6.7900 Machine Learning (Fall 2023)

Lecture II: Introduction to Supervised Learning and Empirical Risk Minimization

Snapshots of history

2013

Topic
Intro
Intro: Estimation
Intro: Bayesian methods
Regression
Regression: regularization
Classification: probabilistic models
Classification: Bayesian methods
Classification: Support vector machines
Classification: More SVMs, perceptron
Holiday
Kernel methods: classification
Kernel methods: regression
Exam 1: 7:30 - 9:30 PM
Graphical models
Graphical models: message passing
Graphical models: sampling
Graphical models: parameter estimation
Graphical models: structure learning
Graphical models: temporal
Non-parametric: nearest neighbor, trees
Non-parametric: bagging, boosting
Non-parametric: Bayesian
Holiday
Practicality: feature selection, multi-class
Exam 2: 7:30 - 9:30 PM
Topics: unsupervised learning
Topics: reinforcement learning
Topics: deep networks

2019

9/5	Introduction, Overview, Basics
Supervised learning	
9/10	Classification 1: Optimization, Loss Function, Regularization
9/12	Classification 2: SVMs & Kernels, Bayes Classifier, ROC, Logistic Regres
9/17	Classification 3: Naive Bayes, Generalization
9/19	Classification 4: VC Dimension
9/24	Regression 1: Linear / Polynomial Regression, Kernel, Predictive Distrib
9/26	Regression 2: Bayesian, Shrinkage, Regularization and SGD
10/1	Neural networks (feed-forward): Theory, Representation Theorem
10/3	Neural networks: Optimization
10/8	Neural networks: Structured prediction, Language Modeling Word2vec
10/10	Neural networks: Robustness to Dataset Shift
10/15	*Holiday*
Unsupervised learning	
10/17	Dimensionality Reduction: PCA practice
10/22	**Exam 1 7:30 - 9:30 PM**
10/24	Dimensionality Reduction: PCA theory, NMF, t-SNE
10/29	Matrix Estimation
10/31	Clustering: mixture model, K-means
11/5	Topic models
11/7	Variational Learning
11/12	Deep Generative Models
Probabilistic modeling	
11/14	Sampling, MCMC, Gibbs
11/19	Gaussian processes, Using prior knowledge about world
Decision making	
11/21	Acting under Uncertainty, Model Predictive Control
11/26	Markov Decision Processes
11/28	*Holiday*
12/3	Reinforcement Learning, Bandit
12/5	AlphaGoZero and/or Liberatus
12/10	**Project Presentation**

2022

introduction
elements of optimization and MLE
S1: regression, regularization, optimization
S1: regression, bias variance
S1: classification, losses, optimization
S1: classification, ERM, PAC learning
U1: unsupervised learning
U1: graphical models
R1: decision problems, bandits
R1: markov decision problems, RL
Exam #1 (in class)
S2: complexity, regularization, generalization
S2: on-line learning, regret
S2: robustness, adversarial
S2: uncertainty, calibration
S2: neural models, overparameterization
S/U: pre-training, contrastive learning
S/U: co-variate shift, domain adaptation
U2: latent variable models, identifiability
U2: variational inference, VAEs
U2: deep generative models, diffusion
MIT Holiday
R2: deep RL, function approximation
R2: deep RL, policy gradient, robustness
Exam #2 (in class)
Contemporary applications and topics
Contemporary applications and topics

Semester at a glance

how it works (method)

why it works (theory)

Sep 7	Introduction
Sep 12	Supervised learning, formulation, ERM
Sep 14	Regularization, optimization
Sep 19	Linear vs nonlinear, bias v/s variance
Sep 21	PAC intro, finite hypothesis class, infinite hypothesis, VC
Sep 26	On-line learning, regret
Sep 28	Decision problems, bandits
Oct 3	Neural/deep architectures (supervised)
Oct 5	Robustness, stability, adversarial predictions
Oct 12	Uncertainty, conformal prediction
Oct 17	Complexity, generalization
Oct 19	Oct 19: Quiz #1 (in class)
Oct 24	Unsupervised learning, dimensionality reduction
Oct 26	Generative models, auto-regressive
Oct 31	Deep generative models, VAEs, GANs
Nov 2	Flows, Diffusion models
Nov 7	Deep RL, policy gradient, PPO/TRPO
Nov 9	Markov decision problems, Value Based Deep RL, Q-Learning
Nov 14	DQN, Practical Considerations in RL
Nov 16	Covariate shift, domain adaptation
Nov 21	Few-Shot Learning, transfer learning, in-context learning
Nov 28	Self-supervised learning, masking, contrastive
Nov 30	Foundation models
Dec 5	State-of-the-art LLMs (guest lecture?)
Dec 7	Dec 7: Quiz #2 (in class)
Dec 12	Deep RL/AI applications

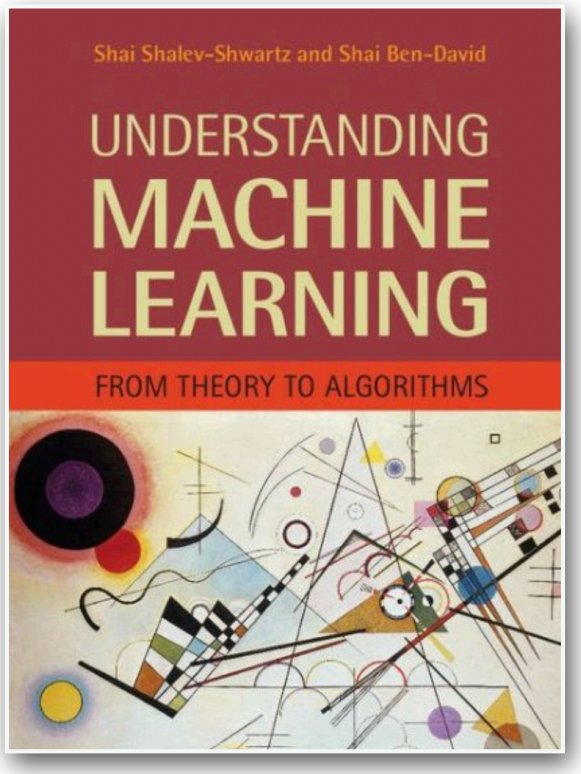
Outline for Today

- Supervised Learning
  - Formal setup
  - Terminology

- Bayes classifier
- Bayes classifier optimality
- Nearest-Neighbor classifier
- Bayes vs. Nearest Neighbor

- Empirical Risk Minimization (ERM)
  - Formal setup
  - Overfitting pitfall
  - Inductive bias
  - Decomposition

References

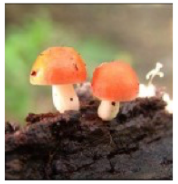


Understanding Machine Learning: From Theory to Algorithms, Shalev-Shwartz and Ben-David; Cambridge University Press, 2014.

- Slides edited from: Suvrit Sra
- Devroye, Györfi, Lugosi. Probabilistic Theory of Pattern Recognition. Springer 1996. Chapter 2 for Bayes, Chapter 5 for Nearest neighbor
- Chen, Shah (2018). Explaining the Success of Nearest Neighbor Methods in Prediction.
- Double descent talk: https://www.youtube.com/watch?v=JS-Bl36aVPs

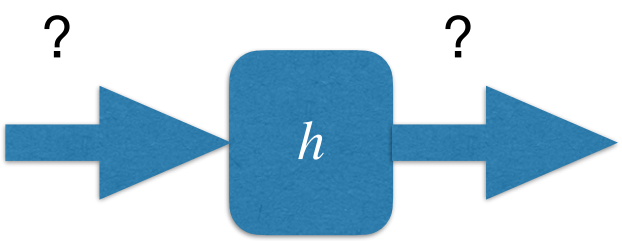
Supervised Learning

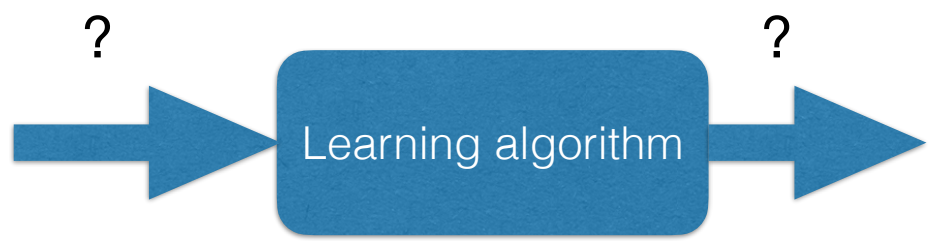
- We can easily illustrate the task by providing a diverse set of examples, exercising the underlying relation between images and categories

	Image	**Category (~ 1K)**
millions of images		mushroom
		flamingo
		keeshond
		cherry
	...	

Classification - Terminology

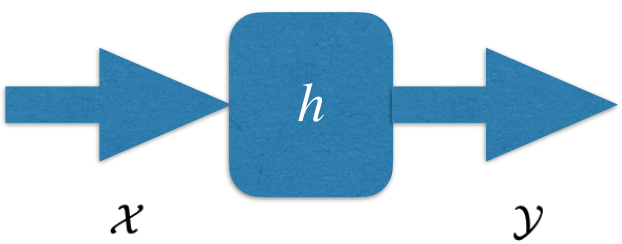
- **Data domain**: An arbitrary set $\mathcal{X}$. Often just $\mathcal{X} = \mathbb{R}^d$ (assuming that the members of $\mathcal{X}$ are represented via feature vectors; some authors write $\Phi(x)$ to emphasize this)
- **Label domain**: A discrete set $\mathcal{Y}$; e.g., $\{0, 1\}$ or $\{-1, 1\}$.
- **Training data**: A finite collection $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ of pairs drawn from $\mathcal{X} \times \mathcal{Y}$
- **Classifier**: A prediction rule $h : \mathcal{X} \to \mathcal{Y}$ (we'll write h_S to emphasize dependence of h on the training data). We call h a *hypothesis*, *prediction rule*, or *classifier*.

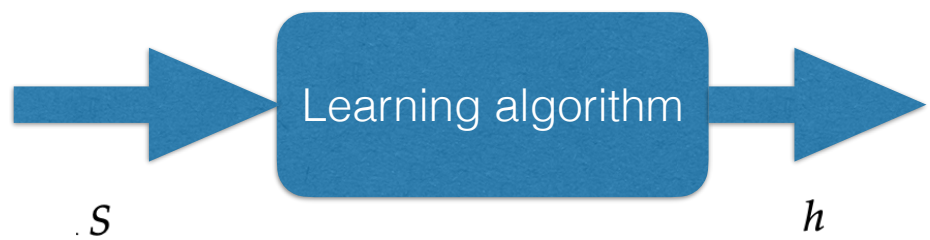




Classification - Terminology

- **Data domain**: An arbitrary set $\mathcal{X}$. Often just $\mathcal{X} = \mathbb{R}^d$ (assuming that the members of $\mathcal{X}$ are represented via feature vectors; some authors write $\Phi(x)$ to emphasize this)
- **Label domain**: A discrete set $\mathcal{Y}$; e.g., $\{0, 1\}$ or $\{-1, 1\}$.
- **Training data**: A finite collection $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ of pairs drawn from $\mathcal{X} \times \mathcal{Y}$
- **Classifier**: A prediction rule $h : \mathcal{X} \to \mathcal{Y}$ (we'll write h_S to emphasize dependence of h on the training data). We call h a *hypothesis*, *prediction rule*, or *classifier*.





What about Regression?

Classification	$y \in \{0,1,\ldots,k\}$
Regression	$y \in \mathbb{R}$

Think of as k categories, not as numbers — there's no sense of ordering.

Most of today's discussion focuses on classification; but the generalization to regression is straightforwardly.

Classification - An assumption

- **Data distribution**: Joint distribution $\mathbb{P}$ on $\mathcal{X} \times \mathcal{Y}$.

Important assumption: $\mathbb{P}$ is fixed but unknown.

We will write (X, Y) to denote a random variable with X taking values in $\mathcal{X}$ and Y taking values in $\mathcal{Y}$.

Think about when and how the above assumption can be violated, and what might we want to do to handle those situations

- If $\mathbb{P}$ is not fixed…
- If $\mathbb{P}$ is known…

Classification - Measuring Success

- **Measuring success**: Error of classifier aka risk aka generalization error:

$$L(h) \equiv L_{\mathbb{P}}(h) := \mathbb{P}(h(X) \neq Y)$$

i.e., the error of classifier h is the probability of randomly choosing a pair $(x, y) \sim \mathbb{P}$ for which $h(x) \neq y$

- **Goal**: Minimize the risk / misclassification error

Question: Which classifier does the "best" job?

Bayes Classifier

- **Class conditional distribution**: Let $\mathcal{Y} = \{0, 1\}$. We define

$$\eta(x) := \mathbb{P}(Y = 1 \mid X = x) = \mathbb{E}[Y \mid X = x].$$

Bayes Classifier

$$h^*(x) := \begin{cases} 1, & \text{if } \eta(x) = \mathbb{P}(Y = 1 | X = x) > \frac{1}{2}, \\ 0, & \text{otherwise.} \end{cases}$$

Theorem (BC optimality). For any classifier $h : \mathbb{R}^d \to \{0, 1\}$, $\mathbb{P}(h^*(X) \neq Y) \leq \mathbb{P}(h(X) \neq Y)$, i.e., h^* is an optimal classifier.

Bayes Classifier Optimality

Hint: Consider $\mathbb{P}(h(X) \neq Y \mid X = x) - \mathbb{P}(h^*(X) \neq Y \mid X = x)$

$$= \eta(x)\left([\![h^*(x) = 1]\!] - [\![h(x) = 1]\!]\right) + (1 - \eta(x))\left([\![h^*(x) = 0]\!] - [\![h(x) = 0]\!]\right)$$
$$= (2\eta(x) - 1)\left([\![h^*(x) = 1]\!] - [\![h(x) = 1]\!]\right)$$

Proof. Given $X = x$, the conditional error probability of any classifier h may be written as:

$$\begin{aligned}
\mathbb{P}\left(h(X) \neq Y | X = x\right) &= 1 - \mathbb{P}\left(Y = h(X) | X = x\right) \\
&= 1 - \left(\mathbb{P}\left(Y = 1, h(X) = 1 | X = x\right) + \mathbb{P}\left(Y = 0, h(X) = 0 | X = x\right)\right) \\
&= 1 - \left([\![h(x) = 1]\!]\mathbb{P}\left(Y = 1 | X = x\right) + [\![h(x) = 0]\!]\mathbb{P}\left(Y = 0 | X = x\right)\right) \\
&= 1 - \left([\![h(x) = 1]\!]\eta(x) + [\![h(x) = 0]\!](1 - \eta(x))\right)
\end{aligned}$$

where $[\![\cdot]\!]$ is the Iverson bracket, i.e. $[\![z]\!] = 1$ if $z =$ 'true' and 0 if $z =$ 'false'. Thus, for every $x \in \mathbb{R}^d$, we have:

$$\begin{aligned}
&\mathbb{P}\left(h(X) \neq Y | X = x\right) - \mathbb{P}\left(h^*(X) \neq Y | X = x\right) \\
&\quad = \eta(x)\left([\![h^*(x) = 1]\!] - [\![h(x) = 1]\!]\right) + (1 - \eta(x))\left([\![h^*(x) = 0]\!] - [\![h(x) = 0]\!]\right).
\end{aligned}$$

Since $[\![h^*(x) = 0]\!] = 1 - [\![h^*(x) = 1]\!]$, the above equals to $(2\eta(x) - 1)\left([\![h^*(x) = 1]\!] - [\![h(x) = 1]\!]\right)$ which is non-negative based on the definition of h^* ($\eta(x) > 1/2 \Leftrightarrow [\![h^*(x) = 1]\!] = 1$). Thus we have

$$\int \mathbb{P}(h(X) \neq Y | X = x) d\mathbb{P}(x) \geq \int \mathbb{P}(h^*(X) \neq Y | X = x) d\mathbb{P}(x).$$

or equivalently, $\mathbb{P}(h(X) \neq Y) \geq \mathbb{P}(h^*(X) \neq Y)$. $\square$

Bayes Classifier: some thoughts

Exercise: Verify the following useful formulae

$$
\begin{aligned}
L^* &= \inf_{h:\mathbb{R}^d \to \{0,1\}} \mathbb{P}(h(X) \neq Y) \\
&= \mathbb{E}[\min\{\eta(X), 1 - \eta(X)\}] \\
&= \tfrac{1}{2} - \tfrac{1}{2}\mathbb{E}[|2\eta(X) - 1|].
\end{aligned}
$$

(*Hint:* Use notation from above proof)
We call L^* the Bayes Error (the minimum error possible any classifier; this is an idealized quantity)

Question: Why is this "idealized?"

Bayes Classifier

Class conditional distribution: Let $\mathcal{Y} = \{0, 1\}$. We define

$$\eta(x) := \mathbb{P}(Y = 1 \mid X = x) = \mathbb{E}[Y \mid X = x].$$

Bayes Classifier

$$h^*(x) := \begin{cases} 1, & \text{if } \eta(x) = \mathbb{P}(Y = 1 | X = x) > \frac{1}{2}, \\ 0, & \text{otherwise.} \end{cases}$$

More practical approach?

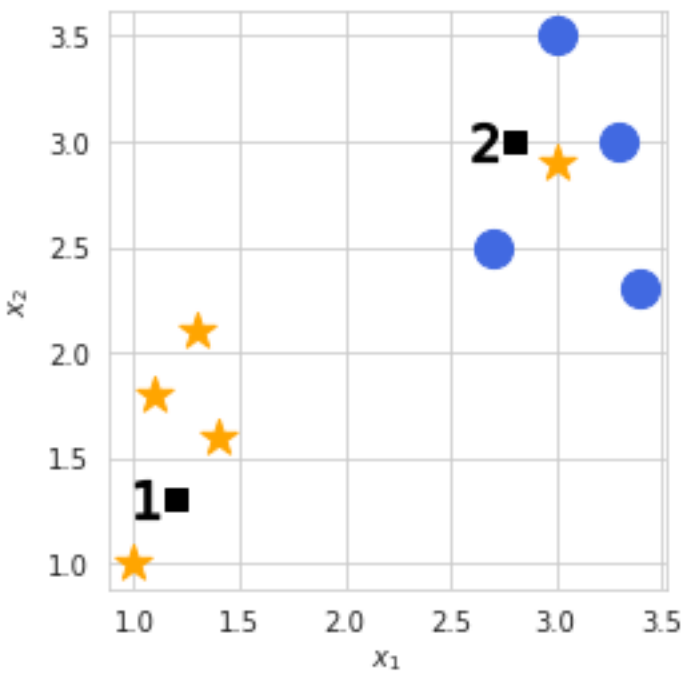
Nearest Neighbor Classifier

Training: None (or rather: **memorize** the data!)

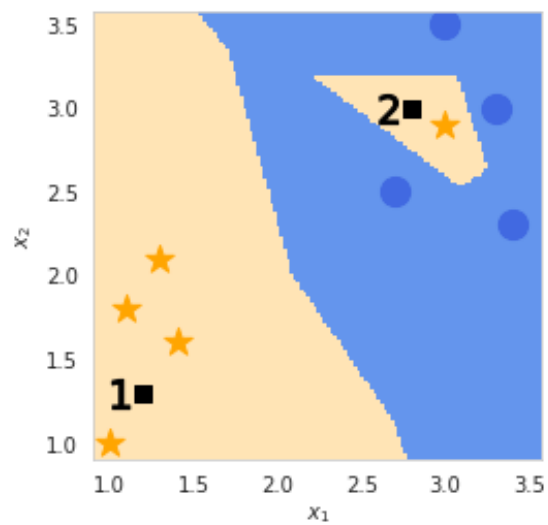
Testing: ***for*** each test data point 'x' ***do***:

find the 'k' points in training data **nearest** to 'x'

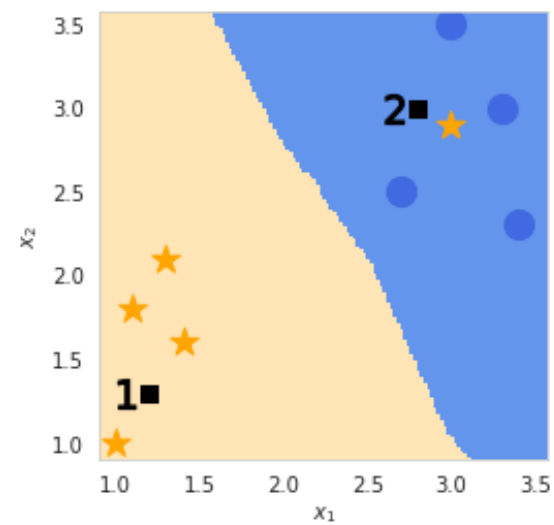
predict label 'y' for 'x' by taking (possibly weighted) majority label of the 'k' points from above



$k = 1$

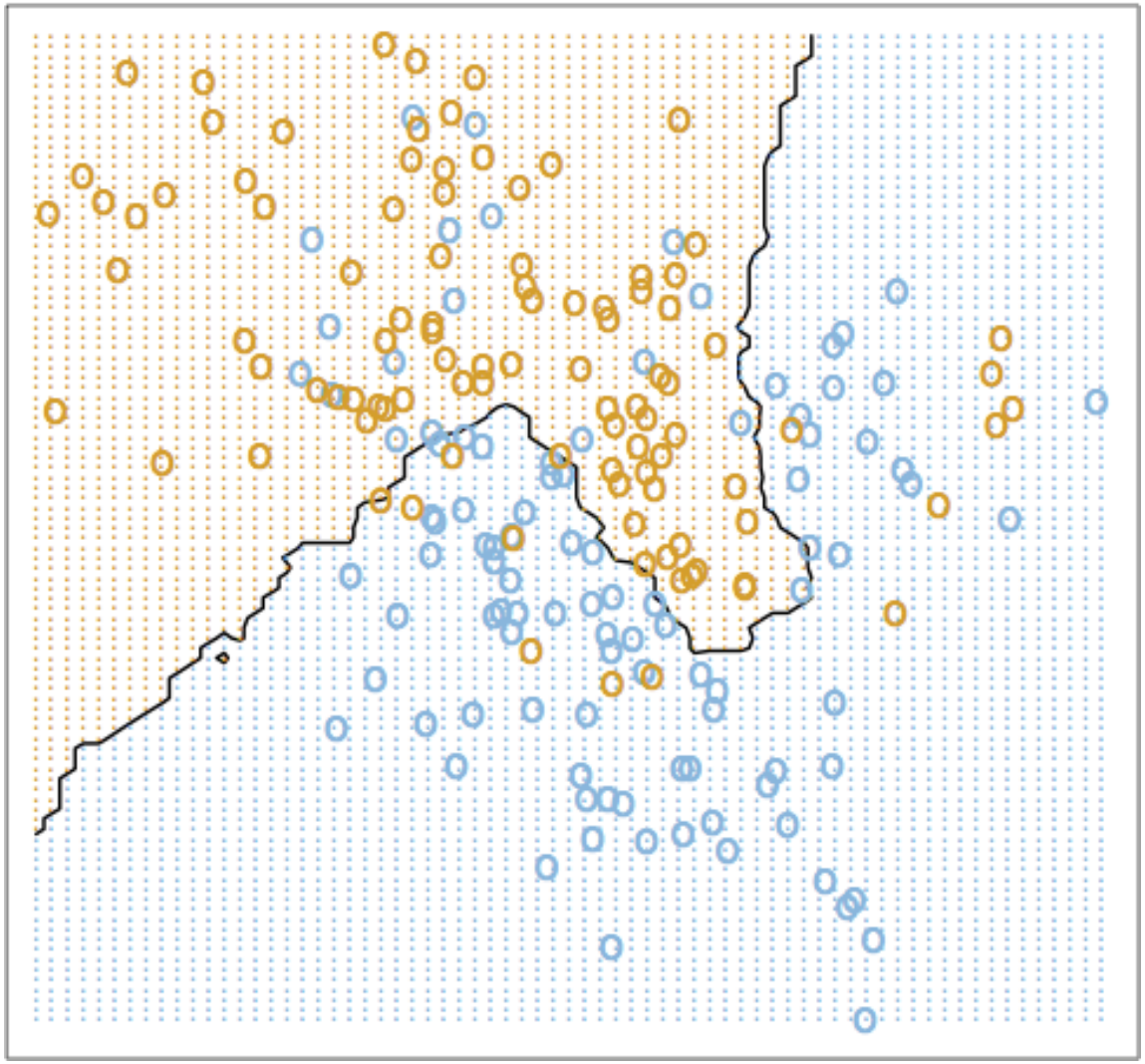


$k = 5$



What would the $k = 9$ case look like?

Nearest Neighbor Classification



k-NN can learn complex **nonlinear** classifiers

Image: Elements of Statistical Learning Theory

1-nearest-neighbor vs. Bayes

Asymptotically, it can be shown that the error of the 1-nearest-neighbor classifier is

$$L_{NN} = \mathbb{E}[2\eta(X)(1-\eta(X))].$$

Theorem. $L_{NN} \leq 2L^*$ (where L^* is Bayes error)

Proof. Given $X = x$, let $X'(n)$ the closest data point to x amongst given n observations. Then due to $\mathcal{X} \subset \mathbb{R}^d$ (i.e. complete, separable metric space), it can be argued that $X'(n) \to x$ as $n \to \infty$ with probability 1. Further, η is continuous. Therefore, $\eta(X'(n)) \to \eta(x)$ as $n \to \infty$ with probability 1. Let $Y'(n)$ be the label observed associated $X'(n)$. Then,

$$\begin{aligned}
\mathbb{P}(h_{\text{1-NN}}(x) \neq Y | X = x) &= \mathbb{P}(Y'(n) \neq Y | X = x) \\
&= \mathbb{P}(Y'(n) = 1, Y = 0 | X = x) + \mathbb{P}(Y'(n) = 0, Y = 1 | X = x) \\
&\overset{(a)}{=} \mathbb{P}(Y'(n) = 1 | X = x)\mathbb{P}(Y = 0 | X = x) + \mathbb{P}(Y'(n) = 0 | X = x)\mathbb{P}(Y = 1 | X = x) \\
&= \eta(X'(n))(1 - \eta(x)) + (1 - \eta(X'(n))\eta(x) \\
&\to 2\eta(x)(1 - \eta(x)) \\
&\overset{(b)}{=} 2\min\{\eta(x), 1 - \eta(x)\}\max\{\eta(x), 1 - \eta(x)\} \\
&\overset{(c)}{\leq} 2\min\{\eta(x), 1 - \eta(x)\}.
\end{aligned}$$

In above, (a) follows from the fact that $Y'(n)$ and Y are generated independently per our generative model; (b) from that the fact for $\alpha, \beta \in \mathbb{R}$ we have $\alpha\beta = \min\{\alpha, \beta\}\max\{\alpha, \beta\}$; and (c) from the fact that $\eta(x) \in [0, 1]$ as it is probability. Then, the claim of theorem follows by recalling that the Bayes risk $L^* = \mathbb{E}[\min\{\eta(X), 1 - \eta(X)\}]$. □

Bayes Classifier

Class conditional distribution: Let $\mathcal{Y} = \{0, 1\}$. We define

$$\eta(x) := \mathbb{P}(Y = 1 \mid X = x) = \mathbb{E}[Y \mid X = x].$$

Bayes Classifier

$$h^*(x) := \begin{cases} 1, & \text{if } \eta(x) = \mathbb{P}(Y = 1 | X = x) > \frac{1}{2}, \\ 0, & \text{otherwise.} \end{cases}$$

$$\begin{aligned}
L^* &= \inf_{h:\mathbb{R}^d \to \{0,1\}} \mathbb{P}(h(X) \neq Y) \\
&= \mathbb{E}[\min\{\eta(X), 1 - \eta(X)\}]
\end{aligned}$$

Classification: what would we like?

- Ideally, we want non-asymptotic results, to better understand how many examples (i.e., how large N) do we need to attain a certain error rate
- We may also have some prior knowledge about (X, Y) that we may wish to incorporate
- Noise, robustness, adversarial learning, and other concerns
- All of these can be accommodated; let us look at another more explicit paradigm

Empirical Risk Minimization

What is Empirical Risk Minimization?

Learner does not know $\mathbb{P}(X, Y)$, so true error (Bayes error) is not known to the learner. However,

- **Training Error**: The error that the classifier incurs on the training data

$$L_S(h) := \frac{1}{N} \# \{i \in [N] \mid h(x_i) \neq y_i\},$$

aka *empirical risk*

- **ERM principle**: Seek predictor that minimizes $L_S(h)$
- **Pitfall**: Overfitting!

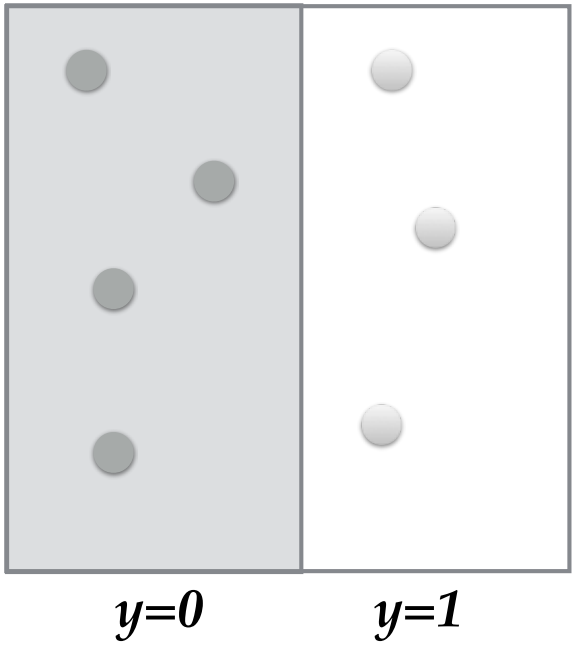
- **Measuring success**: Error of classifier aka risk aka generalization error:

$$L(h) \equiv L_{\mathbb{P}}(h) := \mathbb{P}(h(X) \neq Y)$$

Overfitting: Pitfall of ERM

$$S = \{(x_i, y_i) \mid 1 \leq i \leq N\}$$

$$h(x) = \begin{cases} y_i, & \text{if } x = x_i \\ 0, & \text{otherwise} \end{cases}$$



x distributed uniformly in the unit square

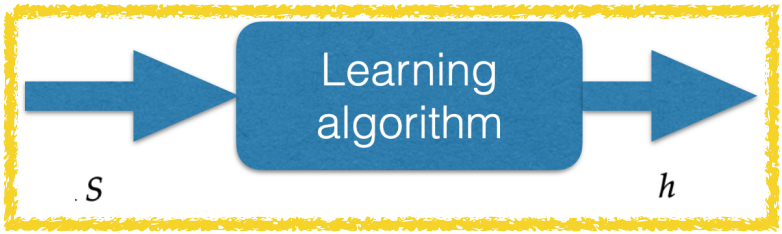
- This classifier has 0 empirical risk!
- As bad as a random guess (error probability on unseen data =1/2)

How to tackle overfitting?

- Rather than give up on ERM, we search for settings where it may actually work.

- **Inductive bias**: Apply ERM over a restricted search space.
  1. Learner chooses a *hypothesis class* $\mathcal{H}$ (i.e., set of predictors it is going to optimize over) in advance before having seen any training data
  2. $\text{ERM}_{\mathcal{H}}$ uses ERM to learn $h : \mathcal{X} \to \mathcal{Y}$ by using S

$$\text{ERM}_{\mathcal{H}}(S) \in \operatorname*{argmin}_{h \in \mathcal{H}} L_S(h)$$



- **Note**: Ideally $\mathcal{H}$ should be governed by knowledge of data. But even "simple" choices of $\mathcal{H}$ can overfit if we are not careful. Of course, overly strong inductive bias can lead to underfitting.

ERM Theory

Question: When does ERM work? In other words, if we minimize $L_S(h)$, what bearing does that have on $L(h)$?

Goal of learning theory is to study this (and such) question(s).

Informally, if for all $h \in \mathcal{H}$, $L_S(h)$ is a good approximation to $L(h)$, then ERM will also return a good hypothesis

$$L_{\mathbb{P}}(h_S) \leq \min_{h \in \mathcal{H}} L_{\mathbb{P}}(h) + \epsilon$$

Explore: When and why might a certain hypothesis class be "better" for learning?

▶ **Training Error**: The error that the classifier incurs on the training data

$$L_S(h) := \frac{1}{N} \# \{ i \in [N] \mid h(x_i) \neq y_i \},$$

Error of classifier aka risk aka

$$\equiv L_{\mathbb{P}}(h) := \mathbb{P}(h(X) \neq Y)$$

ERM: bias-complexity decomposition

$$L_{\mathbb{P}}(h_S) = \epsilon_{\text{apx}} + \epsilon_{\text{est}}$$

Thus, prob of error on random (unseen) data, decomposes into

$$\epsilon_{\text{apx}} := \min_{h \in \mathcal{H}} L_{\mathbb{P}}(h) \quad (\text{APPROX ERROR})$$

$$\epsilon_{\text{est}} := L_{\mathbb{P}}(h_S) - \epsilon_{\text{apx}} \quad (\text{ESTIMATION ERROR})$$

- Approximation error also referred to as structural error (bias)
- Estimation error also referred to as variance

Informally, if for all $h \in \mathcal{H}$, $L_S(h)$ is a good approximation to $L(h)$, then ERM will also return a good hypothesis

$$L_{\mathbb{P}}(h_S) \leq \min_{h \in \mathcal{H}} L_{\mathbb{P}}(h) + \epsilon$$

ERM: bias-complexity decomposition

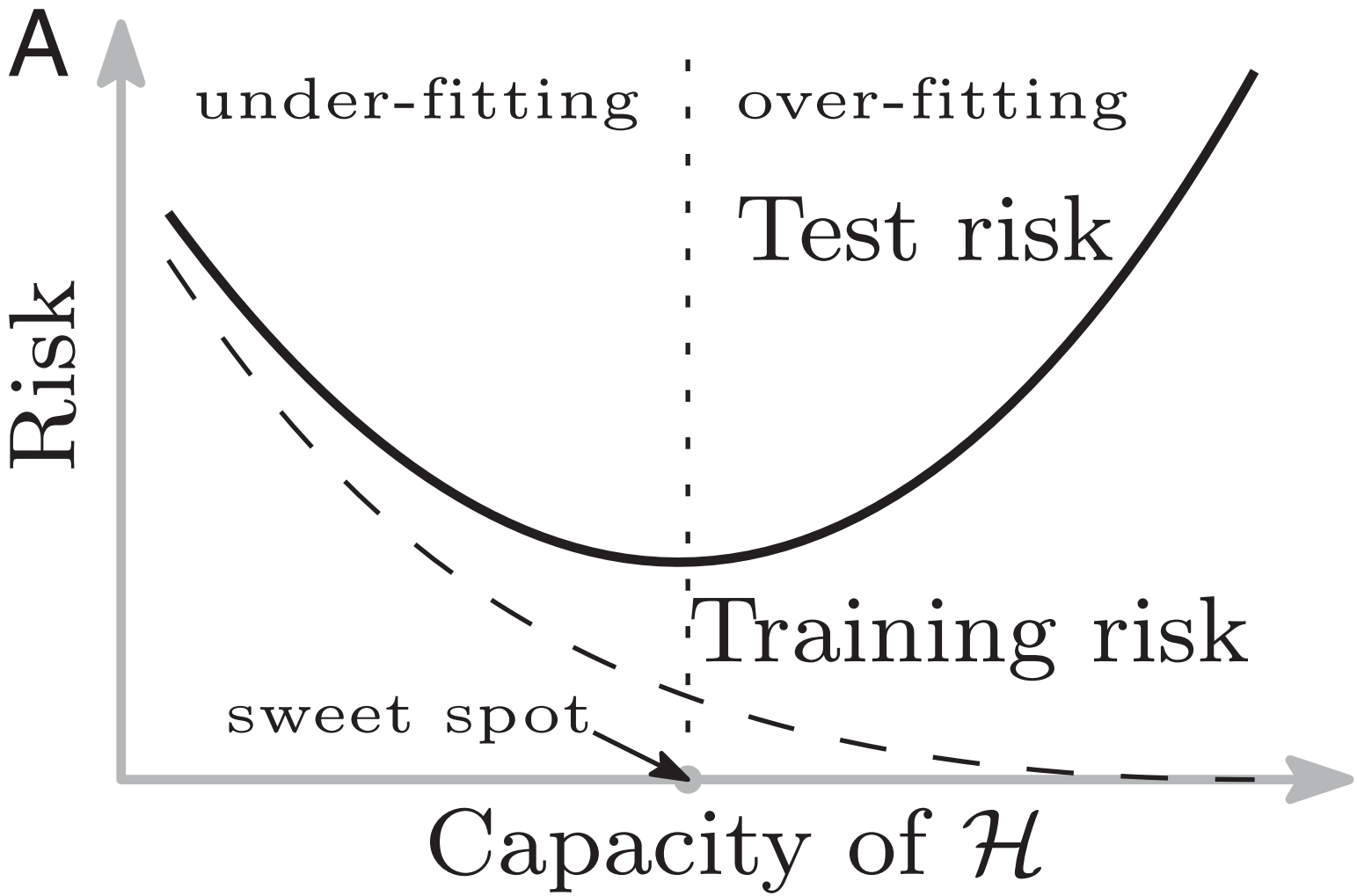
- **Approx error**: Min risk achievable by a predictor in $\mathcal{H}$. Measures how much risk due to inductive bias (observe, does not depend on N or S)
- **Estimation error**: Difference between approx error and error achieved by the ERM predictor (on test data). This error arises because training error (empirical risk) is just a proxy for the true risk.

$$\epsilon_{\text{apx}} := \min_{h \in \mathcal{H}} L_{\mathbb{P}}(h) \quad (\textsc{Approx Error})$$

$$\epsilon_{\text{est}} := L_{\mathbb{P}}(h_S) - \epsilon_{\text{apx}} \quad (\textsc{Estimation Error})$$

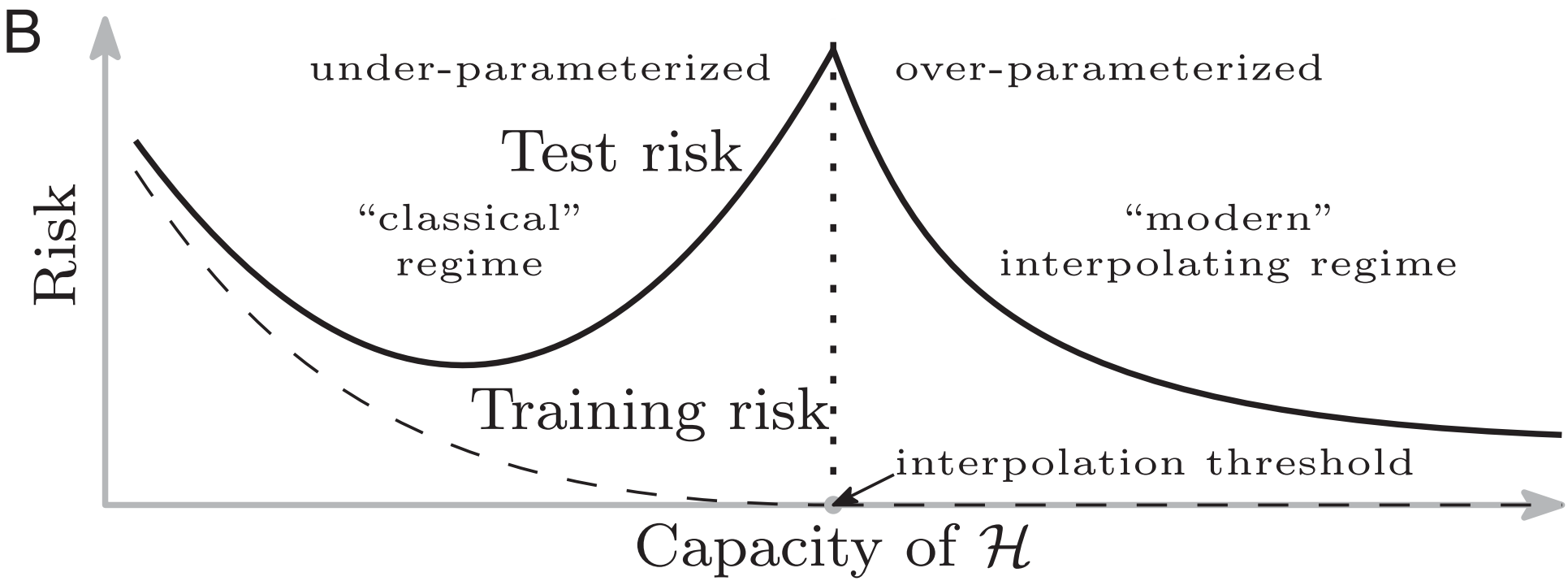
- Quality of estimation depends on training set size N and on the "richness / complexity" of the hypothesis class (e.g., for a finite hypothesis class, estimation error increases as $\log |\mathcal{H}|$ and decreases with N).

ERM: Bias-complexity tradeoff



Classical training vs test curve

ERM: Bias-Complexity Tradeoff



"Modern" viewpoint on generalization: the double-descent curve

Reconciling modern machine-learning practice and the classical bias–variance trade-off

Mikhail Belkin[a,b,1], Daniel Hsu[c], Siyuan Ma[a], and Soumik Mandal[a]