Towards more trust in overparameterized models for high-dimensional data

Fanny Yang, Statistical Machine Learning Group, D-INFK, ETH Zurich

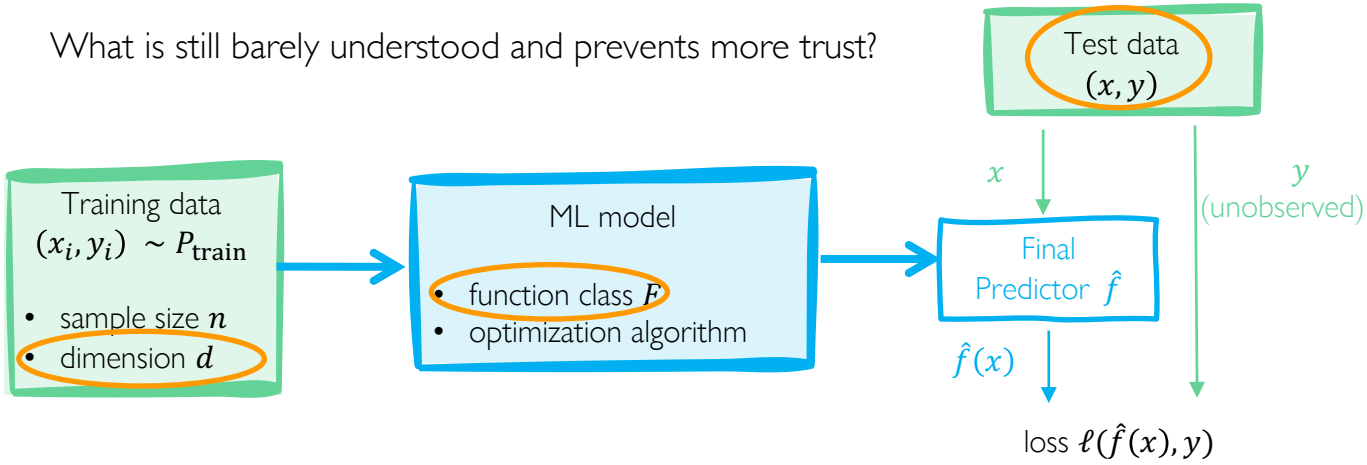
ELLIS Doctoral Symposium, 28. September 2021

Soon AI will be replacing some humans…

…long way to go until really trustworthy

Supervised learning for modern ML

What is still barely understood and prevents more trust?



Two important settings for which understanding is missing

1. High-dimensionality: Data dimension d is large
   Overparameterization: large model classes F that can perfectly fit data

2. Reliability: how model acts when test data $(x, y) \nsim P_{train}$

Plan for Part 1: high-dimensional regression

- Setting up: Regression in the modern data regime
  - Recap: regression in the classical regime
  - Regression in the modern data regime
  - Overparameterized models and estimators
- Two examples where classical intuition fails

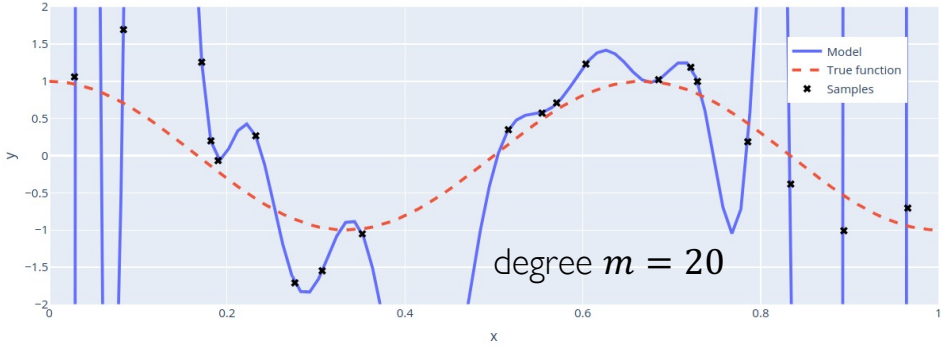
Recap: Regression with the square loss

- **Data generation**: covariates $x \sim P_x$, labels $y = f^\star(x) + \epsilon$ with $x \in R^d$ and $y \in R$, noise ϵ
- **Observe**: training set D with n i.i.d. data points
- **Goal**: Find $\hat{f}_D$ that is close to $f^\star$ in some model class F
- **Vanilla estimator**: minimizer of Mean Square Error

$$\hat{f}_D = \operatorname{argmin}_{f \in F} \sum_{i=1}^{n} \left(y_i - f(x_i)\right)^2$$

Classical setting

- sample size $n \gg d$ dimension
- Conventional wisdom: best fit with large model → large variance

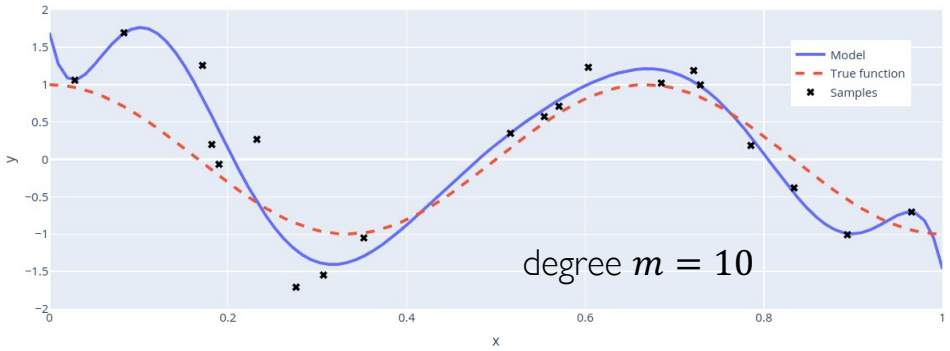


Classical setting

- sample size $n \gg d$ dimension

- Conventional wisdom: best fit with large model → large variance

⇨ regularize model complexity avoiding perfect fit → smaller variance



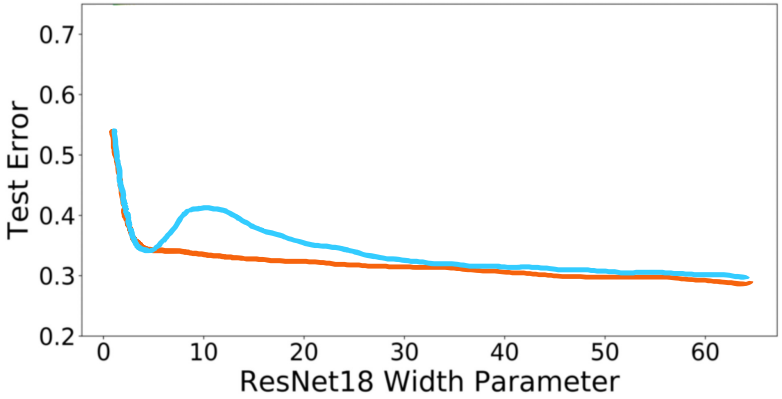
Modern setting

- sample size vs. dimension:
  e.g. ImageNet $n \sim 10^6, d > 10^4$, gene data $n \sim 10^1$ to $10^4, d > 10^4$

→ high-dimensional regime $n \asymp d^{\alpha}$, here: $\alpha \in (0,2)$

- Modern practice: use overparameterized models (can fit data perfectly)

  and without regularization still seems to generalize at least as well

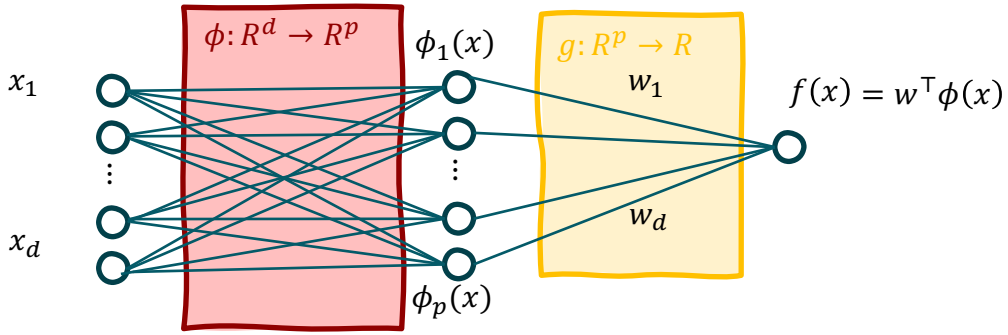


Optimal regularization

No regularization

*[Nakkiran et al. '20]

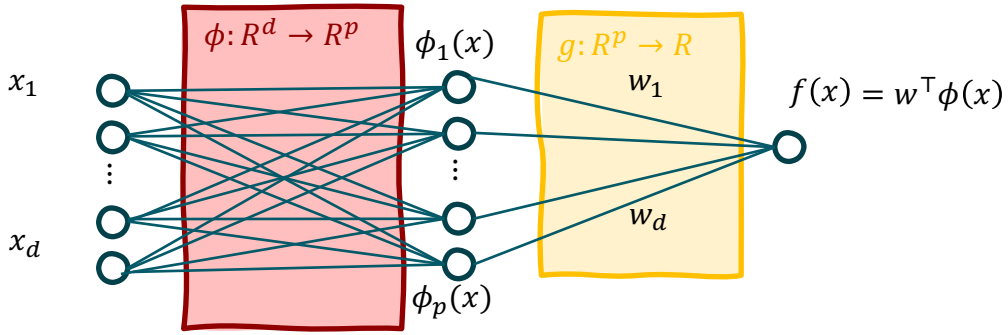
Overparameterized models in a nutshell



Unifying diagram for overparameterized models for high dimensional data $d \gg n$:

- Linear models are already overparameterized for $p = d$ for $\phi(x) = x$
- Kernels can fit nonlinear functions using $p > d$ fixed nonlinear features ϕ
- Neural networks can fit nonlinear functions using p trained features ϕ

Overparameterized models in a nutshell



For overparameterized models F, many models achieve $MSE = 0$

→ initialized at 0, first order method often yields minimum-norm interpolator

$$\hat{f}_D = \operatorname{argmin}_{f \in F} ||f|| \text{ s.t. } y_i = f(x_i) \text{ for all } i$$

for some norm (depends on F and algorithm)

Evaluation of the estimators

What's the expected loss on a test sample or

estimation error $\left\|\hat{f}_D - f^\star\right\|^2_{L^2(P_x)} = E_{X\sim P_x}\left(\hat{f}_D(X) - f^\star(X)\right)^2$

dependent on the model class F, algorithm, n, d?

Study the average error (over data D)

using bias-variance decomposition:

$$\mathrm{E_D}\left\|\hat{f}_D - f^\star\right\|^2 = \underbrace{E_D\left\|\hat{f}_D - E_D\hat{f}\right\|^2}_{\text{variance}} + \underbrace{\left\|E_D\hat{f}_D - f^\star\right\|^2}_{\text{bias}}$$

Plan for Part 1: high-dimensional regression

- Setting up: Regression in the modern data regime
- Two examples where classical intuition fails
  - Example I: Linear models where $p = d$
    Minimum-ℓ_2-norm interpolation when $d \asymp n$
    Intuition from classical setting $d < n$: larger $n \to$ smaller variance
  - Example II: Nonlinear models via kernels with $p = \infty$
    Kernel estimators for $d^{\alpha} \asymp n$
    Intuition for fixed d: you can learn nonlinear functions

Intuition from classical theory does not hold!

Linear least-squares regression for $d \gg n$

Given data matrix $X = \begin{pmatrix} - & x_1 & - \\ & \vdots & \\ - & x_n & - \end{pmatrix}$ with rows $x_i \sim N(0, I_d)$ and labels $y = Xw^\star + \epsilon$

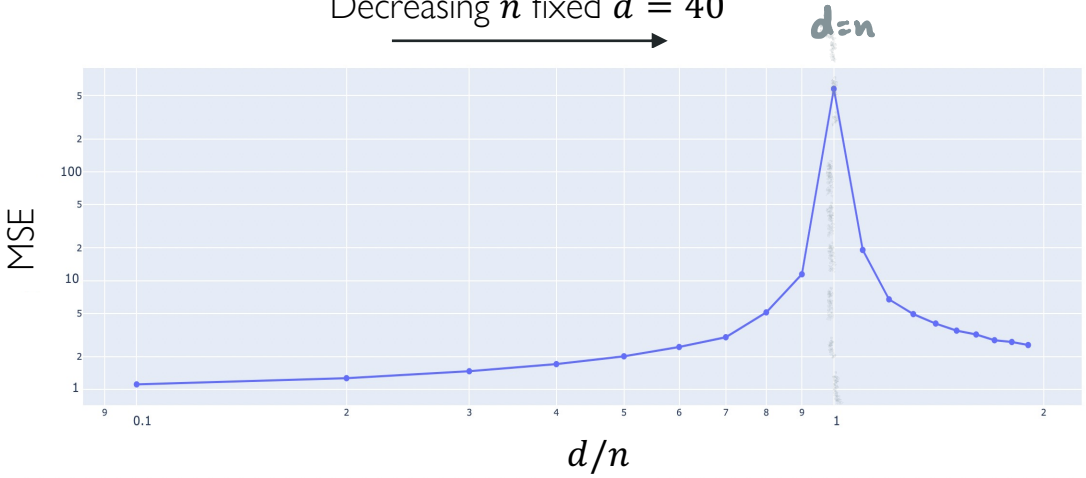
For $d < n$: MSE minimizer

$$\hat{w} = \text{argmin}_{w \in R^d} ||y - Xw||_2$$

For $d > n$: min-ℓ_2-norm interpolator

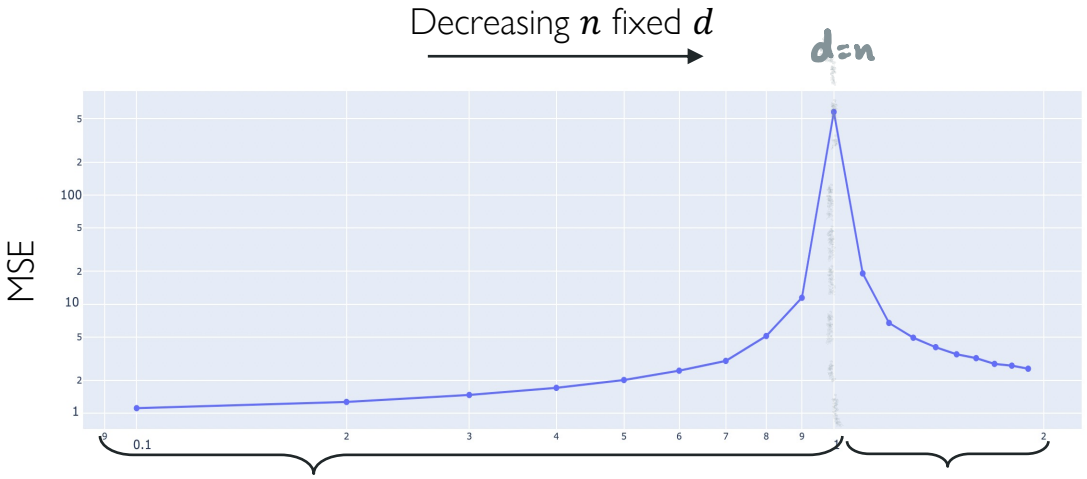
$$\hat{w} = \text{argmin}_{w \in R^d} ||w||_2 \quad \text{s.t. } y = Xw$$

Decreasing n fixed $d = 40$



Linear least-squares regression for $d \gg n$

Goal now: Compare and build intuition for the two regimes



n decreases for fixed d
→ MSE increases

n decreases for fixed d
→ MSE decreases

Recap: Bias and variance for linear regression

For $d < n$: MSE minimizer

$$\hat{w} = \text{argmin}_w ||y - Xw||_2$$

For $d > n$: min-ℓ_2-norm interpolator

$$\hat{w} = \text{argmin}_w ||w||_2 \quad \text{s.t. } y = Xw$$

Solution for both can be written as

$$\hat{w} = (X^\top X)^\dagger X^\top y = \underbrace{(X^\top X)^\dagger X^\top X w^\star}_{\Pi_X w^\star = E_D \hat{w}} + \underbrace{(X^\top X)^\dagger X^\top \epsilon}_{=: w_n \text{ fits noise}}$$

$\Pi_X w^\star = E_D \hat{w}$ → determines bias

$=: w_n$ fits noise → determines variance

Average MSE: $\mathrm{E}_D ||\hat{w}_D - w^\star||^2 = \underbrace{||E_D \hat{w} - w^\star||^2}_{\text{bias } ||\Pi_X w^\star - w^\star||^2} + \underbrace{E_D ||\hat{w}_D - E_D \hat{w}||^2}_{\text{variance } E_D ||w_n||^2}$

= Test loss for $X \sim N(0, I)$

Recap: Bias and variance for linear regression

For $d < n$: MSE minimizer

$\hat{w} = \text{argmin}_w ||y - Xw||_2$

For $d > n$: min-ℓ_2-norm interpolator

$\hat{w} = \text{argmin}_w ||w||_2$ s.t. $y = Xw$

Solution for both can be written as

$$\hat{w} = (X^\top X)^\dagger X^\top y = \underbrace{(X^\top X)^\dagger X^\top X w^\star}_{\Pi_X w^\star = E_D \hat{w}} + \underbrace{(X^\top X)^\dagger X^\top \epsilon}_{=: w_n \text{ fits noise}}$$

$\Pi_X w^\star = E_D \hat{w}$ → determines bias

$=: w_n$ fits noise → determines variance

$\Pi_X w^\star = w^\star$ → bias = 0

MSE minimizer for noise fit

$w_n = \text{argmin}_w ||\epsilon - Xw||_2$

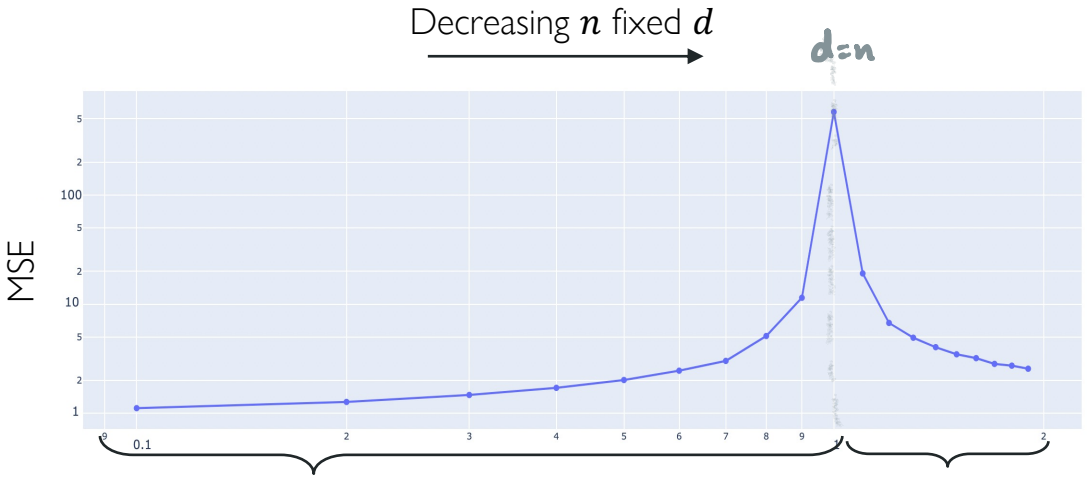
bias $||\Pi_X w^\star - w^\star||^2 \neq 0$

min-ℓ_2-norm interpolator of noise

$w_n = \text{argmin} ||w||_2$ s.t. $\epsilon = Xw$

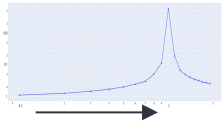
Example I: Linear LS regression for $d \gg n$

Goal now: Compare and build intuition for the two regimes



n decreases for fixed d
→ variance increases

n decreases for fixed d
→ MSE decreases

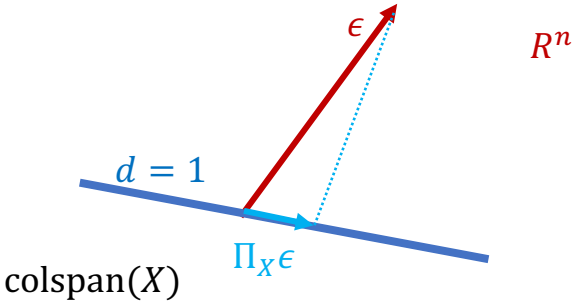


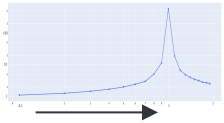
Variance for $\mathbf{d} < n$ increases with d/n

$$w_n = \text{argmin}_w \left\| \begin{matrix} n \\ \epsilon \end{matrix} - \begin{matrix} d \\ n \\ X \end{matrix} \begin{matrix} \\ w \\ d \end{matrix} \right\|$$

$$Xw_n = \Pi_X \epsilon$$

projection onto **colspan** (X)



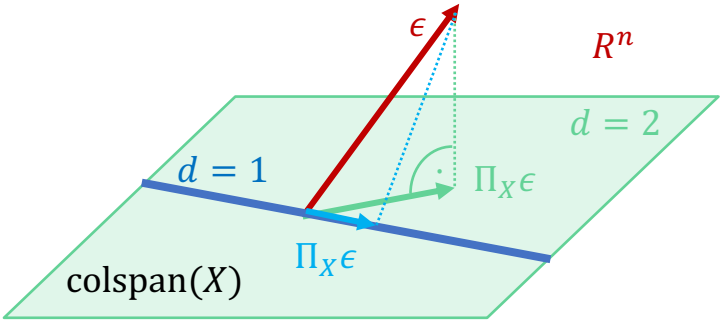


Variance for $\mathbf{d} < n$ increases with d/n

$$w_n = \text{argmin}_w \left\| \begin{matrix} n \\ \epsilon \end{matrix} - \begin{matrix} d \\ n \\ X \end{matrix} \begin{matrix} \\ w \\ d \end{matrix} \right\|$$

$$Xw_n = \Pi_X \epsilon$$

projection onto $\mathbf{colspan}\ (X)$



$\frac{d}{n}$ increases (increasing d fixed n)

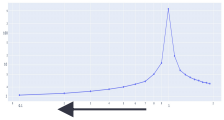
→ $\mathbf{colspan}\ X$ larger

→ projection $\Pi_X\epsilon$ closer to ϵ

→ norm of $\Pi_X\epsilon$ increases

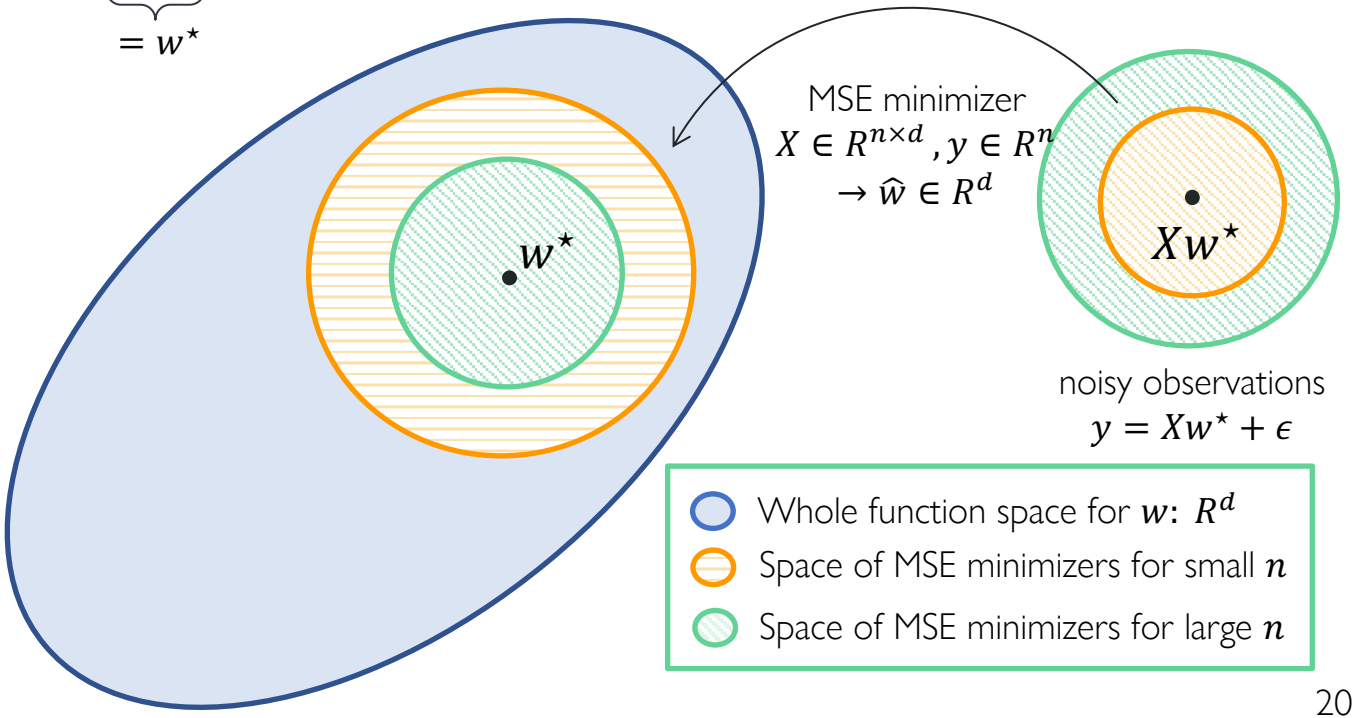
For simplicity: orthogonal $X^\top X = nI_d$ such that $\left|\left|w_n\right|\right|_2 = \frac{1}{\sqrt{n}}\left|\left|Xw_n\right|\right|_2 = \frac{1}{\sqrt{n}}\left|\left|\Pi_X\epsilon\right|\right|_2$

→ variance $E_D\left|\left|w_n\right|\right|^2$ increases with $\frac{d}{n}$



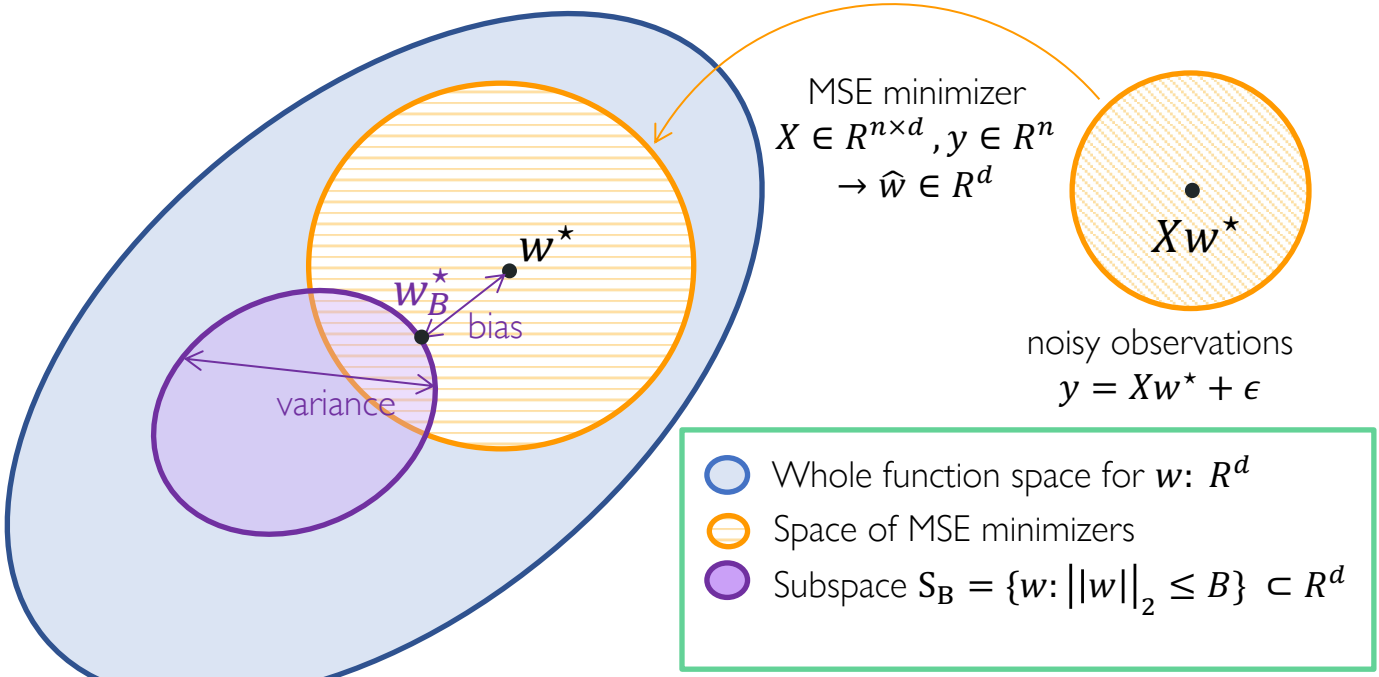
Variance for $d < n$ decreases with n

$\hat{w} = \underbrace{\Pi_X w^\star}_{= w^\star} + w_n$ with bias $\left|\left|\Pi_X w^\star - w^\star\right|\right|^2 = 0$ and variance $E_D\left|\left|w_n\right|\right|^2 = E_D \frac{\left|\left|\Pi_X \epsilon\right|\right|_2^2}{n}$



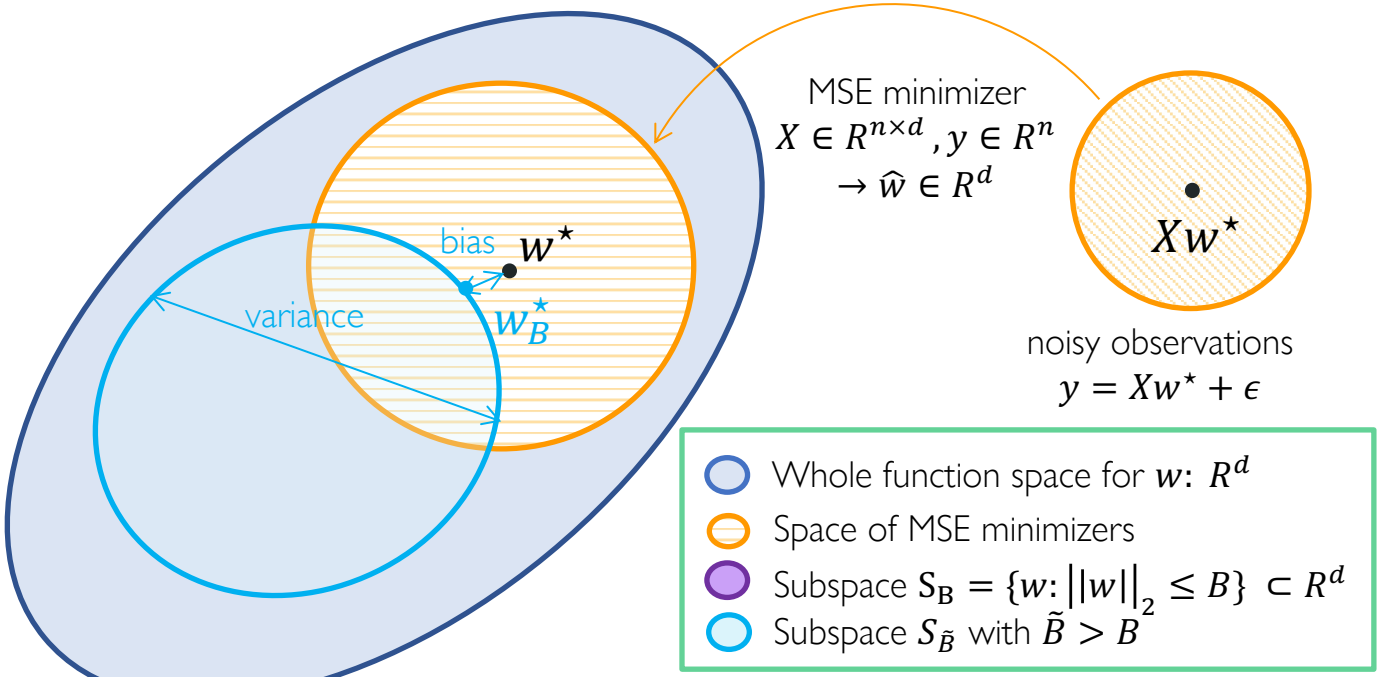
Recap: Bias variance trade-off for $d < n$

How did we achieve the bias-variance tradeoff? → explicit regularization!



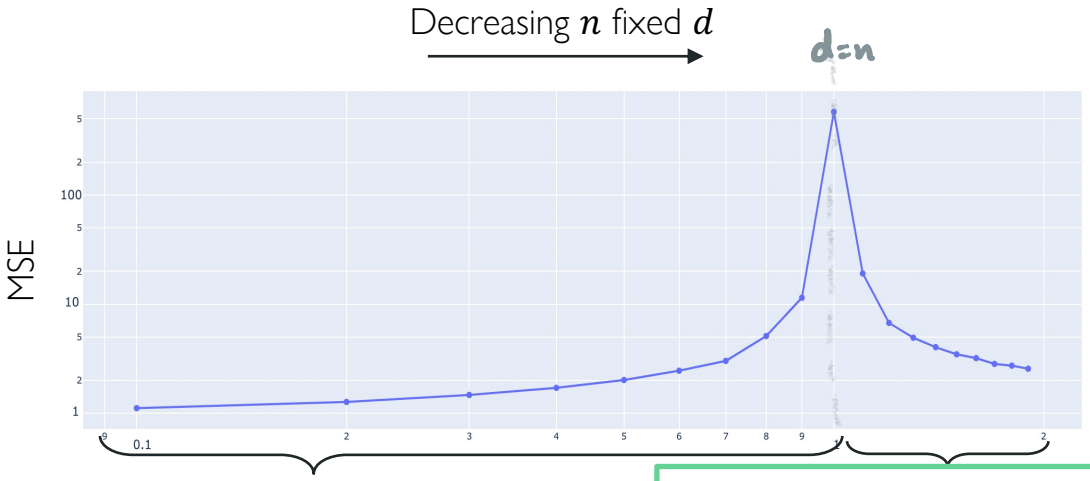
Recap: Bias variance trade-off for $d < n$

How did we achieve the bias-variance tradeoff? → explicit regularization!



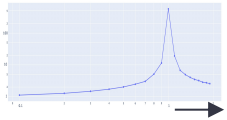
Example I: Linear LS regression for $d \gg n$

Goal now: Compare and build intuition for the two regimes



n decreases for fixed d
→ variance increases

n decreases for fixed d
→ MSE decreases

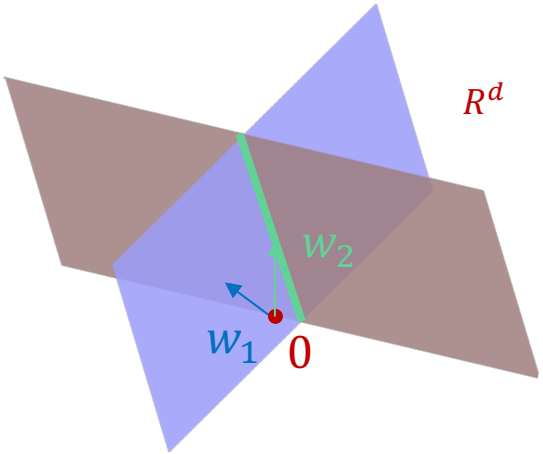


Bias and variance for $d > n$

Bias $||\Pi_X w^\star - w^\star||^2$ increases with $\frac{d}{n}$ since $\Pi_X w^\star$ projects on smaller space → closer to $w^\star$

Variance $E_D||w_n||_2^2$ with $w_n = \text{argmin}_w ||w||_2$ s.t. $n\boxed{\epsilon} = n\boxed{X}\overset{d}{\boxed{w}}\,d$

- is a point in the intersection of n hyperplanes $x_i^T w = \epsilon_i$
- the one that has **minimum distance** $||w_n||_2$ to origin

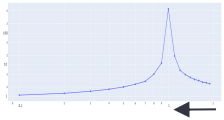


$\frac{d}{n}$ decreases (increasing n fixed d)

→ intersection smaller

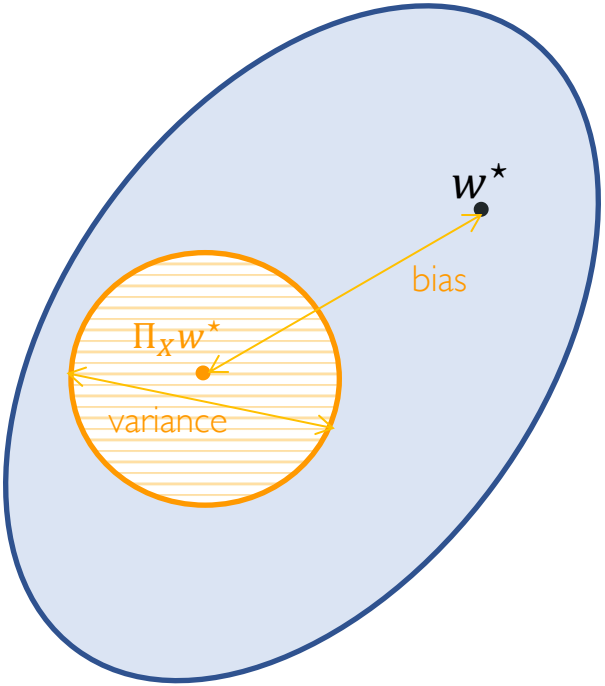
→ minimum distance = $||w_n||_2$ larger

⇨ variance decreases with $\frac{d}{n}$!

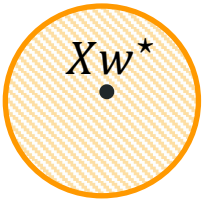


Bias variance trade-off for $d > n$

$\hat{w} = \Pi_X w^\star + w_n$ has bias $\left|\left|\Pi_X w^\star - w^\star\right|\right|^2$ and variance $E_D\left|\left|w_n\right|\right|^2$



min-ℓ_2-norm interpolators:
$X \in R^{n \times d}, y \in R^n \rightarrow \hat{w} \in R^d$

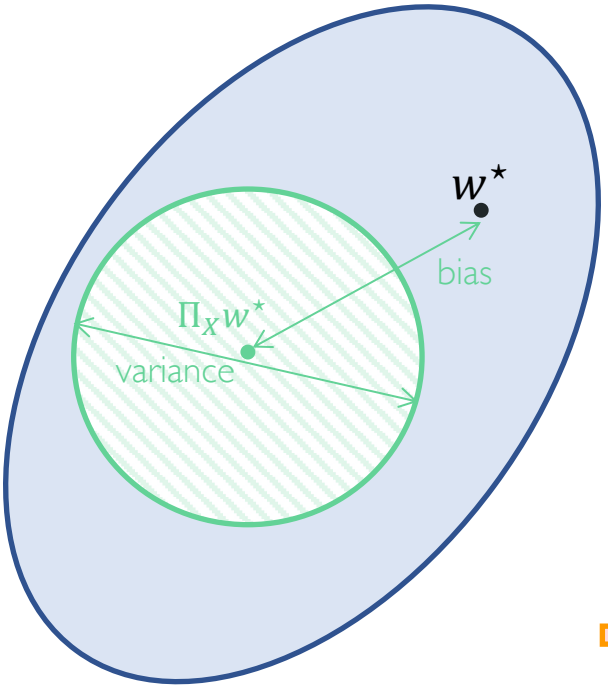


noisy observations
$y = Xw^\star + \epsilon$

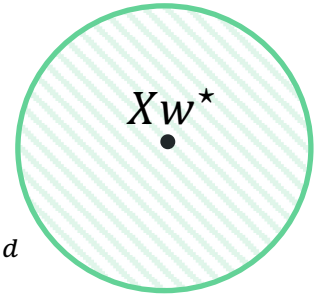
Whole function space R^d
Effectively reached for small n

Bias variance trade-off for $d > n$

$\hat{w} = \Pi_X w^\star + w_n$ has bias $\left|\left|\Pi_X w^\star - w^\star\right|\right|^2$ and variance $E_D\left|\left|w_n\right|\right|^2$



min-ℓ_2-norm interpolators:
$X \in R^{n \times d}, y \in R^n \rightarrow \hat{w} \in R^d$



noisy observations
$y = Xw^\star + \epsilon$

- Whole function space R^d
- Effectively reached for small n
- Effectively reached for large n

⇒ Bias variance trade-off via $\frac{d}{n}$!

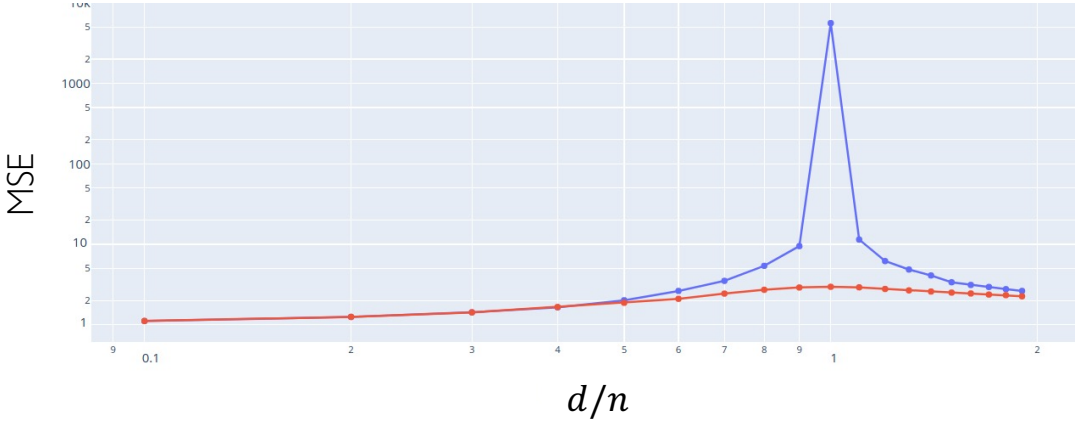
High-dimensional (non)-asymptotic rates

- for min-ℓ_2-norm and different covariances Σ_d
  - $d \asymp n$: Hastie et al. '19
  - $d \asymp n \log n$: Bartlett et al. '20; Muthukumar et al. '20; Koehler et al. '21
- for min-ℓ_1-norm for sparse ground truth $\Sigma_d = I_d$
  - $d \asymp n \log n$: Chinot et al. '20, WD**Y** '21 (in preparation)
- Related work on logistic regression: Deng et al. '19, Chinot et al. '21
- For adversarial robustness: Javanmard et al. '20, DTAH**Y** '21

*This list is by no means complete!

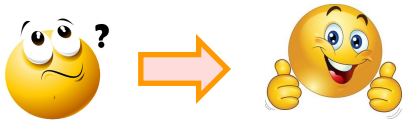
Addendum: What to do in practice?

… just regularize!



Plan for Part 1: high-dimensional regression

- Regression in the modern data regime
- Two examples where classical intuition fails
  - Example I: Linear models where $p = d$
    Minimum-ℓ_2-norm interpolation when $d \asymp n$
    Intuition in the modern regime: larger $n \to$ larger variance

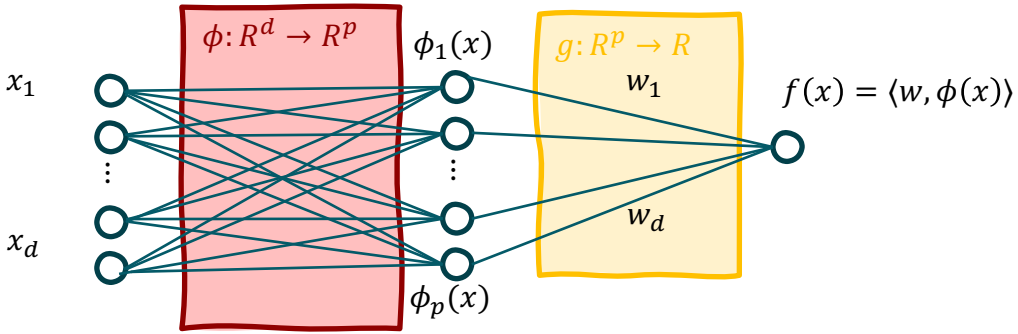


  - Example II: Nonlinear models via kernels with $p = \infty$
    Kernel estimators for $d^\alpha \asymp n$ for large d
    Intuition for fixed d: you can learn nonlinear functions

Intuition from classical theory does not hold!

Recap: Kernel regression

- Data generation: $x \sim P_x$, $y = f^\star(x) + \epsilon$ with $x \in R^d$ and $y \in R$
- Observe: n i.i.d. data points in training set D
- Goal: Find $\hat{f}_D$ that is close to $f^\star$ in kernel class F induced by a kernel $K(x, x') = \langle \phi(x), \phi(x') \rangle$ with $\phi(x) \in R^p$



Recap: Kernel regression

- Data generation: $x \sim P_x$, $y = f^\star(x) + \epsilon$ with $x \in R^d$ and $y \in R$
- Observe: n i.i.d. data points in training set D
- Goal: Find $\hat{f}_D$ that is close to $f^\star$ in kernel class F induced by a kernel $K(x, x') = \langle \phi(x), \phi(x') \rangle$ with $\phi(x) \in R^p$

Consider universal kernel estimators for $p \to \infty$. Implementations yield

- Avoiding perfect fitting: kernel ridge regression

$$\hat{f}_D = \operatorname{argmin}_{f \in F} \sum_i^n \big(y_i - f(x_i)\big)^2 + \lambda \big|\big|f\big|\big|_F^2$$

- Perfect fitting: minimum-Hilbert-norm interpolator

$$\hat{f}_D = \operatorname{argmin}_{f \in F} \big|\big|f\big|\big|_F \text{ s.t. } y_i = f(x_i) \text{ for all } i$$

(the solution of gradient descent on square loss upon convergence)

Kernels and neural networks– previous work

- **Practice:** neural networks can learn highly nonlinear functions very well

  But don't know which interpolating solution it finds!

- **Theory:** kernel estimators can learn arbitrary nonlinear functions

  solution of simple convex problem → analyzable (a lot of previous work)

  where $||\cdot||_K$-norm induces structure dependent on K (e.g. smoothness)

- **Recent trend:** infinite-width neural networks (NN) behave like certain kernels (NTK)

  → use kernel learning to understand why NN work well*

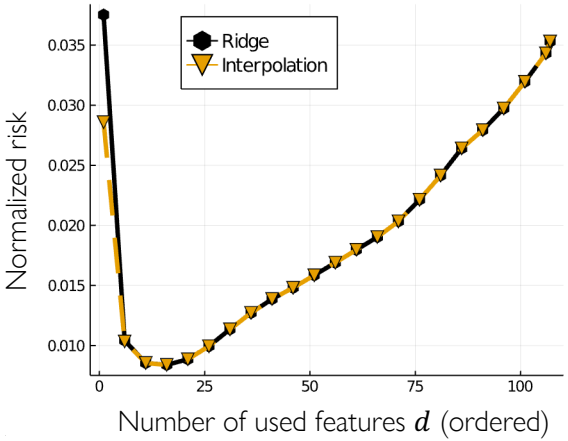
We show: Futile effort when considering vanilla fully-connected NN!

*e.g. [Belkin et al. '18, Liang & Rakhlin '20]

… vanilla kernels fail in high dimensions

Vanilla fully connected-NTK behave "similarly" to Laplace kernels[1] - a good thing?

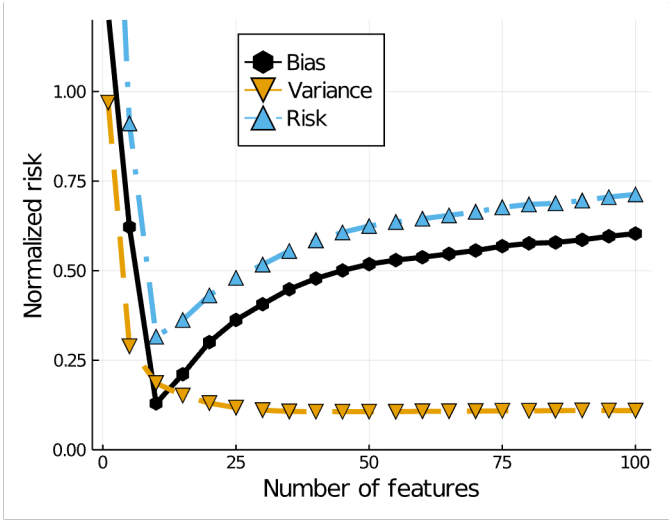
- on CIFAR10: FC-NTK : ~52%[2], Laplace kernel: ~52%[2]
- Laplace kernel for fitting $f^*(x)$: true housing price, $n = 371$ (on basically noiseless data)



[1][Geifman et al. '20, Bietti et al. '20] [2][Lee at al. '20, Belkin et al. '18]

Bias variance trade-off for kernels

Laplace kernel for $f^\star(x) = 0.5 \sum_{i=1}^{4} x_{(2i+1)}^2 - \sum_{i=1}^{4} x_{(2i)}$ for fixed $n = 500$



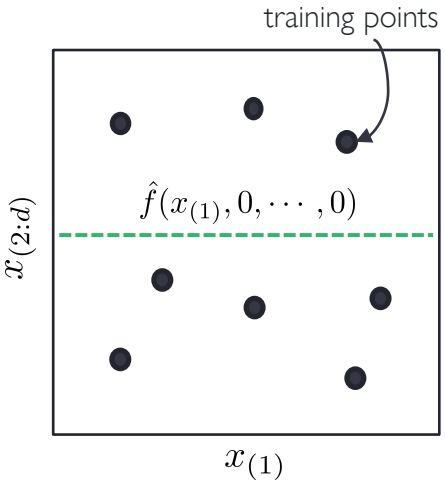
Similar to linear as $\frac{d}{n}$ increases, but bias increase dominates variance decrease!

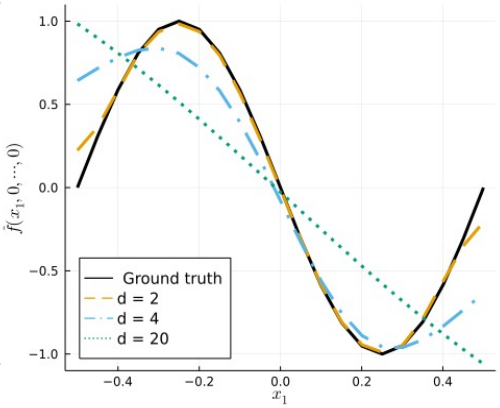
Goal now: Characterize the bias as a function of d vs. n!

Kernels learn low-degree polynomials

Setting

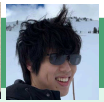
- $x \sim U([-.5, .5]^d)$
- $y = \sin(x_1)$
- $n = 100$ i.i.d.
- Laplace kernel





As dimension grows, the estimator degenerates to a low degree polynomial

Main result: Polynomial approx. barrier

Theorem (DWY '21, ICML) - *simplified, informal version*

Assume simplest setting $x \sim N(0, I_d)$, then as $d, n \to \infty, \frac{d^\alpha}{n} \to c > 0$

$$\left\| \hat{f} - f^\star \right\| \geq \inf_{p \in P_{\leq 2\alpha}} \|f^\star - p\| \quad \text{almost surely}$$

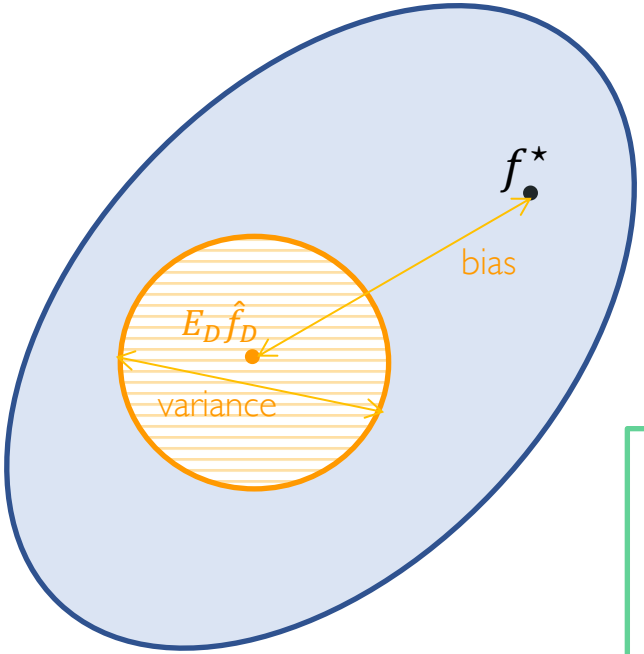
where $P_{\leq 2\alpha}$ is the set of polynomials of degree at most 2α, any $\alpha > 0$

- more generally can assume $\mathrm{x} \sim N(0, \Sigma_d)$ and replace d by $\mathrm{tr}(\Sigma_d)$
- for rotationally invariant kernels $k_\tau(\mathrm{x}, \mathrm{y}) = h\left(\frac{\|x\|^2}{\tau}, \frac{\|y\|^2}{\tau}, \frac{x^\top y}{\tau}\right)$
  - different functions h such as RBF (Laplacian, Gaussian), inner product, fully connected NTK of any depth
  - for different scalings τ

*poly. barrier restricted to particular distr., scaling $\tau = d$: [Ghorbani et al. '19, '20]

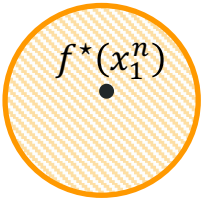
Bias variance trade-off for kernel estimators

Same story as for linear models…



kernel estimators:

$y \in R^n \rightarrow \hat{f}_D \in F$



noisy observations

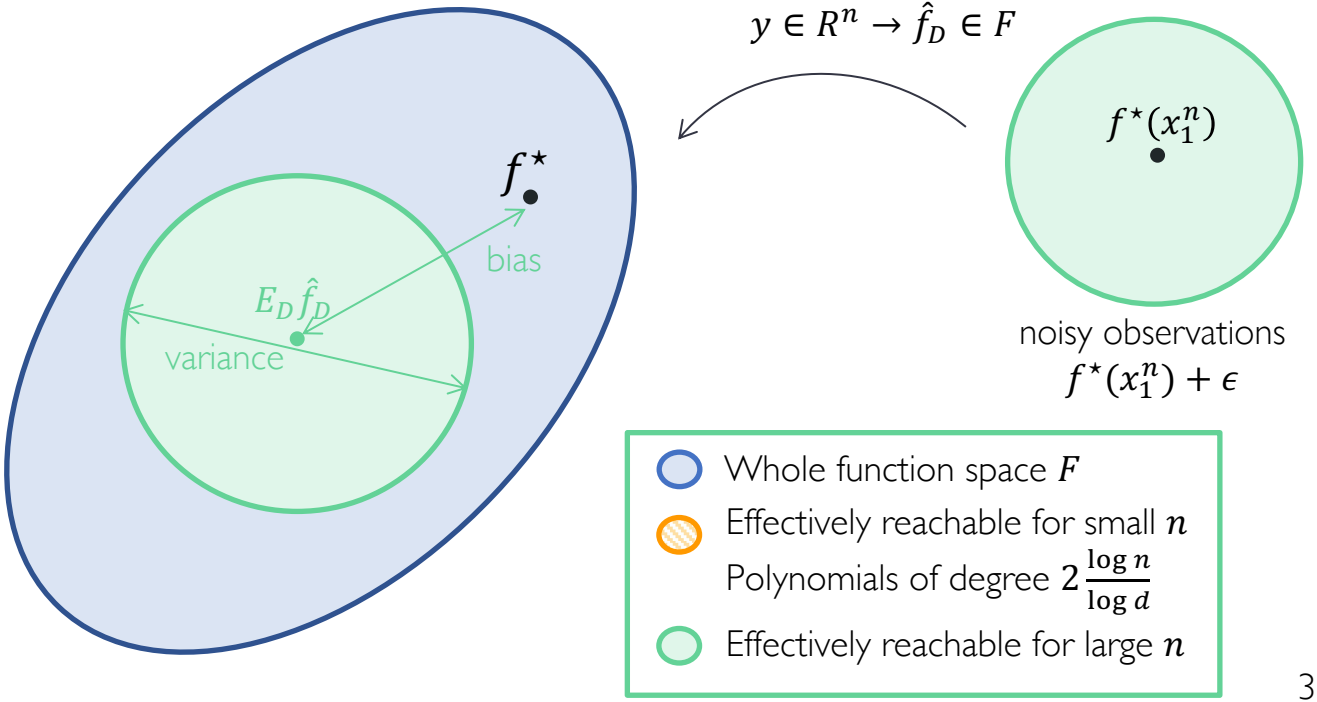
$f^\star(x_1^n) + \epsilon$

- Whole function space F
- Effectively reachable for small n

  Polynomials of degree $2\frac{\log n}{\log d}$

Bias variance trade-off for kernel estimators

Same story as for linear models…

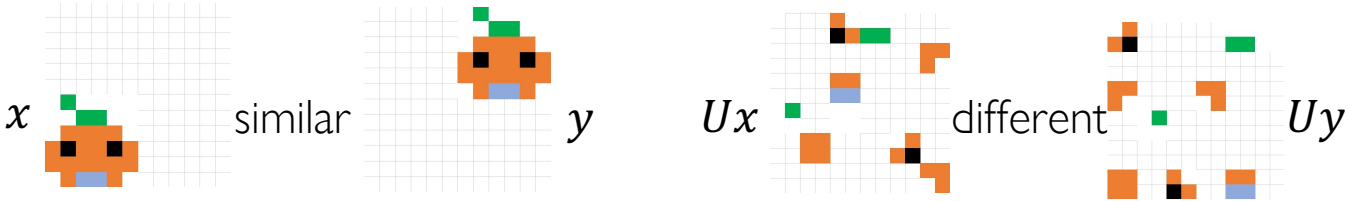


Why rotation invariance increases bias with d

- rotationally invariant kernels should satisfy $k(x, y) = k(Ux, Uy)$
  with U orthonormal matrix (think of U permutation matrix)

- kernel method $\hat{f}$ predicts similar values for similar samples, similarity defined by $k(x, y)$

Example 1: ground truth are human labelers, e.g. images
→ good classifier $\hat{f}$ should find similar what we find similar, i.e. $k(x, y)$ should reflect that



not using spatial structure, pays attention to every pixel independently

Why rotation invariance is bad for large d

- rotationally invariant kernels should satisfy $k(x, y) = k(Ux, Uy)$
  U: orthonormal matrix (e.g. permutation matrix)

- kernel method predicts similar values for similar samples, similarity defined by $k(x, y)$

- further can often write $k(x, y) \sim \sum_{j=0}^{\infty} g_j \left(\frac{x^\top y}{d}\right)^j$ in high dimensions for some g_j

Example II: ground truth depends only on first variable
→ good predictor should map any two x, y with $x_1 = y_1$ to same value, $k(x, y)$ high

However, assume $x_{2:d}, y_{2:d}$ random and large $d \to \frac{x^\top y}{d} \approx \frac{x_1 y_1}{d}$ small → $k(x, y)$ small

generally: low-dimensional features dominated by irrelevant features

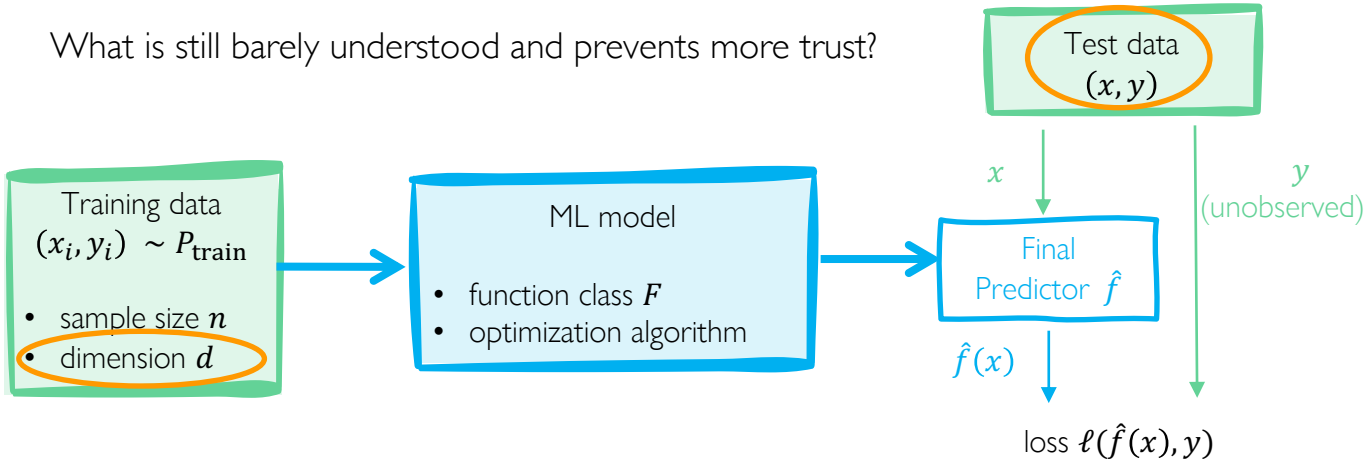
Is there any hope for kernels in large d?

- For images: structured kernels actually work well in practice...?
  - convolutional kernels can achieve ~90% on CIFAR10[1]
  - can we analyze the asymptotic limits of such kernels?
- For functions depending on few variables:
  - nonlinear feature selection before kernel
  - or A(utomatic) R(elevance) D(etermination) kernels[2]

[1]e.g. [Shankar et al. '20], [2]e.g. [MacKay '94]

Supervised learning for modern ML

What is still barely understood and prevents more trust?



Two important settings for which understanding is missing

1. High-dimensionality: how small average loss can be when d large
2. Reliability: how model acts when test data $(x, y) \nsim P_{train}$

What is this traffic sign?

Fails for small perturbations that don't change the class

*[Eykholt et al. '17, Sitawarin et al. '18]

Which object is this?

self-driving car

truck

sky

Fails on unseen scenarios

*based on a reported Tesla accident in 2016

Which disease does this patient have?

bacterial pneumonia

COVID19 pneumonia

the AI doctor

bacterial pneumonia

bacterial pneumonia
with high confidence

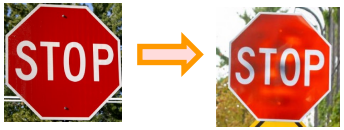
Fails to be uncertain on novel classes

*[Cao et al. '20]

Reliability when $(x, y) \nsim P_{train}$

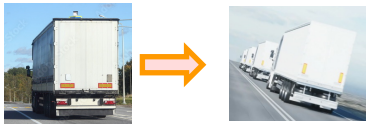
Adversarially transformed

$\mathrm{T}(x)$ with $x \sim P_{\mathrm{train},x}$



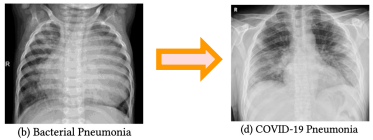
Same $P_{y|x}$ and classes

$x \sim P_{\mathrm{test},x} \neq P_{\mathrm{train},x}$



Same $P_{y|x}$ but new classes

$y \notin \mathrm{supp}(P_{\mathrm{train,y}})$



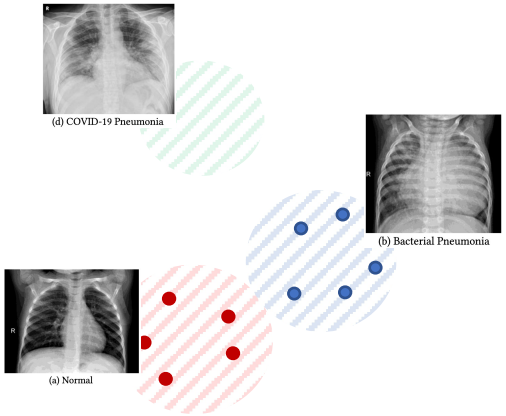
Goals:

- robust generalization: have low expected error on x
- calibration: predictive uncertainty should be accurate
- detect novel class samples

Plan for Part 2: novel class detection

- Easy vs. hard novel class (OOD) detection
- OOD detection using ensembles
- Unknown OOD setting
- Idea: regularized diversity with unlabeled data

Novel classes in the test set



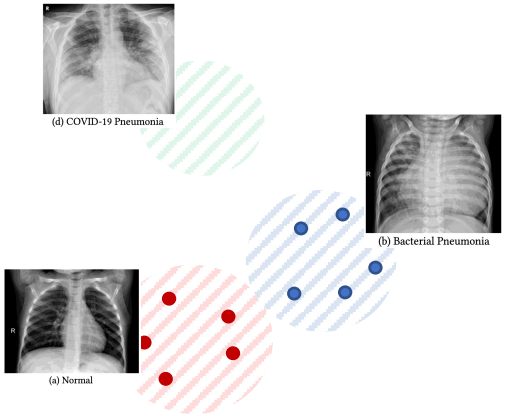
training points

supp_{ID} : training distribution

supp_{OOD}: novel classes (out-of-distribution / OOD)

- Given labeled training points from $supp_{ID}$, test point $x \in supp_{test} = \text{supp}_{ID} \cup \text{supp}_{OOD}$
- Goal: Flag if $x \in supp_{OOD}$, predict if $x \in supp_{ID}$
  (also known as anomaly detection, open set recognition, one-class classification)

Two types of test statistics



training points

$\mathbf{supp}_{ID}$: training distribution

$\mathbf{supp}_{OOD}$: novel classes (out-of-distribution / OOD)

Can view it as classifying between OOD and ID without OOD labels

- view as density estimation problem → flag if probability of $\boldsymbol{x}$ too low
- by-product of predictive uncertainty problem → flag if uncertainty too high

works better with neural networks

Easy vs. hard novel image classes

$supp_{ID}$: CIFAR10 classes 1-5

easy OOD: different dataset

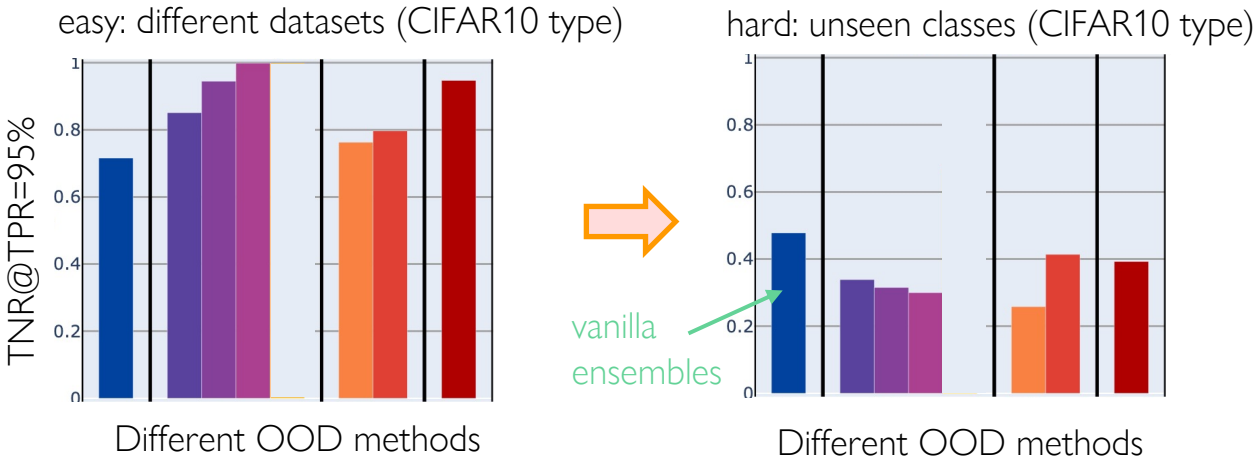
- $supp_{OOD}$: SVHN classes 6-10
- ID far from OOD data

hard OOD: unseen classes

- $supp_{OOD}$: CIFAR10 classes 6-10
- ID close to OOD data

SOTA methods fail for novel classes

- True positive rate (TPR): percentage of truly novel classes marked as OOD
- True negative rate (TNR): percentage of seen classes marked as ID



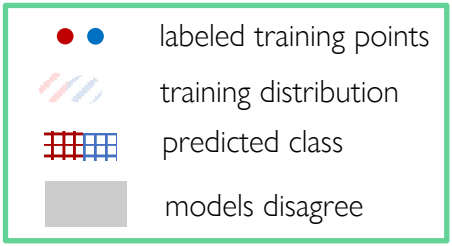
large drop for unseen (novel) class from the same dataset!

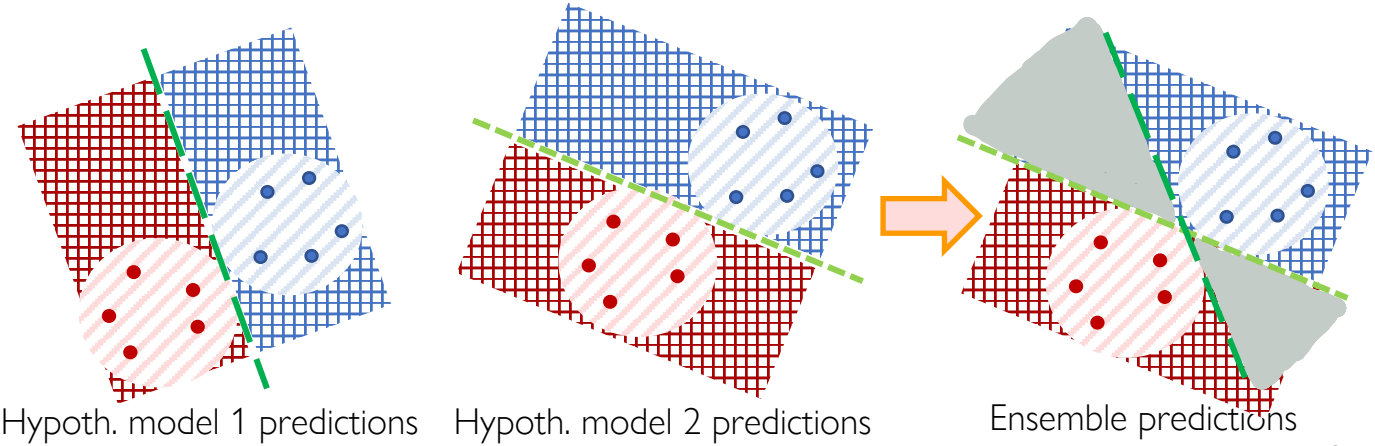
Novel class detection using ensembles

For a binary classification problem consider two models that

- have good validation accuracy on old classes
- are different outside of training distribution

→ can mark samples with disagreement as novel



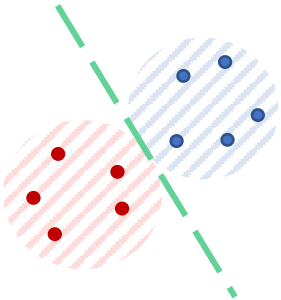


Hypoth. model 1 predictions | Hypoth. model 2 predictions | Ensemble predictions

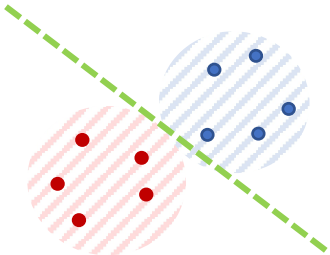
Vanilla ensembles are not diverse enough

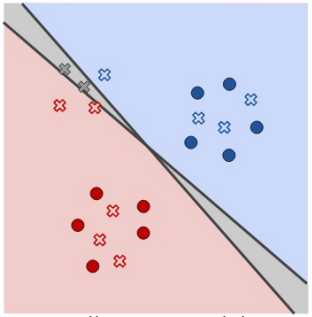
SOTA neural network ensembles tend to agree where they can

Model 1: random init 1

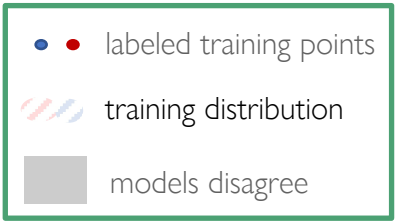


Model 2: random init 2





vanilla ensemble

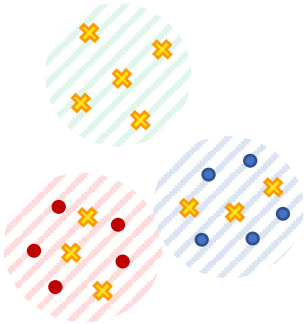


Hard OOD requires disagreement on larger region

→ more diverse ensembles!

Our setting: Unknown OOD

Often unlabeled test data can be available including OOD (PU-learning)



labeled training points

$supp_{ID}$: training distribution

$supp_{OOD}$: novel classes

some unlabeled points from $supp_{ID} \cup supp_{OOD}$ *(test set)*

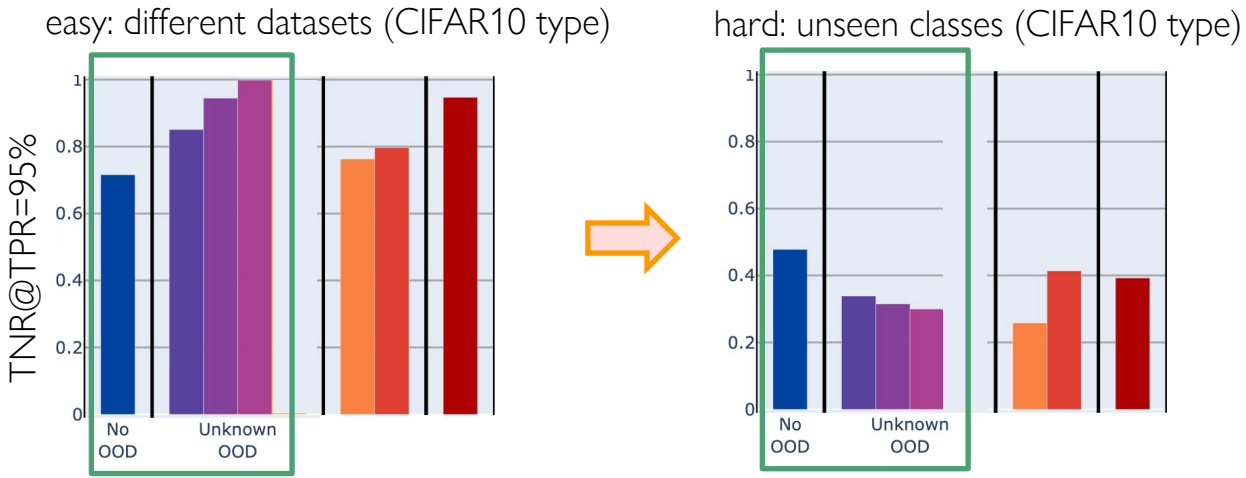
In medical example

- test set: unlabeled X-rays collected during the week when new disease arrives
- even though predict using old model, valuable to detect new diseases by end of week

Is the problem now trivial?

Previous unknown OOD methods still fail

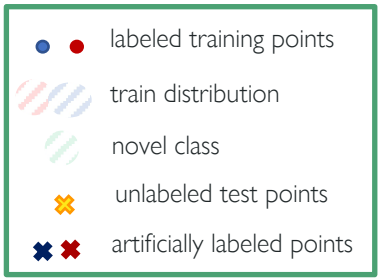
- True positive rate (TPR): percentage of truly novel classes marked as OOD
- True negative rate (TNR): percentage of seen classes marked as ID

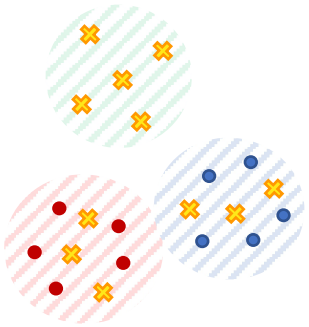


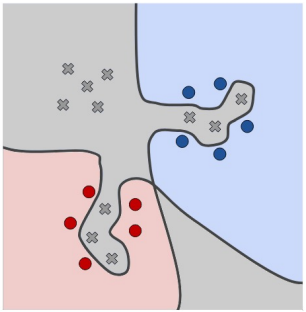
Not obvious how to leverage unknown OOD!

*[Kiryo et al. '17, Yu et al. '19, based on Lee '18]

Idea: regularized diversity with unlabeled data



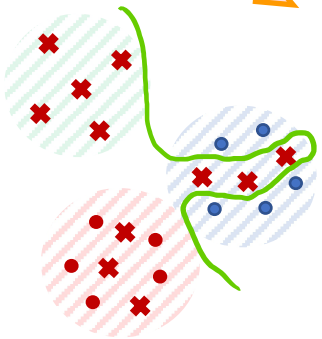


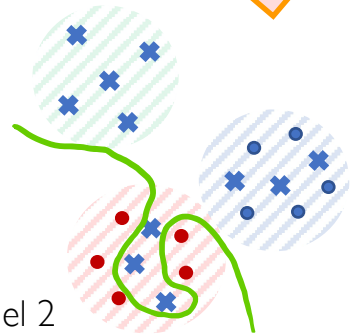


without regularization

label as

label as

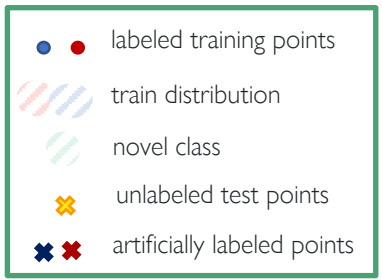


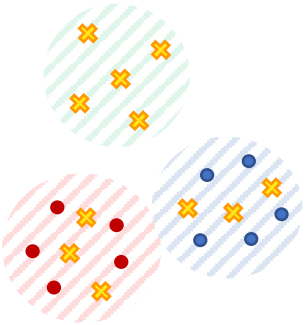


Model 1

Model 2

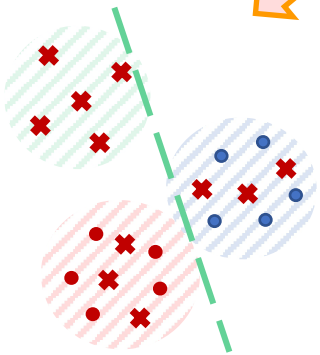
Idea: regularized diversity with unlabeled data

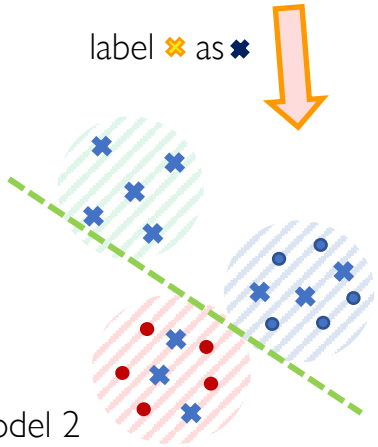




label as

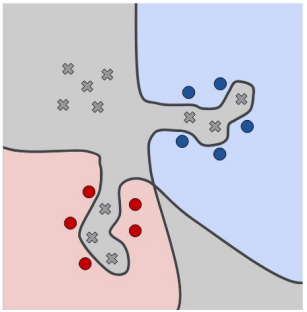
label as



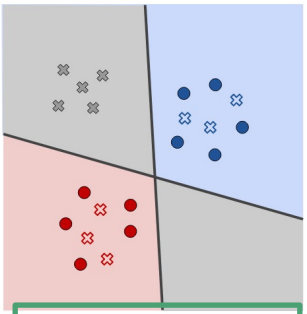


Model 1

Model 2



without regularization

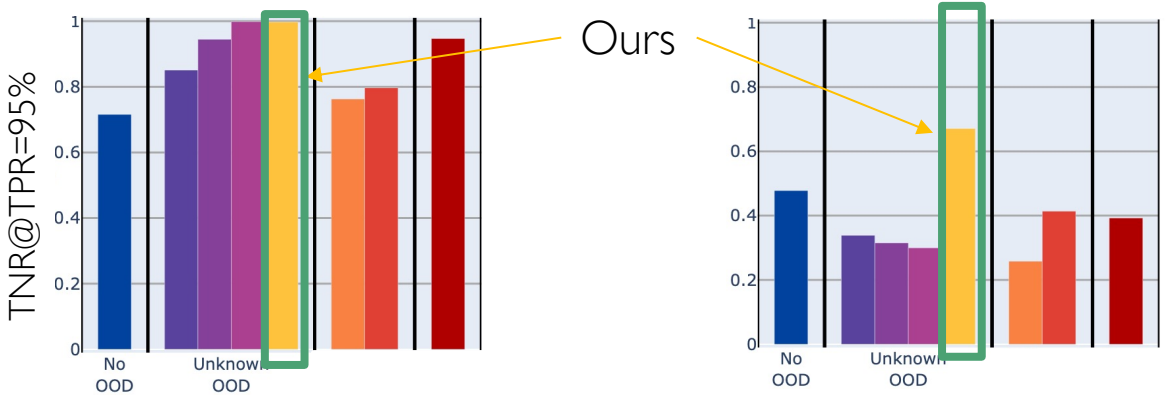


with regularization

Learning ensembles that disagree

Algorithm (TSY '20) – ensembles with regularized disagreement

1. Learn model that fit different artificial labels c = 1 to K on unlabeled set but are regularized to have high validation accuracy (e.g. fine-tuning + early stopping)
2. For new point x, average pairwise disagreement between classifiers c = 1 to K
3. flag as OOD if disagreement > some threshold

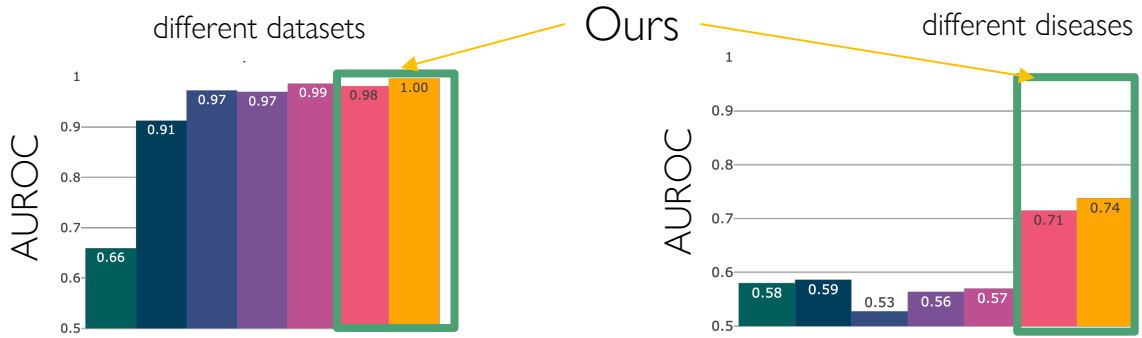


Learning ensembles that disagree

Algorithm (TSY '20) – ensembles with regularized disagreement

1. Learn model that fit different artificial labels c = 1 to K on unlabeled set but are regularized to have high validation accuracy (e.g. fine-tuning + early stopping)
2. For new point x, average pairwise disagreement between classifiers c = 1 to K
3. flag as OOD if disagreement > some threshold

Novel classes in chest X-ray + retinal datasets




Thanks!