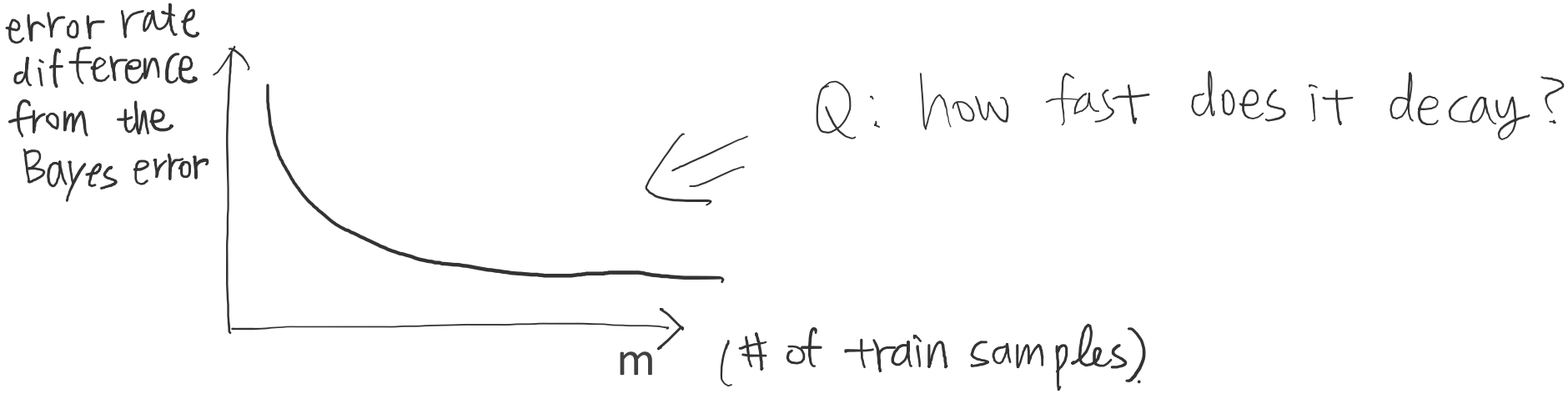
**Department of Computer Science**

THE UNIVERSITY OF ARIZONA

*slides credit: built upon CSC 580 Fall 2021 lecture slides by Kwang-Sung Jun

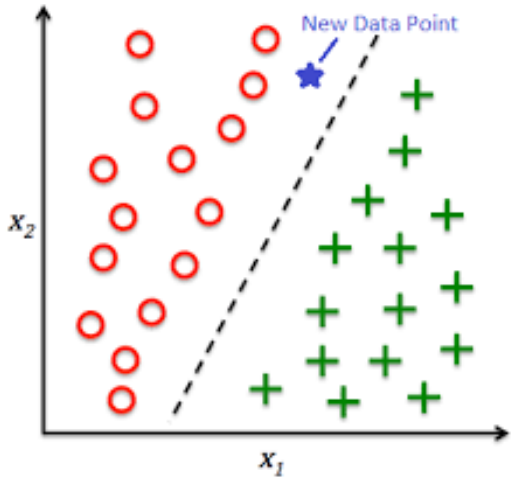
# Learning theory

- An attempt to understand machine learning at a fundamental level, with minimal assumptions
- Why?
  - Provides mathematical explanation on how many samples are needed to achieve the target error rate (e.g., 5% error rate).
  - Provides the **fundamental limits** of any ML algorithms (e.g., statements like “without 100 samples, you are not guaranteed to have ≤ 5% error rate”).
  - What characterizes the **difficulty** of a particular ML problem? (e.g., for kernel regression, the slope of the learning curve depends on the ‘effective dimension’)




(* examples above is just to give you a sense; rigorous statements are more involved.)

# The setup: Classification




- $\mathcal{X}$: the space of instances (e.g., $\mathbb{R}^d$)
- $\mathcal{Y}$ : the space of labels (e.g., $\{-1,1\}$)
- $\mathcal{D}$: distribution of (instance, label)
- $S \sim \mathcal{D}^m$ ($D^m := D \times \cdots \times D$).
- Hypothesis class $\mathcal{H}$ (= function class = <u>concept</u> class): each $h$ is a mapping: $\mathcal{X} \to \mathcal{Y}$
  - $\mathcal{H}$ is usually of infinite cardinality. E.g., linear classifiers.
  - In this lecture, we mostly handle the case of $|\mathcal{H}| < \infty$. E.g., discretized linear classifiers.
- $\mathrm{err}(h)$: the generalization error.
- $\widehat{\mathrm{err}}(h)$: the training error.

(↓ automatically implies that there is no label noise)

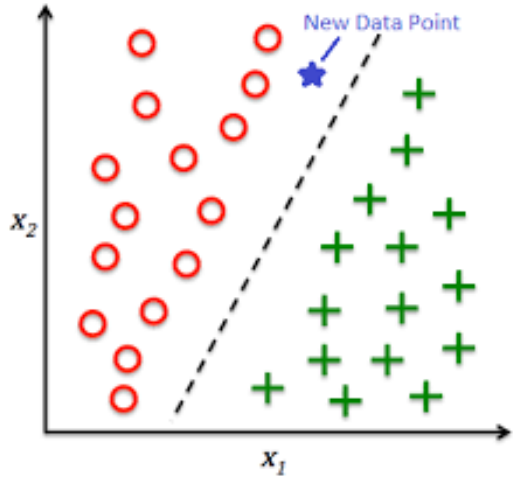
- The <u>realizable</u> case: there exists $h^* \in \mathcal{H}$ such that $\mathrm{err}(h^*) = 0$
- The <u>agnostic</u> case: realizability may or may not be true

https://towardsdatascience.com/supervised-learning-basics-of-classification-and-main-algorithms-c16b06806cd3

# Probably Approximately Correct (PAC) framework

- Suppose an ML algorithm A returns $\hat{h} \in \mathcal{H}$ using a training set $S \sim \mathcal{D}^m$
- Goal: We'd like to say something like "if you have $m$ samples, the error rate of $\hat{h}$ will be at most 1/m" and like this to be true **at any case**.

- Turns out, this is hard to say this "**at any case**". Why?
- We may be unlucky that $S$ is unrepresentative of $\mathcal{D}$



# Probably Approximately Correct (PAC) framework.

- Suppose an ML algorithm A returns $\hat{h} \in \mathcal{H}$ using a train set $S \sim \mathcal{D}^m$
- Goal: We'd like to say something like "if you have $m$ samples, the error rate of $\hat{h}$ will be at most 1/m" and like this to be true **at any case**.

- Turns out, this is hard to say this "**at any case**". Why?
- **Amended goal**: We'd like to say something like
  "if you have $m$ samples, the error rate of A will be at most 1/m, with probability at least $.95$".

approximately correct

probably

- This kind of guarantee is called: Probably approximately correct (PAC) guarantee.

# PAC framework

**[Def]** Let $\mathcal{H}$ be a hypothesis space. Let $\mathcal{A}$ be an algorithm that outputs $\hat{h}_m$.

We say that $\mathcal{A}$ '**PAC-learns $\mathcal{H}$**' with **sample complexity** function $m(\epsilon, \delta)$

$\Leftrightarrow$ For any distribution $\mathcal{D}$ realizable with respect to hypothesis space $\mathcal{H}$, $\epsilon > 0, \delta > 0$, we have that

$$m \geq m(\epsilon, \delta) \;\Rightarrow\; P\left(err\left(\hat{h}_m\right) \leq \epsilon\right) \geq 1 - \delta$$

$\epsilon$: target error rate

$\delta$: target failure rate

**[Thm]** If $|\mathcal{H}| < \infty$, then there exists an algorithm $\mathcal{A}$ that PAC-learns $\mathcal{H}$ with $m(\epsilon, \delta) = \frac{\ln|\mathcal{H}| + \ln\left(\frac{1}{\delta}\right)}{\epsilon}$.

- This means: if we fix $m$, then w.p. $\geq 1 - \delta$, $\text{err}\left(\hat{h}_m\right) \leq \frac{\ln|\mathcal{H}| + \ln\left(\frac{1}{\delta}\right)}{m}$
- Tip: Don't focus too much on $\ln\left(\frac{1}{\delta}\right)$; if $\delta = 0.05$ then $\ln\left(\frac{1}{\delta}\right) \approx 3$

sample complexity and the error bound are reciprocal: if you have one, you can derive the other.

# Proof of the theorem

**[Thm]** If $|\mathcal{H}| < \infty$, then there exists an algorithm $\mathcal{A}$ that PAC-learns $\mathcal{H}$ with $m(\epsilon, \delta) = \frac{\ln|\mathcal{H}| + \ln\left(\frac{1}{\delta}\right)}{\epsilon}$.

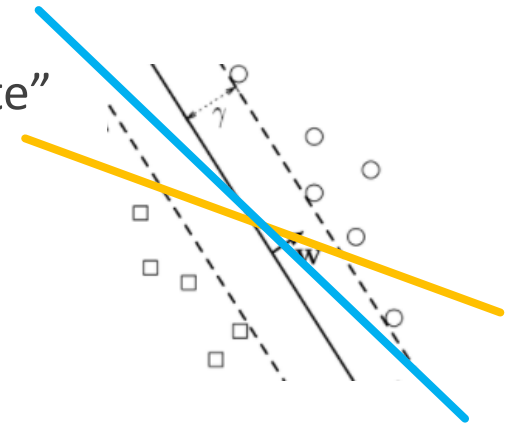
**(The proof)** By construction: Let $\mathcal{A}$ be the ERM:

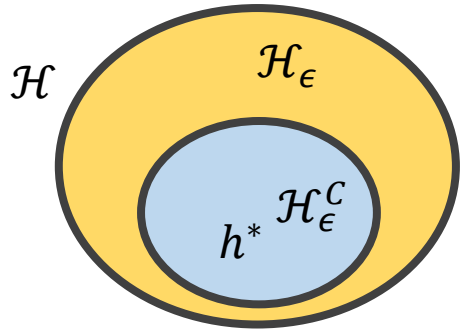
$$\hat{h}_m = \arg\min_{h \in \mathcal{H}} \left[ \widehat{\mathrm{err}}(h) \coloneqq \frac{1}{m} \sum_{i=1}^{m} 1\{h(x_i) \neq y_i\} \right]$$

(break ties arbitrarily)

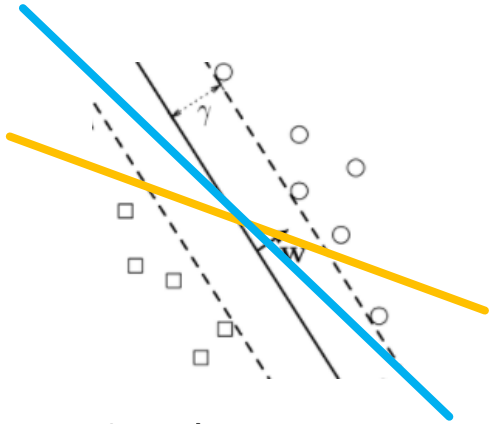
- Observe: $\mathcal{D}$ is realizable => $\widehat{\mathrm{err}}(h^*) = 0$ (no noise in training data) => $\widehat{\mathrm{err}}\left(\hat{h}_m\right) = 0$

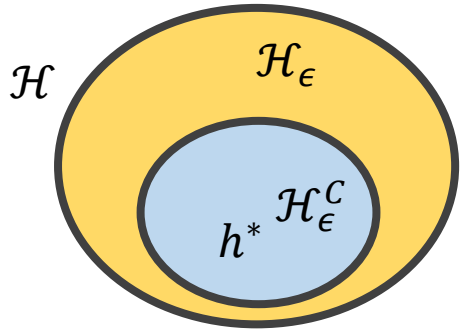
- Define: $\mathcal{H}_\epsilon = \{h \in \mathcal{H} : \mathrm{err}(h) \geq \epsilon\}$
- Intuition: we will show that our training examples can "invalidate" all classifiers in $\mathcal{H}_\epsilon$, with high probability



# Proof of the theorem





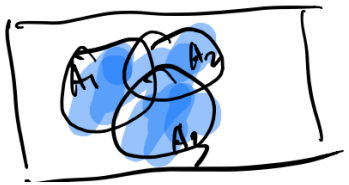

- Define event

$$E = \{\forall h \in \mathcal{H}_\epsilon : \widehat{\mathrm{err}}(h) > 0\}$$

(in words, all high-error classifiers are invalidated by the training data)

- Note: $E$ happens => $\hat{h}_m \notin \mathcal{H}_\epsilon$ => $\mathrm{err}(\hat{h}_m) \leq \epsilon$

$$P\left(\bigcup_{i=1}^{n} A_i\right) \leq \sum_{i=1}^{n} P(A_i)$$




- Thus, it suffices to show $\mathrm{P}(E) \geq 1 - \delta$. Why is this true?

$$\mathrm{P}(E^C) = P(\exists h \in \mathcal{H}_\epsilon : \widehat{\mathrm{err}}(h) = 0)$$

$$\leq \sum_{h \in \mathcal{H}_\epsilon} P(\widehat{\mathrm{err}}(h) = 0) \qquad \text{(union bound)}$$

$$= \sum_{h \in \mathcal{H}_\epsilon} \prod_{i=1}^{m} P(h(x_i) = y_i) \qquad \text{(independence of examples)}$$

$$= \sum_{h \in \mathcal{H}_\epsilon} \left(1 - \mathrm{err}(h)\right)^m \qquad \text{(definition of } \mathrm{err}(h)\text{)}$$

$$\leq |\mathcal{H}|(1-\epsilon)^m \qquad (\mathrm{err}(h) \geq \epsilon, |\mathcal{H}_\epsilon| \leq |\mathcal{H}|)$$

$$\leq |\mathcal{H}| e^{-m\epsilon} \leq \delta \qquad (1 + x \leq e^x, m \geq m(\epsilon, \delta))$$

# Application 1

example:


- Learning conjunctions of Boolean literals.
  - each instance $x = (x(1), \ldots, x(d))$ has d Boolean features
  - Example hypothesis: $x(1) \wedge x(2) \wedge \neg x(4)$
  - Recall realizability: the label is determined by some $h^* \in \mathcal{H}$.
  - $|\mathcal{H}| = 3^d$ (each variable can be either present with no negation, present with negation, or not present)

- Q: How many training examples suffice to ensure that with probability $\geq 0.99$, the ERM will return a hypothesis with error $\leq 0.05$?

- A: $m \geq \frac{1}{.05}\left(\ln\left(3^d\right) + \ln\left(\frac{1}{.01}\right)\right)$
  - If d=10, then m $\geq$ 312
  - If d=100, then m $\geq$ 2290

# Application 2

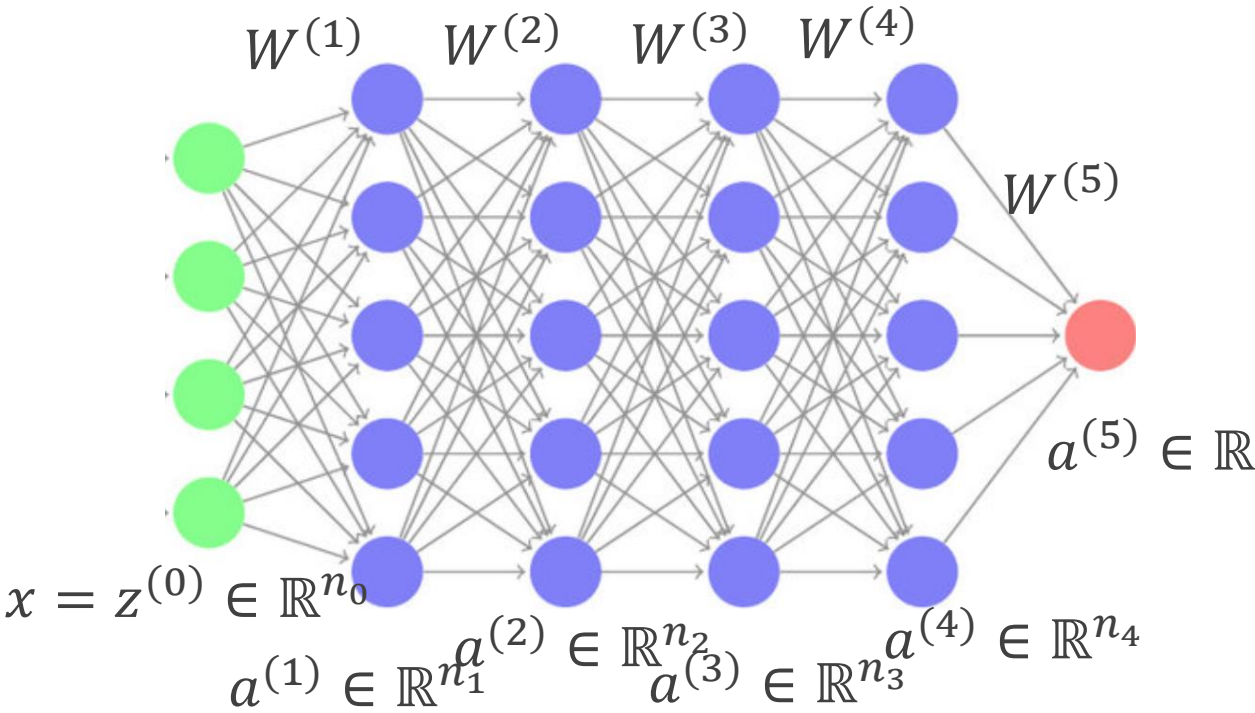
- Learning neural networks with a fixed architecture
- $\mathcal{H} = \{L$ –layer MLPs with floating-point weight with $n_l$ nodes at layer $l\}$
- #parameters of $h \in \mathcal{H}$:s

$$\sum_{l=1}^{L} n_{l-1}\, n_l$$

- Every neural net can be represented by

$$32\sum_{l=1}^{L} n_{l-1}\, n_l \text{ bits}$$

$$\Rightarrow \log|\mathcal{H}| \leq 32\sum_{l=1}^{L} n_{l-1}\, n_l$$




- Training sample size $m \geq \Omega\left(\sum_{l=1}^{L} n_{l-1}\, n_l\right)$ ensures that ERM returns a low-error model

# The Agnostic PAC framework

- Realizable setting: we have assumed that there exists a hypothesis with zero error (denoted by $h^*$) is in the hypothesis class $\mathcal{H}$.
  - Certainly, not realistic.

- Agnostic setting
  - $h^* = \arg\min_{h \in \mathcal{H}} \mathrm{err}(h)$ does not necessarily has zero error.
  - true or false: "this means that the Bayes error is not zero"?

**[Def]** Let $\mathcal{A}$ be an algorithm that takes in $m$ samples and outputs a classifier $\hat{h}_m$ from a hypothesis class $\mathcal{H}$. Then, we say $\mathcal{A}$ '<u>agnostically</u> PAC-learns hypothesis class $\mathcal{H}$' with **<u>sample complexity</u>** function $m(\epsilon, \delta)$.

$\Leftrightarrow$ For any distribution $\mathcal{D}$ and values $\epsilon > 0$, and $\delta > 0$, we have that

$$m \geq m(\epsilon, \delta) \;\Rightarrow\; P\left(\mathrm{err}\left(\hat{h}_m\right) - \min_{h \in \mathcal{H}} \mathrm{err}(h) \leq \epsilon\right) \geq 1 - \delta$$

# Background: Hoeffding's bound

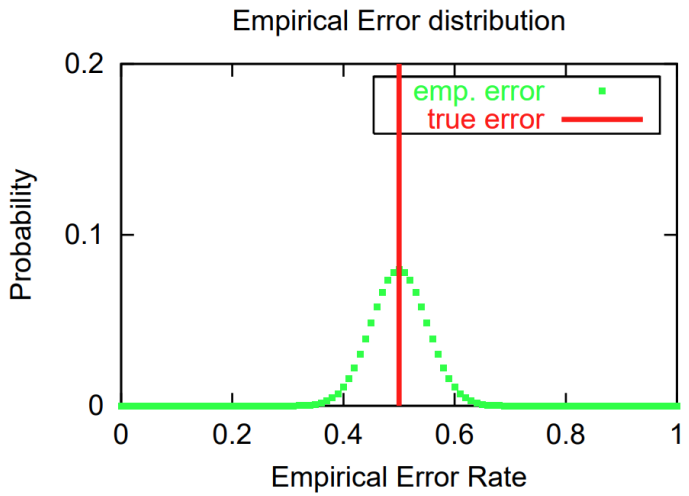
(Thm) Let $X_1, \ldots, X_m$ be i.i.d. random variables from a distribution $\mathcal{D}$ that is supported on [0,1] with mean $\mu$. Then,

$$P\left(\left|\mu - \frac{1}{m}\sum_{i=1}^{m} X_i\right| \geq \epsilon\right) \leq 2e^{-2m\epsilon^2}$$

- equivalently, $P\left(\left|\mu - \frac{1}{m}\sum_{i=1}^{m} X_i\right| < \epsilon\right) \geq 1 - 2e^{-2m\epsilon^2}$

- Exercise: Compute a confidence bound on $\mu$ that holds with probability at least $1 - \delta$.

- Application: let $h$ be a hypothesis. Set $X_i := 1\{h(x_i) \neq y_i\}$
  - Then, $\mu = \mathrm{err}(h)$ and $\frac{1}{m}\sum_{i=1}^{m} X_i = \widehat{\mathrm{err}}(h)$.
  - $P(|\mathrm{err}(h) - \widehat{\mathrm{err}}(h)| < \epsilon) \geq 1 - 2e^{-2m\epsilon^2}$
  - In other words, w.p. $1 - \delta$, $|\mathrm{err}(h) - \widehat{\mathrm{err}}(h)| \leq \sqrt{\frac{\ln 2/\delta}{2m}}$

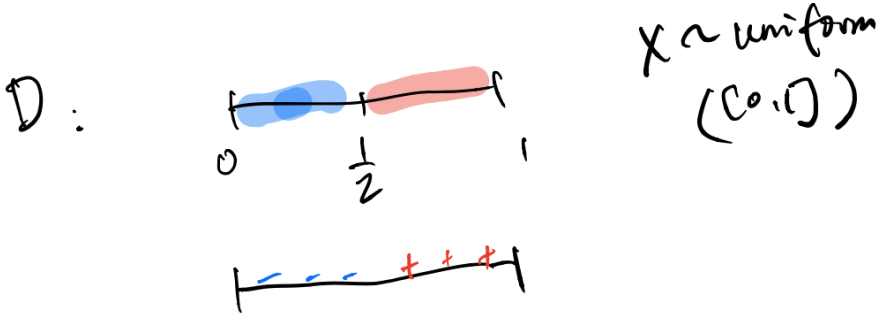


# Applications of Hoeffding's bound

- Does Hoeffding's bound imply that for any trained classifier $\hat{h}_m$, w.p. $1-\delta$,

$$\mathrm{err}(\hat{h}_m) - \widehat{\mathrm{err}}(\hat{h}_m) \leq \left|\mathrm{err}(\hat{h}_m) - \widehat{\mathrm{err}}(\hat{h}_m)\right| \leq \sqrt{\frac{\ln 2/\delta}{2m}} \ ?$$

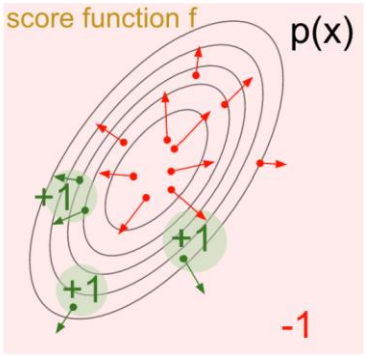
- What if $\hat{h}_m$ is a classifier that memorizes data? (e.g. it outputs default label + for unseen examples)
  - Example: learning a threshold in [0,1] interval
  - $\widehat{\mathrm{err}}(\hat{h}_m) = 0$
  - $\mathrm{err}(\hat{h}_m) = 1/2$

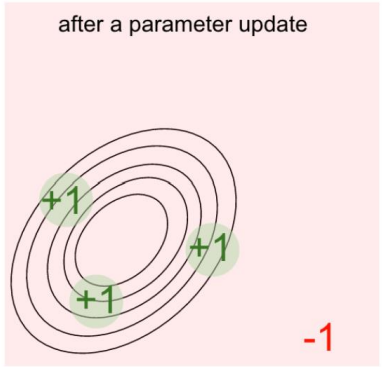



- Hoeffding's bound does not apply to $\hat{h}_m$!
  - Key reason: $\hat{h}_m$ is *not* chosen before seeing the training examples, violating the assumption
- How to analyze the performance of $\hat{h}_m$?
  - A common approach: assume that $\hat{h}_m \in \mathcal{H}$, and bound $|\mathrm{err}(h) - \widehat{\mathrm{err}}(h)|$ for all $h \in \mathcal{H}$

# Assigned reading quiz last time

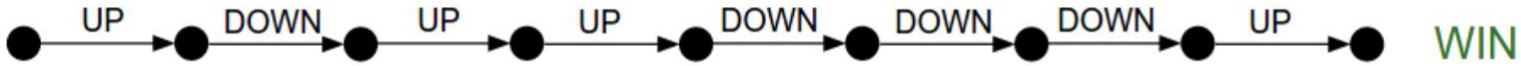
- Andrej Karpathy, “Deep Reinforcement learning: Pong from Pixels”
- What reinforcement learning (RL) method does the author use to train a game-playing agent? What is its main idea?
  - Policy gradient method




- What are some differences between human and this RL agent in solving the game
  - Human can start playing reasonably without receiving rewards
  - Human incorporate prior knowledge, e.g. intuitive physics

- What is the “credit assignment problem”? Why is this a challenge in RL?



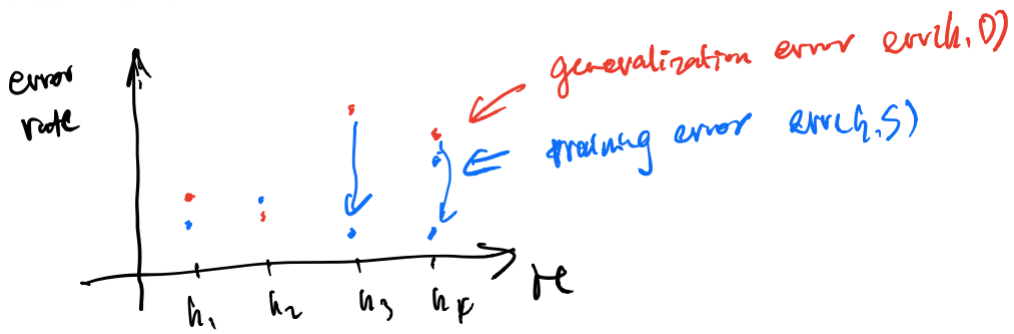
# Uniform convergence via Hoeffding's inequality

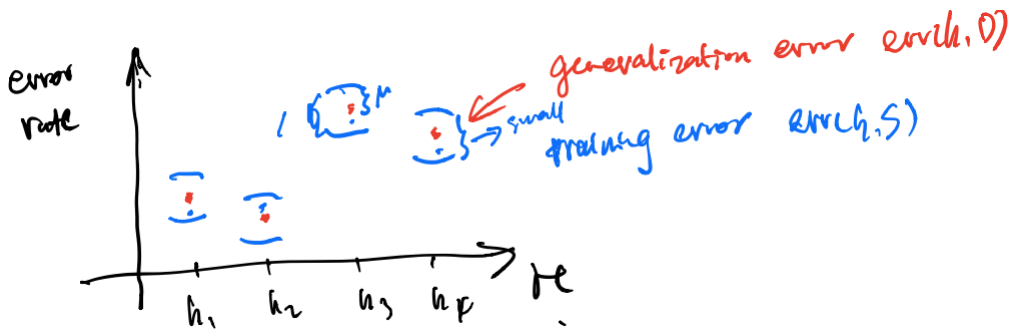
- **Thm (uniform convergence)**: If $|\mathcal{H}| < \infty$, then with training sample size

$m \geq m(\epsilon, \delta) = \frac{8\left(\ln|\mathcal{H}| + \ln\left(\frac{2}{\delta}\right)\right)}{\epsilon^2}$, we have that w.p. $\geq 1 - \delta$, the following happens:

$$\text{for all } h \in \mathcal{H}, |\text{err}(h) - \widehat{\text{err}}(h)| \leq \epsilon/2$$

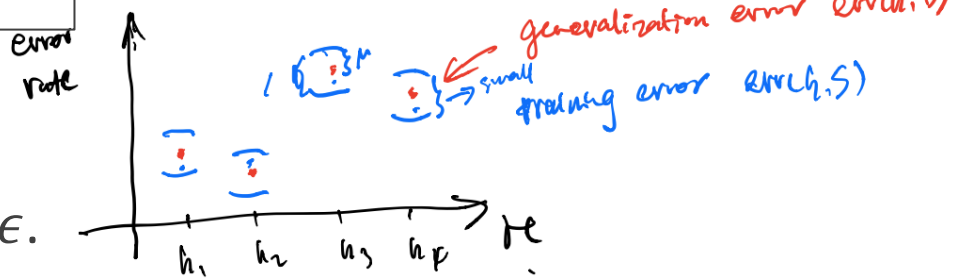
- Why do we care about this? This guarantees that the ERM classifier $\hat{h}_m := \arg\min_{h \in \mathcal{H}} \widehat{\text{err}}(h)$ has good generalization error (see next slide), establishing agnostic PAC learnability



# Uniform convergence => agnostic PAC learning

$$h^* = \arg\min_{h \in \mathcal{H}} \mathrm{err}(h)$$




**Lemma**: If it holds that

$$\text{for all } h \in \mathcal{H}, |\mathrm{err}(h) - \widehat{\mathrm{err}}(h)| \le \epsilon/2,$$

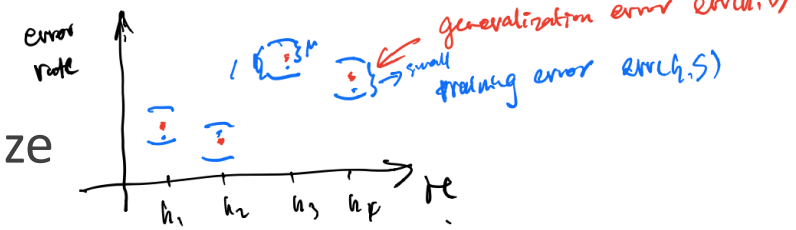
then $\hat{h}_m := \arg\min_{h \in \mathcal{H}} \widehat{\mathrm{err}}(h)$ satisfies that $\mathrm{err}(\hat{h}_m) - \mathrm{err}(h^*) \le \epsilon$.

Proof:
$$\mathrm{err}(\hat{h}_m) \le \widehat{\mathrm{err}}(\hat{h}_m) + \epsilon/2 \qquad \text{(Assumption w/ } h = \hat{h}_m\text{)}$$
$$\le \widehat{\mathrm{err}}(h^*) + \epsilon/2 \qquad \text{(Optimality of ERM)}$$
$$\le (\mathrm{err}(h^*) + \epsilon/2) + \epsilon/2 \qquad \text{(Assumption w/ } h = h^*\text{)}$$
$$= \mathrm{err}(h^*) + \epsilon$$

**Thm (sample complexity of agnostic PAC learning):** If $|\mathcal{H}| < \infty$, then ERM agnostic PAC-learns $\mathcal{H}$ with $m(\epsilon, \delta) = \frac{8\left(\ln|\mathcal{H}| + \ln\left(\frac{2}{\delta}\right)\right)}{\epsilon^2}$.

Proof sketch: $m \ge m(\epsilon, \delta) \Rightarrow$ Uniform convergence $\Rightarrow$ ERM has excess error with high prob.

# Establishing uniform convergence




- **Thm (uniform convergence)**: If $|\mathcal{H}| < \infty$, then with training sample size $m \geq m(\epsilon, \delta) = \frac{8\left(\ln|\mathcal{H}| + \ln\left(\frac{2}{\delta}\right)\right)}{\epsilon^2}$, we have that w.p. $\geq 1 - \delta$, the following happens:

$$\text{for all } h \in \mathcal{H}, |\text{err}(h) - \widehat{\text{err}}(h)| \leq \epsilon/2$$

Proof: define $E = \{\forall h \in \mathcal{H}: |\text{err}(h) - \widehat{\text{err}}(h)| \leq \epsilon/2\}$; suffices to show $\text{P}(E) \geq 1 - \delta$

$$\begin{aligned}
\text{P}(E^C) &= P(\exists h \in \mathcal{H}: |\text{err}(h) - \widehat{\text{err}}(h)| > \epsilon/2) \\
&\leq \textstyle\sum_{h \in \mathcal{H}} P(|\text{err}(h) - \widehat{\text{err}}(h)| > \epsilon/2) && \text{(union bound)} \\
&\leq \textstyle\sum_{h \in \mathcal{H}} 2e^{-\frac{m\epsilon^2}{2}} && \text{(Hoeffding)} \\
&= 2|\mathcal{H}|e^{-\frac{m\epsilon^2}{2}} && \text{(algebra)} \\
&\leq \delta && (m \geq m(\epsilon, \delta))
\end{aligned}$$

# Discussion

All these results are 'worst-case' bounds!

**[Thm]** If $|\mathcal{H}| < \infty$, then there exists an algorithm $\mathcal{A}$ that PAC-learns $\mathcal{H}$ with $m(\epsilon, \delta) = \frac{\ln|\mathcal{H}| + \ln\left(\frac{1}{\delta}\right)}{\epsilon}$.

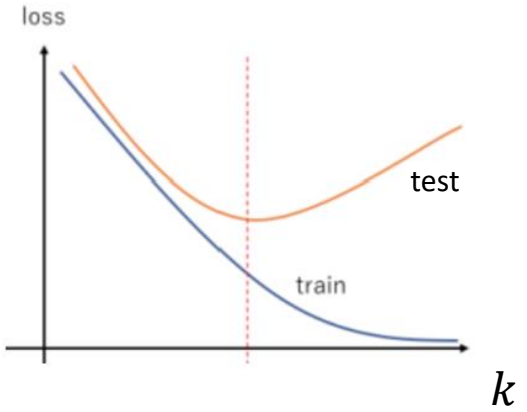
**[Thm]** If $|\mathcal{H}| < \infty$, then there exists an algorithm $\mathcal{A}$ that agnostically PAC-learns $\mathcal{H}$ with $m(\epsilon, \delta) = \frac{8\left(\ln|\mathcal{H}| + \ln\left(\frac{2}{\delta}\right)\right)}{\epsilon^2}$. I.e., $m \geq m(\epsilon, \delta) \Rightarrow P\left(\text{err}\left(\hat{h}_m\right) - \min_{h \in \mathcal{H}} \text{err}(h) \leq \epsilon\right) \geq 1 - \delta$

Remember: performance on individual $\mathcal{D}$ can be much better.

# A learning-theoretic view of model selection

- **Problem** (model selection): given a nested collection of hypotheses classes $\mathcal{H}_1 \subseteq \cdots \subseteq \mathcal{H}_K$ and a training sample $S$, how to learn a classifier from $\cup_{k=1}^{K} \mathcal{H}_k$ that has low generalization error?

- Challenge:
  - Can do ERM with $\mathcal{H}_k$ alone, but what $k$ to choose?
  - Smaller $k$ => $\min_{h \in \mathcal{H}_k} \mathrm{err}(h)$ is large => underfitting
  - Larger $k$ => $\mathcal{H}_k$ too complex => overfitting




- Suggestion from Learning theory: choose $\hat{h} \in \cup_{k=1}^{K} \mathcal{H}_k$ with the *smallest upper bound* of generalization error
  - $(\hat{h}, \hat{k}) = \mathrm{argmin}_{k \in \{1,\ldots,K\}, h \in \mathcal{H}_k} \underbrace{\left( \widehat{\mathrm{err}}(h) + \sqrt{\frac{\ln K|\mathcal{H}_k|/\delta}{2m}} \right)}_{\text{Upper bounds } \mathrm{err}(h) \text{ w.p. } 1-\delta}$

# A learning-theoretic view of model selection

- Suggestion from Learning theory: choose $\hat{h} \in \cup_{k=1}^{K} \mathcal{H}_k$ with the *smallest upper bound* of generalization error

$$(\hat{h}, \hat{k}) = \text{argmin}_{k \in \{1,\ldots,K\}, h \in \mathcal{H}_k} \left( \widehat{\text{err}}(h) + \sqrt{\frac{\ln K|\mathcal{H}_k|/\delta}{2m}} \right)$$

"Occam's Razor" – simpler explanations are preferred

- Bound minimization – a sometimes useful heuristic
- Intuition: choosing model conservatively; complexity *regularization*
- Also known as the Structural Risk Minimization (SRM) principle

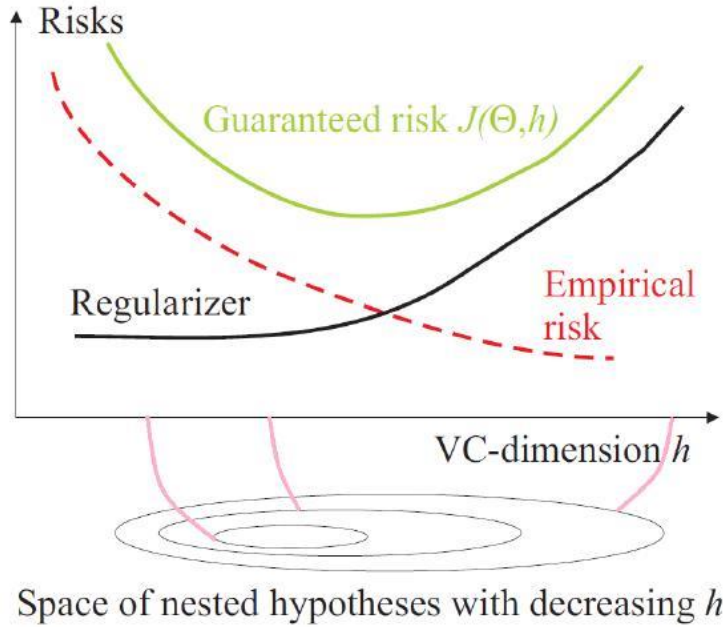



Figure 7: Structure Risk Minimization for Model Selection [11]

- **Thm:** the SRM output $\hat{h}$ has generalization error bounds competitive with ERM with $\mathcal{H}_k$, for all $k$ simultaneously.

# What if the hypothesis space is not finite?

- This is highly nontrivial, given techniques we learned so far.

- Turns out, even if $|\mathcal{H}| = \infty$, there is an '**effective capacity**' that replaces $\log |\mathcal{H}|$.
  - The 'right' notion of capacity: Vapnik-Chervonenkis (VC) Dimension
  - PAC sample complexity results carry over with $\log |\mathcal{H}|$ replaced with $\mathrm{VC}(\mathcal{H})$

- E.g., for d-dimensional linear classifiers $\mathcal{H}$, $\mathrm{VC}(\mathcal{H}) = O(d)$
  - Its realizable (resp. agnostic) PAC sample complexity $\approx \tilde{O}\left(\frac{d+\ln\left(\frac{1}{\delta}\right)}{\epsilon}\right)$ (resp. $\tilde{O}\left(\frac{d+\ln\left(\frac{1}{\delta}\right)}{\epsilon^2}\right)$)

# Efforts to push learning theory to practice

- Dziugaite & Roy, Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data, 2017

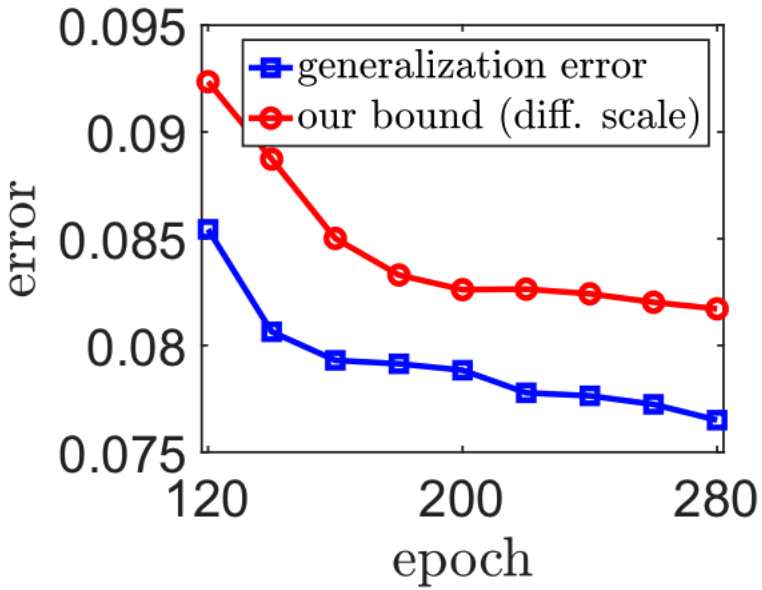
| Experiment | T-600 | T-1200 | T-$300^2$ | T-$600^2$ | T-$1200^2$ | T-$600^3$ | R-600 |
|---|---|---|---|---|---|---|---|
| Train error | 0.001 | 0.002 | 0.000 | 0.000 | 0.000 | 0.000 | 0.007 |
| Test error | 0.018 | 0.018 | 0.015 | 0.016 | 0.015 | 0.013 | 0.508 |
| SNN train error | 0.028 | 0.027 | 0.027 | 0.028 | 0.029 | 0.027 | 0.112 |
| SNN test error | 0.034 | 0.035 | 0.034 | 0.033 | 0.035 | 0.032 | 0.503 |
| PAC-Bayes bound | 0.161 | 0.179 | 0.170 | 0.186 | 0.223 | 0.201 | 1.352 |
| KL divergence | 5144 | 5977 | 5791 | 6534 | 8558 | 7861 | 201131 |
| # parameters | 471k | 943k | 326k | 832k | 2384k | 1193k | 472k |
| VC dimension | 26m | 56m | 26m | 66m | 187m | 121m | 26m |

Table 1: Results for experiments on binary class variant of MNIST. SGD is either trained on (T) true labels or (R) random labels. The network architecture is expressed as $N^L$, indicating $L$ hidden layers with $N$ nodes each. Errors are classification error. The reported VC dimension is the best known upper bound (in millions) for ReLU networks. The SNN error rates are tight upper bounds (see text for details). The PAC-Bayes bounds upper bound the test error with probability $0.965$.

# Bound minimization as learning algorithms?

- There is a school of researchers who believes learning theory can be made practical – it could actually provide a useful upper bound on the test error (**without** looking at the test set)
- Some other researchers disagree.

- At least, they often characterize the '**core quantity**' that matters in reducing test error.
  - e.g., some complicated form of '**norm**' of the weights in the neural networks is claimed to control the generalization error.
  - This can be used to build a practical algorithm – directly use such a norm as a regularizer




http://proceedings.mlr.press/v80/arora18b/arora18b.pdf, "Stronger Generalization Bounds for Deep Nets via a Compression Approach "

# Summary

- Learning theory: the PAC & agnostic PAC learning framework
- Analysis of Empirical Risk Minimization (ERM)
- Capacity measure of hypothesis classes: $\mathrm{VC}(\mathcal{H})$, $\log|\mathcal{H}|$
- Structural Risk Minimization (SRM) for model selection
- Using learning theory to guide practice

# Final review

# General information

- See Piazza @91 about final and project information, if you haven't already!

- Final exam time and venue: Tuesday, Dec 13, 3:30-5:30pm, GS 701
  - You can bring a "cheatsheet" of US letter paper size (two sided) with you
  - The focus will be on Lec 9 onwards (although Lecs 1-8's material may also appear)

- General suggestions:
  - The exam will focus on checking your understanding of basics concepts and ideas, and calculation is likely to be light
  - Review homeworks & midterm solns to make sure you have a solid grasp on the basics

# Lec 9: unsupervised learning

- Clustering
  - The k-means objective function (total quantization error)
  - Lloyd's algorithm
  - Other clustering algorithm (basic understanding)

- PCA
  - Two perspectives: variance maximization and reconstruction error minimization
  - Projection to get low-dimensional representation; reconstruction (HW3, Problem 4)
  - Able to hand-calculate eigenvalues / eigenvectors of small real-symmetric matrices (e.g. 2x2)

# Lec 10: probabilistic ML I: Naïve Bayes

- General recipe for probabilistic ML
  - Specify generative story; estimate the model (MLE); make decisions based on the estimated model

- Generative vs. Discriminative approaches for ML

- MLE for probabilistic models with fully-observed data
  - First, try to write down the data log-likelihood
- The Naïve Bayes model – discrete features, conditional Gaussian features (HW3, Problem 3)
  - Key assumption: the coordinates are conditionally independent given label
  - The MLE
  - The Bayes optimal classifier

# Lec 11: probabilistic ML II: GMMs and EM

- The Gaussian mixture model (GMM)

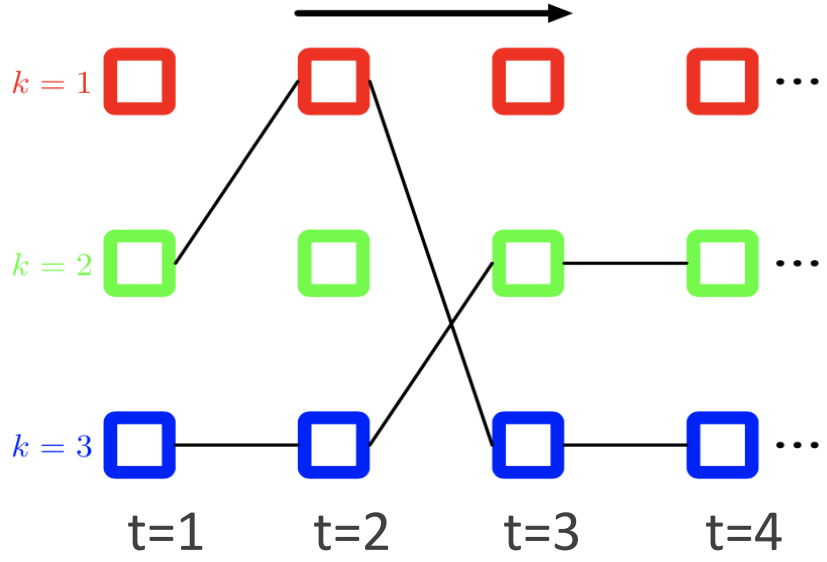
- The EM algorithm
  - What is it used for? When is it used?
  - Main idea: iteratively (E-step) create auxiliary functions that lower bound the likelihood function; (M-step) maximize the auxiliary functions

- EM for GMMs
  - Intuition: clustering by “soft” cluster assignments
  - Connection to Lloyd’s algorithm
  - Able to derive the EM algorithm for other variants of mixture models (e.g. HW3, Problem 2)

# Lec 12: probabilistic ML III: Bayes nets; HMMs

- Bayesian networks
  - Able to do probabilistic reasoning (HW3, Problem 1)
  - Real-world examples of different substructure of Bayesian networks (head-to-head, “explain away”, etc)
  - D-separation: definition
  - Note: D-separation is a sufficient but not necessary condition of conditional independence

- HMMs
  - The generative story
  - Marginal likelihood: forward / backward algorithm
  - Decoding – Viterbi’s algorithm (equiv. to staged shortest path)
  - Learning – the EM algorithm



# Lec 13: Neural networks I

- Basic structure of multi-layer Perceptron
- Loss functions: square loss, cross entropy loss (multiclass classification)
- Stochastic gradient descent; backpropagation for gradient calculation
- Main tool: chain rule for derivatives

- Some useful tricks to train neural networks (basic understanding)
  - Adaptive gradient methods
  - Data augmentation
  - Dropout
  - Batch Normalization
  - …

# Lec 14: Neural networks II: convolutional networks

- Motivation of convolutional neural networks (CNNs)
  - For a local pattern (e.g. edge), have an array of neurons that can detect it in all regions of the input image / sequence
  - Parsimony of convolutional layer: local connection; weight sharing (recall the comparison with fully-connected layer)

- Basic structure
  - Convolutional layer (able to calculate the output shape after passing through a convolution layer, with different strides / filter sizes / padding)
  - Pooling layer
  - Fully-connected layer

- CNN examples (basic understanding):
  - LeNet, AlexNet, VGG, ResNet (skip connections)

# Lec 15: Neural networks III: unsupervised learning

- Autoencoder
  - Main idea: Train an identity mapping in the distribution support, by forcing to learn a low-dimensional “encoding”
  - Nonlinear generalization of PCA
  - How to train?
- Variational Autoencoder (VAE)
  - Main idea: learn the distribution P(x) by introducing latent representation z & modeling P(x|z) using a neural net
  - Training a VAE – three key ideas and their motivations
- Generative Adversarial Network (GAN)
  - Main idea: a game between the generator and discriminator
  - The distribution divergence minimization perspective
  - How to learn the generator? Alternating min / max
- Interpretability of the learned latent representation z

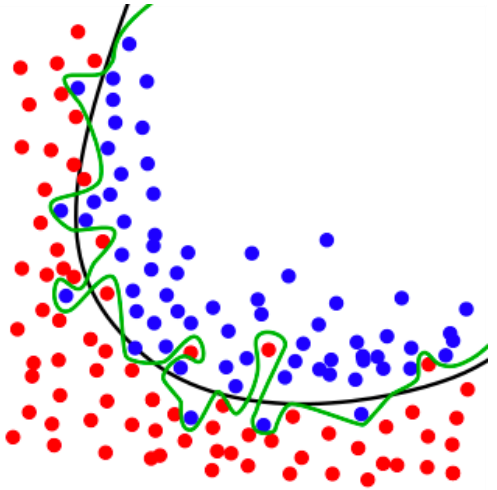
# Lec 16: Reinforcement learning

- The Markov Decision Process (MDP) framework
  - The reward hypothesis
  - Difference between RL and supervised learning
- Policy evaluation – solving Bellman consistency eqn.
  - Gaussian elimination
  - Fixed-point iteration
- Planning (problem setup?)
  - Value iteration – solving Bellman optimality eqn.
  - Policy iteration
- Learning (problem setup?)
  - Unique challenges – exploration
  - Q-learning
    - action value function: Q-function
    - Stochastic approximation view of Q-learning
    - why / how to incorporate function approximation?

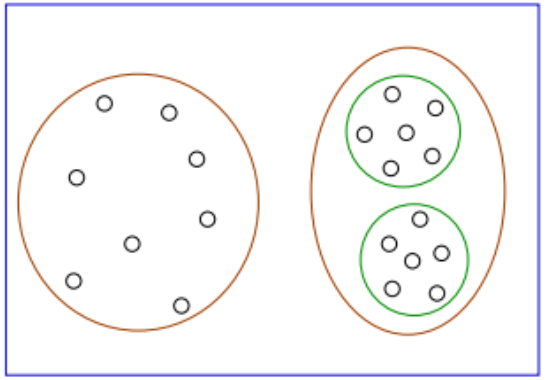
# Lec 17: Learning theory

- The Realizable PAC & Agnostic PAC learning framework
- Basic sample complexity results and their interpretations (recall the examples)
- How can learning theory be used to guide bias-complexity tradeoff?

# Machine learning: a broader view



- Supervised learning -- topics we didn't touch upon:
  - Transfer learning (training/test distribution mismatch) – CIML Chap. 8
  - Fairness in learning – CIML Chap. 8
  - Ensemble methods (e.g. boosting) – CIML Chap. 13
  - Systematic methods for dealing with complex outputs (e.g. multiclass, ranking, structured labels) - CIML Chap. 6, 17



- Unsupervised learning – topics we didn't touch upon:
  - Contrastive learning
  - Using neural network to model sequences (e.g. language models)
  - ....
- Interactive learning paradigms we didn't touch upon:
  - Imitation learning - CIML Chap. 18
  - Active learning (connections to crowdsourcing)
  - Learning with preference-based feedback
  - …