CS 760: Machine Learning
Intro to Learning Theory

Fred Sala

University of Wisconsin-Madison

Nov. 23, 2021

Announcements

- **Logistics**:
  - HW 7 due today
  - Happy Thanksgiving! Enjoy break.
- Class roadmap:

Tues., Nov. 23	**PCA Review & Learning Theory**
Tues., Nov. 30	RL I
Thurs., Dec. 2	RL II
Tues., Dec. 7	RL III
Thurs., Dec 9	Fairness & Ethics

Outline

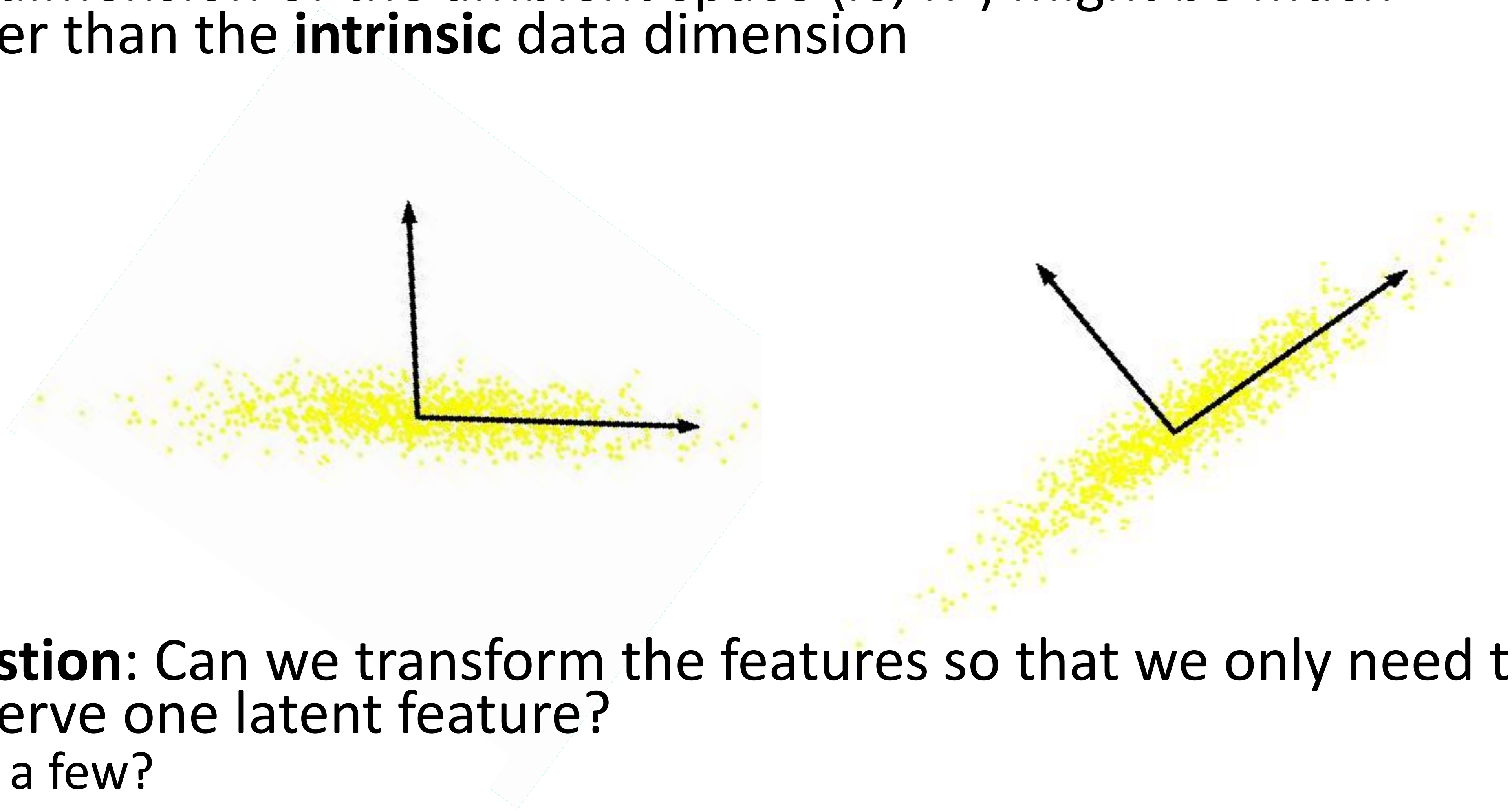
- **Review & PCA**
  - Intuition, operation, interpretations, compression
- **Intro to Learning Theory**
  - Error decomposition, bias-variance tradeoff
- **PAC Learning Framework**
  - Definition, intuition, sample complexity bounds

Outline

- **Review & PCA**
  - Intuition, operation, interpretations, compression
- **Intro to Learning Theory**
  - Error decomposition, bias-variance tradeoff
- **PAC Learning Framework**
  - Definition, intuition, sample complexity bounds

PCA Intuition

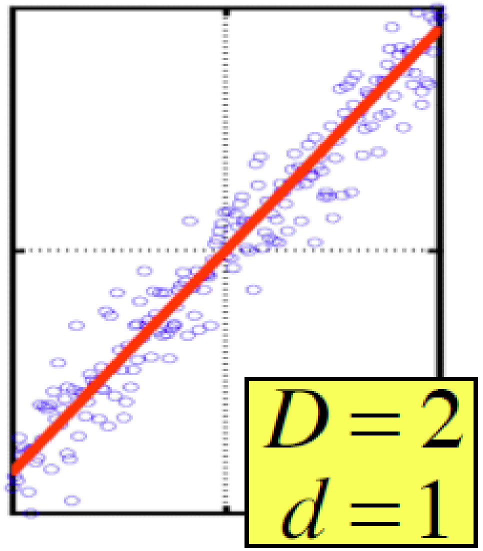
- The dimension of the ambient space (ie, R^d) might be much higher than the **intrinsic** data dimension

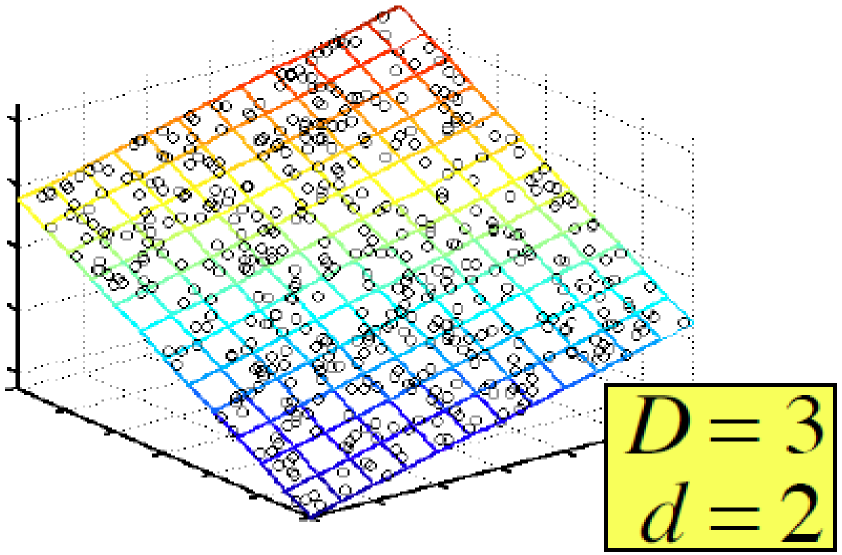


- **Question**: Can we transform the features so that we only need to preserve one latent feature?
  - Or a few?

PCA Intuition

- Some more visualizations

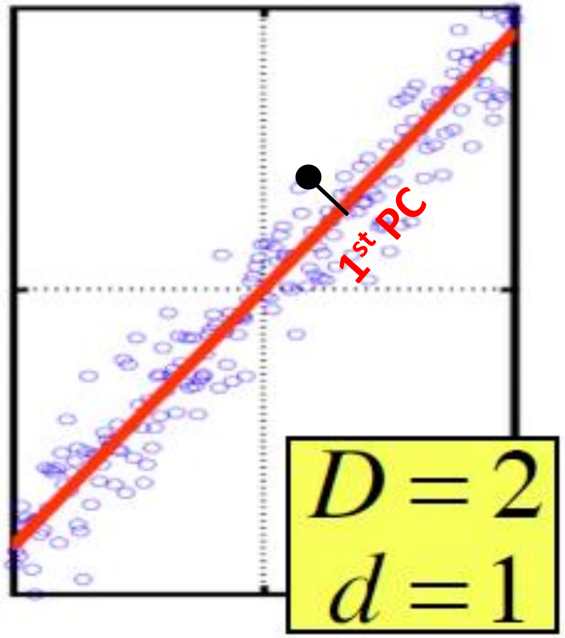




- In case where data lies on or near a low d-dimensional linear subspace, axes of this subspace are an effective representation of the data.

PCA: Principal Components

- **Principal Components (PCs)** are orthogonal directions that capture most of the variance in the data.
  - First PC – direction of greatest variability in data.
  - Projection of data points along first PC discriminates data most along any one direction



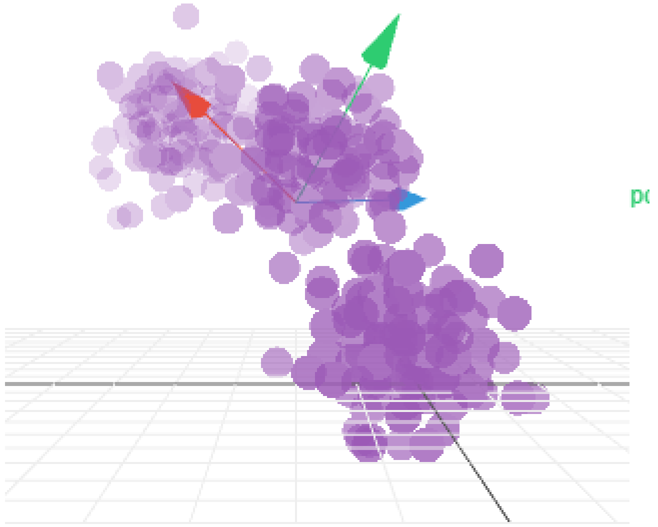
PCA: Principal Components and Projection

- How does dimensionality reduction work? From *d* dimensions to *r* dimensions:
  - Get
  - Orthogonal!

$$v_1, v_2, \ldots, v_r \in \mathbb{R}^d$$

- Maximizing variability
  - Equivalent to **minimizing reconstruction error**

- Then project data onto PCs → d-dimensional



Victor Powell

PCA First Step

- First component,

$$v_1 = \arg\max_{\|v\|=1} \sum_{i=1}^{n} \langle v, x_i \rangle^2$$

- Same as getting

$$v_1 = \arg\max_{\|v\|=1} \|Xv\|^2$$

PCA Recursion

- Once we have *k-1* components, next?

$$\hat{X}_k = X - \sum_{i=1}^{k-1} X v_i v_i^T$$

Deflation

- Then do the same thing

$$v_k = \arg\max_{\|v\|=1} \|\hat{X}_k w\|^2$$

PCA Interpretations

- The v's are eigenvectors of XX^T **(Gram matrix)**
  - We'll see why in a second

- XX^T (proportional to) sample covariance matrix
  - When data is 0 mean!
  - I.e., PCA is eigendecomposition of sample covariance

- Nested subspaces *span(v1), span(v1,v2),…,*

PCA Interpretations: First Component

- Two specific ways to think about the first component
- **Maximum variance direction**
  - What we saw so far

$$\sum_{i=1}^{n} (\mathbf{v}^T \mathbf{x}_i)^2 = \mathbf{v}^T \mathbf{X}\mathbf{X}^T \mathbf{v}$$

- **Minimum reconstruction error**
  - A direction so that projection yields minimum MSE in reconstruction

$$\sum_{i=1}^{n} \|\mathbf{x}_i - (\mathbf{v}^T \mathbf{x}_i)\mathbf{v}\|^2$$

PCA Interpretations: Equivalence

- Interpretation 1.

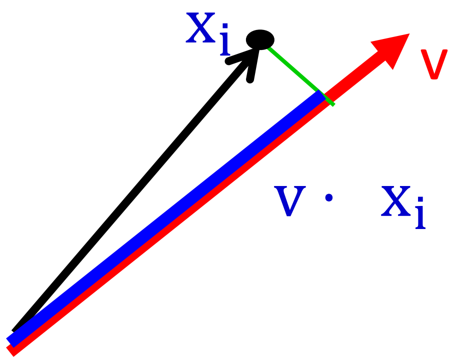
  Maximum variance direction

$$\sum_{i=1}^{n} (\mathbf{v}^T \mathbf{x}_i)^2 = \mathbf{v}^T \mathbf{X}\mathbf{X}^T \mathbf{v}$$

$$\sum_{i=1}^{n} \|\mathbf{x}_i - (\mathbf{v}^T \mathbf{x}_i)\mathbf{v}\|^2$$

- Interpretation 2.

  Minimum reconstruction error



- Why are these equivalent?
  - Use Pythagorean theorem.
  - Maximizing **blue** segment is the same as minimizing the **green**

PCA Covariance Matrix Interpretation

- Recall our first PC, maximized variance:

$$\max_{\mathbf{v}} \quad \mathbf{v}^T \mathbf{X} \mathbf{X}^T \mathbf{v} \quad \text{s.t.} \quad \mathbf{v}^T \mathbf{v} = 1$$

- Constrained optimization
  - Recall our usual approach: Lagrangian + KKT conditions

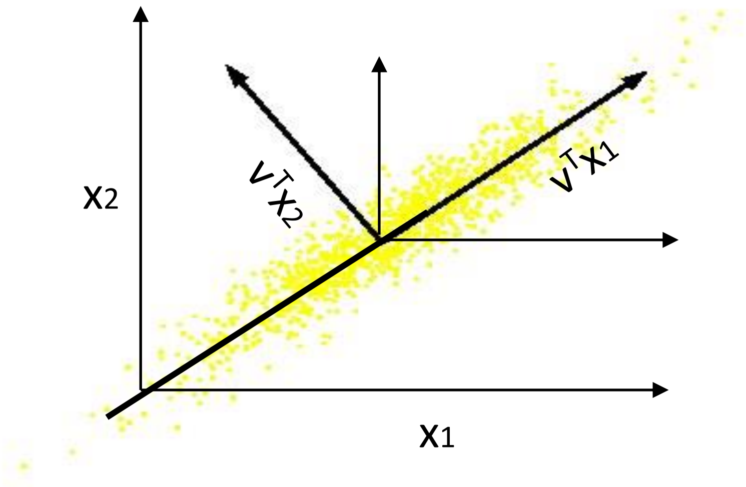
$$\text{Lagrangian: } \max_{\mathbf{v}} \mathbf{v}^T \mathbf{X} \mathbf{X}^T \mathbf{v} - \lambda \mathbf{v}^T \mathbf{v}$$

$$\partial / \partial \mathbf{v} = 0 \quad (\mathbf{X}\mathbf{X}^T - \lambda \mathbf{I})\mathbf{v} = 0 \quad \Rightarrow \boxed{(\mathbf{X}\mathbf{X}^T)\mathbf{v} = \lambda \mathbf{v}}$$

PCA Covariance Matrix Interpretation

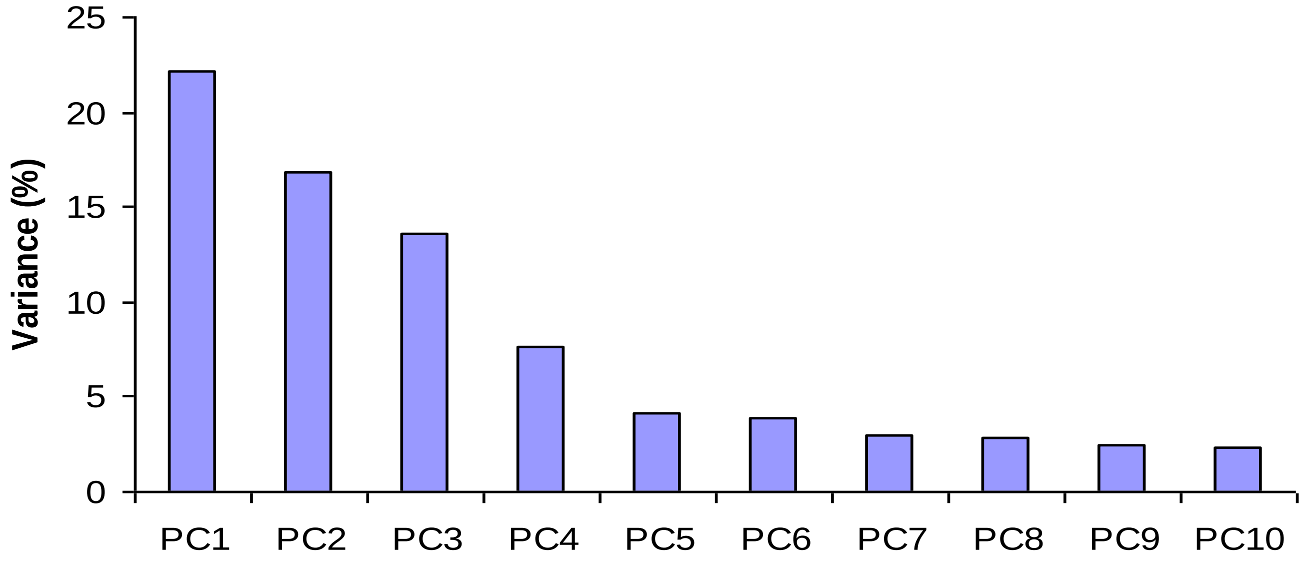
- So… $\Rightarrow (\mathbf{X}\mathbf{X}^T)\mathbf{v} = \lambda\mathbf{v}$
- Means that v (the first PC) is an eigenvector of XX^T
- Its eigenvalue λ denotes the amount of variability captured along that dimension

- PCs are just the eigenvectors…
  - How to find them? Eigendecomposition
- Don't need to keep all eigenvectors
  - Just the ones for largest eigenvalues



PCA Dimensionality Reduction

- In high-dimensional problems, data sometimes lies near a linear subspace, as noise introduces small variability
- Only keep data projections onto principal components with **large** eigenvalues
- Can *ignore* the components of smaller significance.



Application: Image Compression

- Start with image; divide into 12x12 patches
  - I.E., 144-D vector
  - **Original image:**

Application: Image Compression

- Project to 6D,

Compressed

Original

Break & Quiz

Outline

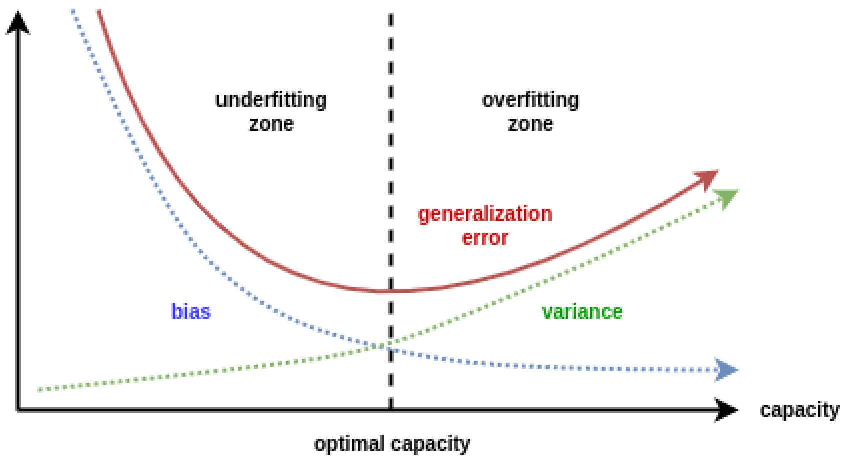
- **Review & PCA**
  - Intuition, operation, interpretations, compression
- **Intro to Learning Theory**
  - Error decomposition, bias-variance tradeoff
- **PAC Learning Framework**
  - Definition, intuition, sample complexity bounds

Learning Theory

- Goal: try to analyze error, and especially generalization
  - i.e., **the expected error** on the whole distribution

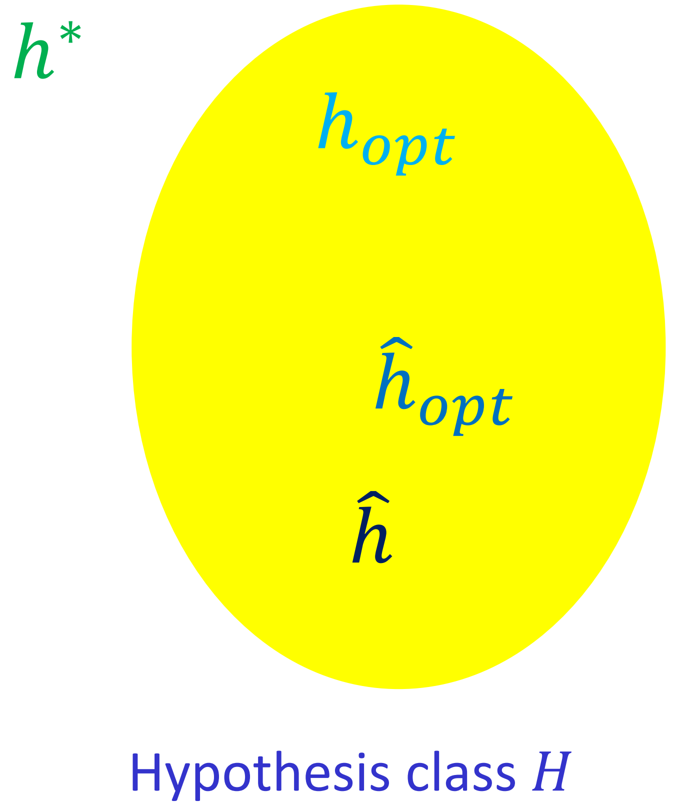
- We will cover a few ideas:
  - Error decomposition & generalization
  - Bias-variance tradeoff
  - PAC framework

- Deep subject overall.



Error Decomposition

- h^*: the optimal function (Bayes classifier)
- h_{opt}: the optimal hypothesis on the data distribution
- $\hat{h}_{opt}$: the optimal hypothesis on the training data
- $\hat{h}$: the hypothesis found by the learning algorithm



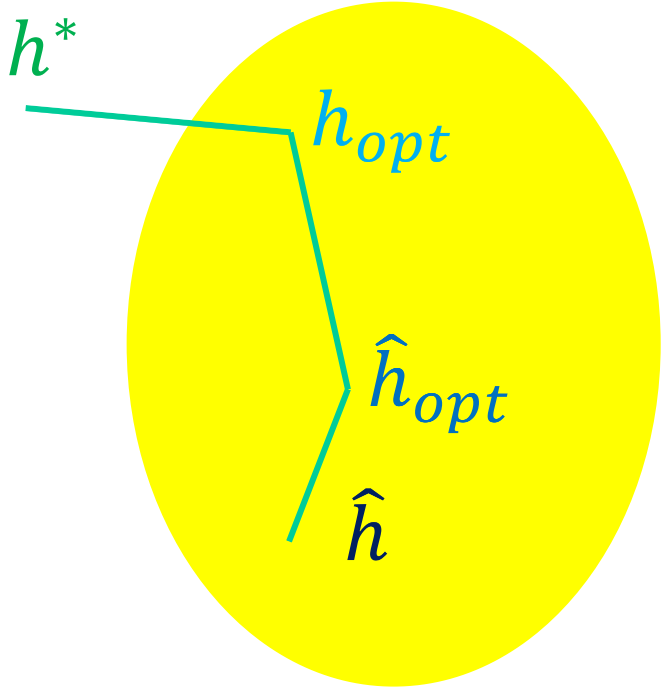
Error Decomposition

$$err(\hat{h}) - err(h^*)$$

$$= err(h_{opt}) - err(h^*)$$

$$+ err(\hat{h}_{opt}) - err(h_{opt})$$

$$+ err(\hat{h}) - err(\hat{h}_{opt})$$



Hypothesis class H

Error Decomposition

$$err(\hat{h}) - err(h^*)$$

$$= err(h_{opt}) - err(h^*)$$

$$+ err(\hat{h}_{opt}) - err(h_{opt})$$

$$+ err(\hat{h}) - err(\hat{h}_{opt})$$

Approximation error due to problem modeling (the choice of hypothesis class)

Estimation error due to finite data

Optimization error due to imperfect optimization

Bounding Estimation Error

$$err(\hat{h}_{opt}) - err(h_{opt})$$

$$= err(\hat{h}_{opt}) - \widehat{err}\,(\hat{h}_{opt})$$

$$+ \widehat{err}\,(\hat{h}_{opt}) - err(h_{opt})$$

$$\leq err(\hat{h}_{opt}) - \widehat{err}\,(\hat{h}_{opt})$$

$$+ \widehat{err}\,(h_{opt}) - err(h_{opt})$$

$$\leq 2 \sup_{h \in H} |err(h) - \widehat{err}(h)|$$

Another Decomposition

$$err(\hat{h}) = \widehat{err}(\hat{h}) + \underbrace{\left[err(\hat{h}) - \widehat{err}(\hat{h})\right]}_{\text{Generalization gap}}$$

$$\leq \widehat{err}(\hat{h}) + \sup_{h \in H} |err(h) - \widehat{err}(h)|$$

- The training error $\widehat{err}(\hat{h})$ is what we can compute
- Need to control the **generalization gap.**
  - How?

Bounding the Generalization Gap

Have: $err(\hat{h}) \leq \widehat{err}(\hat{h}) + \sup_{h \in H} |err(h) - \widehat{err}(h)|$

- How do we deal with the right-hand term?
- Have, for example,

$$|err(h) - \widehat{err}(h)| \leq R(H) + \sqrt{\log\left(\frac{1}{\delta}\right)/2n}$$

for all h in H and where n is the number of samples, R(H) is the **Rademacher complexity** of the function class

Bounding the Generalization Gap

$$|err(h) - \widehat{err}(h)| \leq R(H) + \sqrt{\log\left(\frac{1}{\delta}\right)/2n}$$

for all h in H and where n is the number of samples, R(H) is the **Rademacher complexity** of the function class

- Rademacher complexity: a measure of how “large” the hypothesis is.
  - How much random data can it fit?
  - Other versions: VC complexity, Gaussian complexity

Bias-Variance Tradeoffs

Consider the task of learning a regression model given a training set $D = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$

- A natural measure of the **error** of f is

$$E_D[(y - f(\mathbf{x};\ D))^2]$$

  - Expectation is taken with respect to the real-world distribution of instances (not the empirical one)

Bias-Variance: Derivation

Take a fixed $\mathbf{x}$. Can rewrite:

$$E_D[(y - f(\mathbf{x};\ D))^2] =$$

$$E_D[(y - E[y|\mathbf{x}])^2] + (f(x; D) - E[y|x])^2$$

Variance of y given x (unrelated to model)

Error of f as a predictor

Bias-Variance: Derivation

Let's look at the 2nd term, and take the expectation over datasets:

$$E_D[(f(\boldsymbol{x};D) - E[y|\boldsymbol{x}])^2] =$$

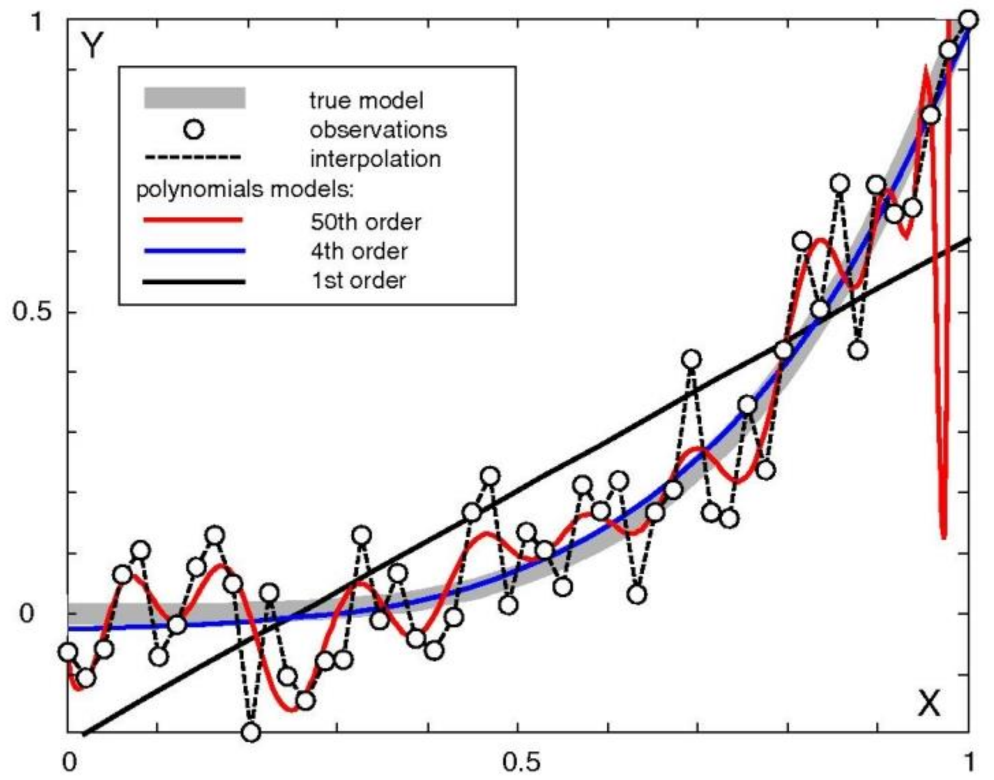
$$= (E_D[f(\boldsymbol{x};D)] - E[y|\boldsymbol{x}])^2 + E_D[(f(\boldsymbol{x};D) - E_D[f(\boldsymbol{x};D)]^2]$$

Bias **Variance**

- **Bias**: if on average $f(\boldsymbol{x};D)$ differs from $E[y|\boldsymbol{x}]$ then $f(\boldsymbol{x};D)$ is a biased estimator of $E[y|\boldsymbol{x}]$
- **Variance**: $f(\boldsymbol{x};D)$ may be sensitive to D and vary a lot from its expected value

Bias-Variance: Polynomial Interpolation

- **Example**:
  - 1st order polynomial has high bias, low variance
  - 50th order polynomial has low bias, high variance
  - 4th order polynomial represents a good trade-off



Bias-Variance: Idea

Predictive error has two controllable components

- expressive/flexible learners reduce *bias*, but increase *variance*

- For many models we can trade-off these two components (e.g. via our selection of k in k-NN)
- The optimal point in this trade-off depends on the particular problem domain and training set size
- Not necessarily a strict trade-off; e.g. with ensembles we can often reduce bias and/or variance without increasing the other term

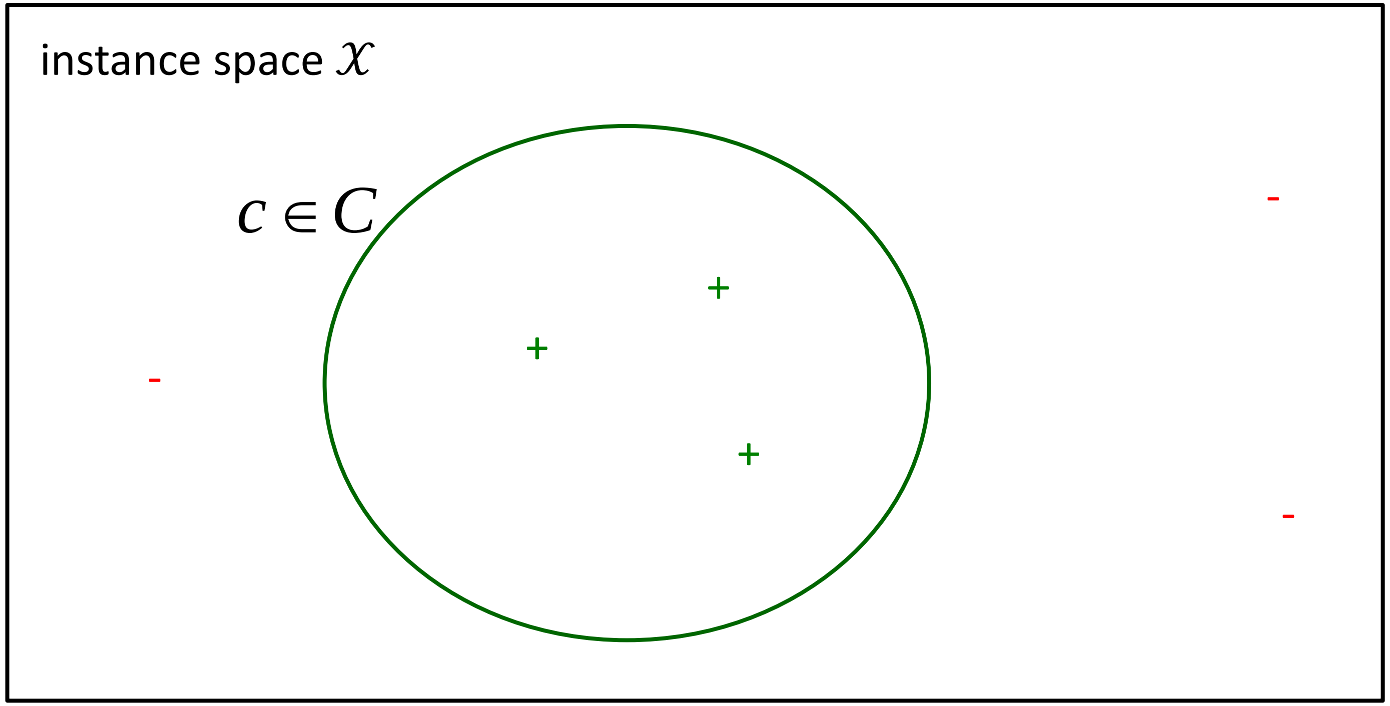
Break & Quiz

Outline

- **Review & PCA**
  - Intuition, operation, interpretations, compression
- **Intro to Learning Theory**
  - Error decomposition, bias-variance tradeoff
- **PAC Learning Framework**
  - Definition, intuition, sample complexity bounds

PAC Learning Setup

PAC learning is a framework used for theoretical analysis. Basic setting:



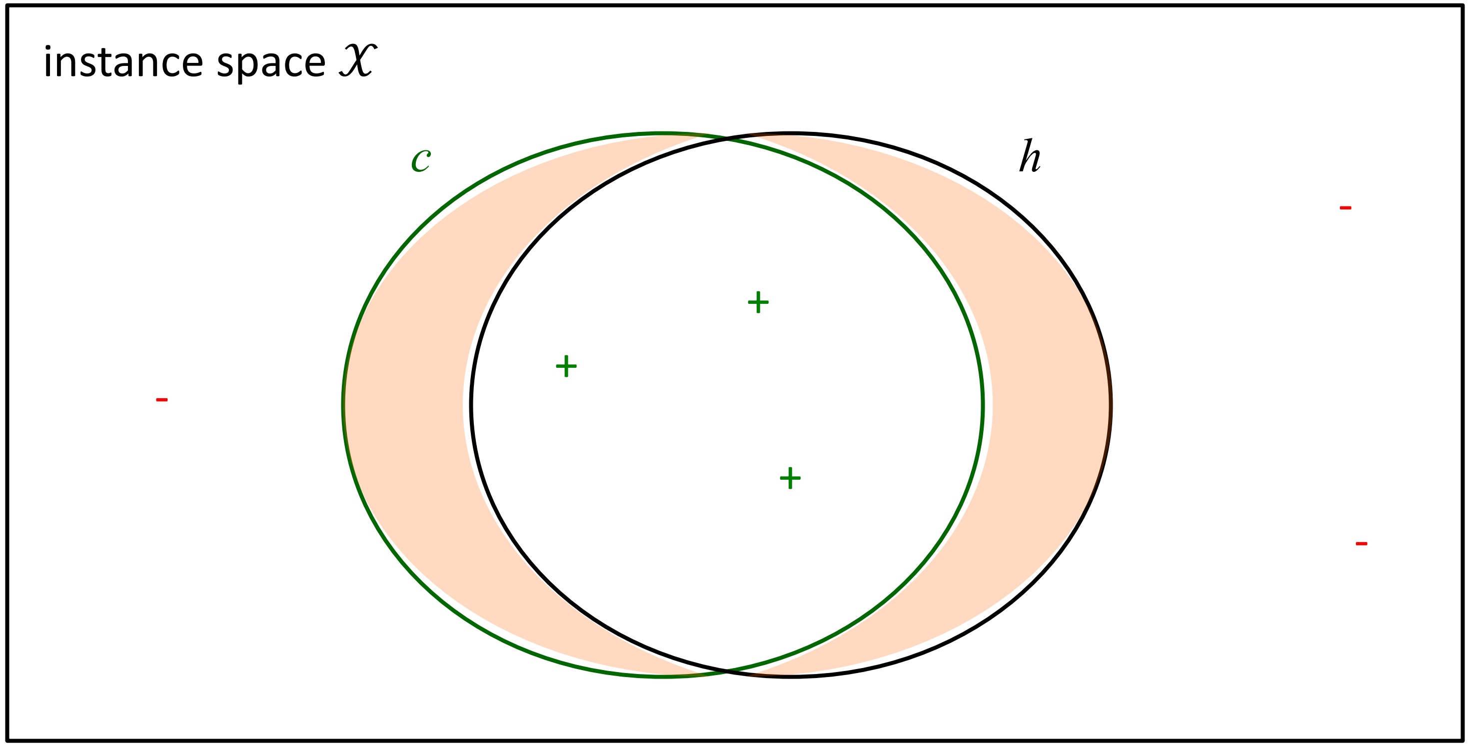
- Set of instances $\mathcal{X}$
- Set of hypotheses (models) H
- Set of possible target concepts C
- Unknown probability distribution $\mathcal{D}$ over instances

PAC Learning Setup

We get a set D of training instances $(\boldsymbol{x}, c(\boldsymbol{x}))$ for some target concept c in C

- each instance $\boldsymbol{x}$ is drawn from distribution $\mathcal{D}$
- class label $c(\boldsymbol{x})$ is provided for each $\boldsymbol{x}$

- learner outputs hypothesis h modeling c
- *Goal:* the *true error* of hypothesis h refers to how often h is wrong on future instances drawn from $\mathcal{D}$



PAC Learning: Two Error Types

We have **two** kinds of errors:

True error: (i.e., on any instance from distribution d):

$$error_{\mathcal{D}}(h) \equiv P_{\mathcal{D}}[c(x) \neq h(x)]$$

Empirical error: (I.e., on our dataset)

$$error_{D}(h) \equiv P_{x \in D}[c(x) \neq h(x)] = \frac{\sum_{x \in D} \delta(c(x) \neq h(x))}{|D|}$$

Goal: Can we bound $error_{\mathcal{D}}(h)$ in terms of $error_{\mathrm{D}}(h)$?

PAC Learning Definition

Consider a class C of possible target concepts defined over a set of instances $\mathcal{X}$ of length n, and a learner L using hypothesis space H

- C is **PAC learnable** by L using H if, for all $c \in C$, distributions $\mathcal{D}$ over $\mathcal{X}$, ε such that $0 < \varepsilon < 0.5$, δ such that $0 < \delta < 0.5$,

- The learner L will, with probability at least $(1\text{-}\delta)$, output a hypothesis $h \in H$ such that $error_{\mathcal{D}}(h) \leq \varepsilon$ in time that is polynomial in the quantities:

$$1/\varepsilon\ ,\ 1/\delta,\ n,\ \text{size}(c)$$

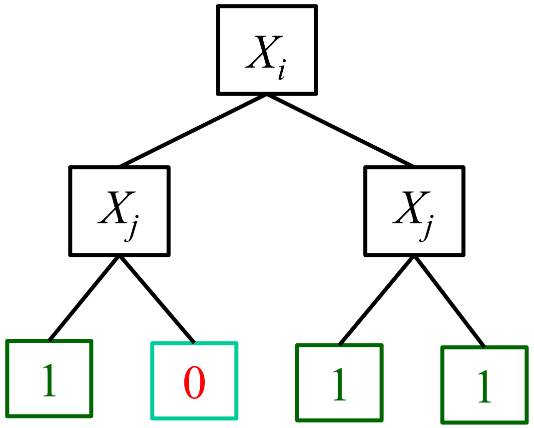
"**Probably Approximately** Correct"

PAC Learning Applications

For finite hypothesis classes, the sample complexity (i.e., the m) so that we get a learner that satisfies the above definition is

$$m \geq \frac{1}{e}\left(\ln|H| + \ln\left(\frac{1}{d}\right)\right)$$

Error tolerance — Size of hypothesis class — Probably correct



Can apply to, for example, decision trees of depth 2 for binary feature vectors

- |H| is the number of splits (ie, n choose 2 times 16: # split choices times # leaf labelings)
- For probability ≥ 0.99 with error ≤ 0.05, number of samples we need is:
- Example: for $n=100$, $m \geq 318$

$$m \geq \frac{1}{.05}\left(\ln\left(8n^2 - 8n\right) + \ln\left(\frac{1}{.01}\right)\right)$$

PAC Learning Discussion

PAC formalizes learning task, allows for non-perfect learning (indicated by ε and δ)

- Requires polynomial computational time

- PAC analysis has been extended to explore a wide range of cases
  - the target concept not in our hypothesis class
  - infinite hypothesis class (VC-dimension theory)
  - noisy training data
  - learner allowed to ask queries
  - restricted distributions (e.g. uniform) over $\mathcal{D}$

- Most analyses are worst case
- Sample complexity bounds are generally not tight

Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov