Measuring Performance

CS 474

Dr. Porter Jenkins

D&C 130: 18-19

18 Whatever principle of [a]intelligence we attain unto in this life, it will rise with us in the [b]resurrection.

19 And if a person gains more [a]knowledge and intelligence in this life through his [b]diligence and obedience than another, he will have so much the [c]advantage in the world to come.

Overfitting

Overfitting

Phenomenon of machine learning algorithms to learn patterns specific to training data that do not generalize to unseen (test) data

In general, as we increase the capacity of the model, and broaden the family of possible functions we can represent, we increase the risk of overfitting

Never underestimate the ability of your neural network to overfit!!

Overfitting

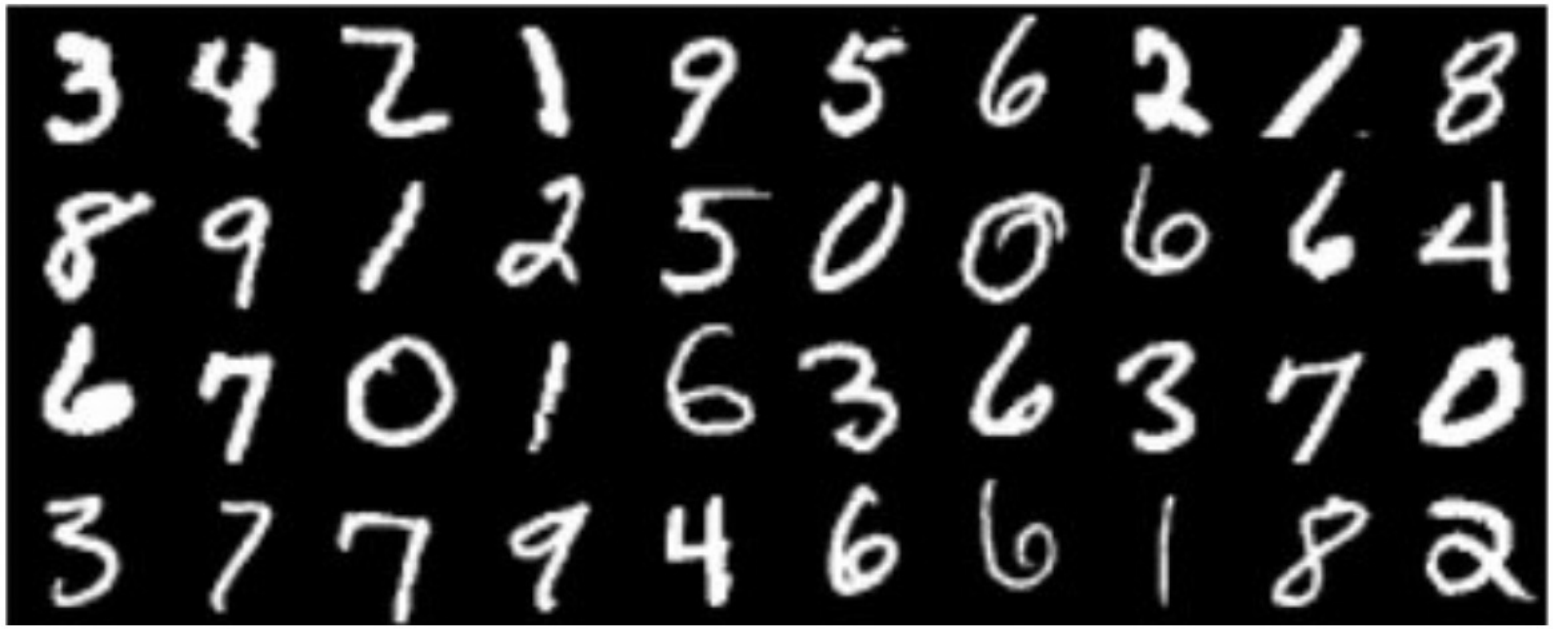
Overfitting

Demo:
https://colab.research.google.com/drive/1hBcLgVu8JBJR3_E-vOSoy63XBQTpeG_O?authuser=2

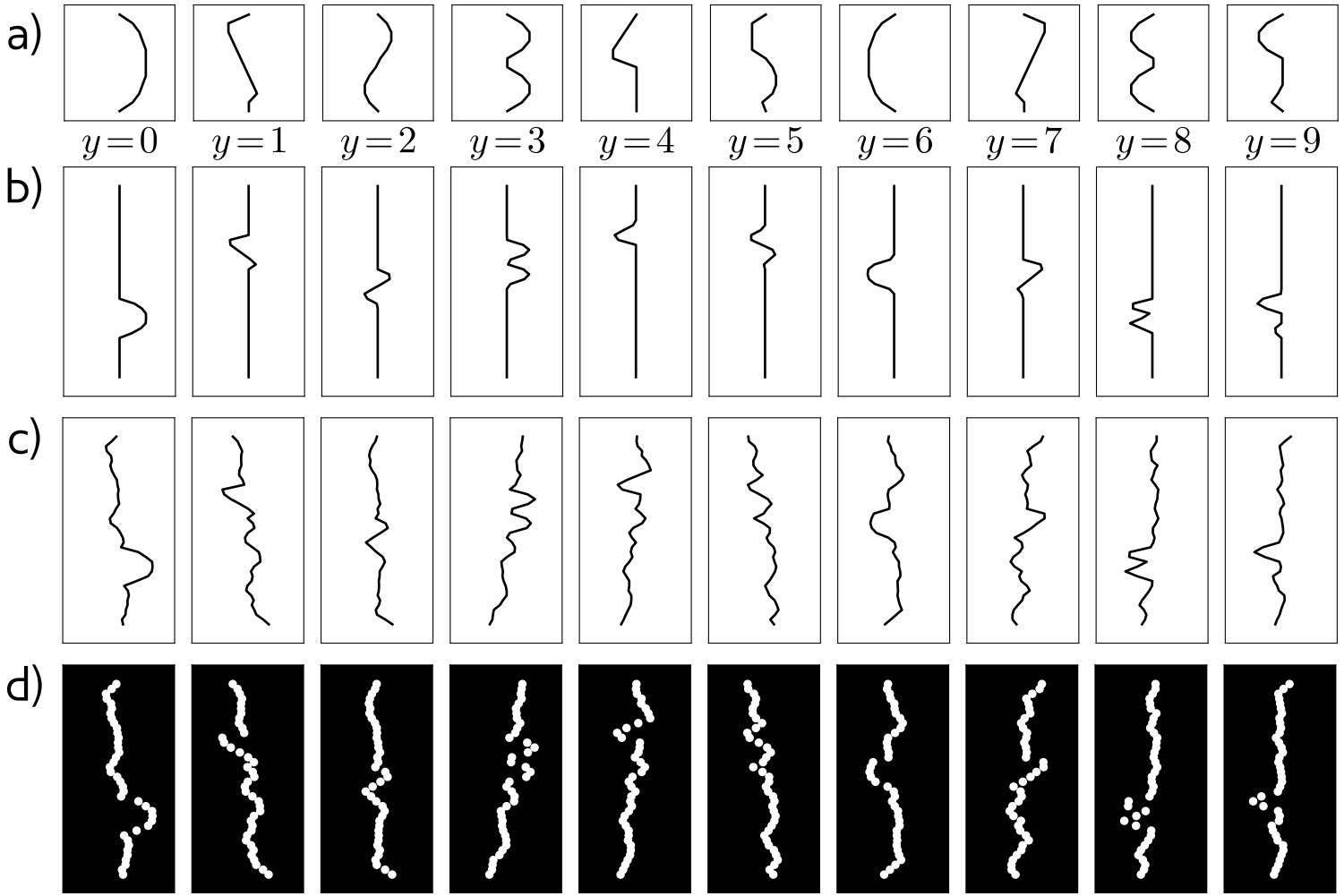
Measuring performance

- MNIST1D dataset model and performance
- Noise, bias, and variance
- Reducing variance
- Reducing bias & bias-variance trade-off
- Double descent
- Curse of dimensionality & weird properties of high dimensional space
- Choosing hyperparameters

MNIST Dataset



MNIST 1D Dataset

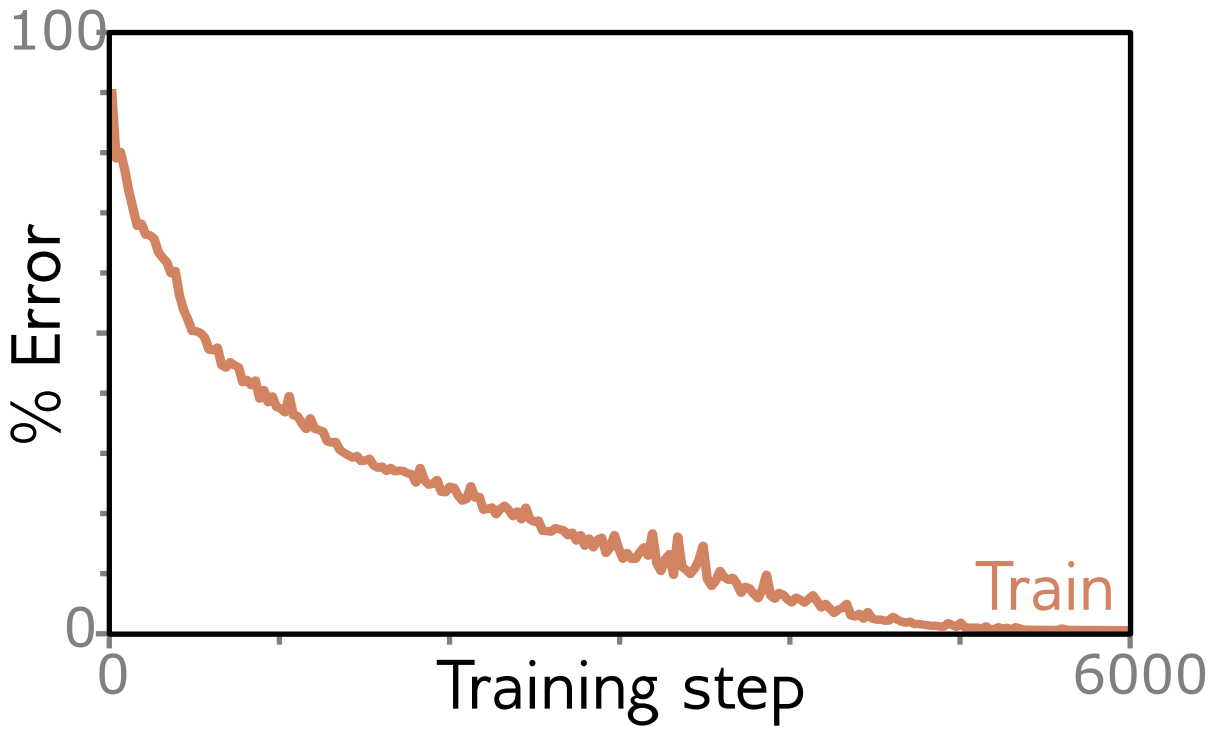


Network

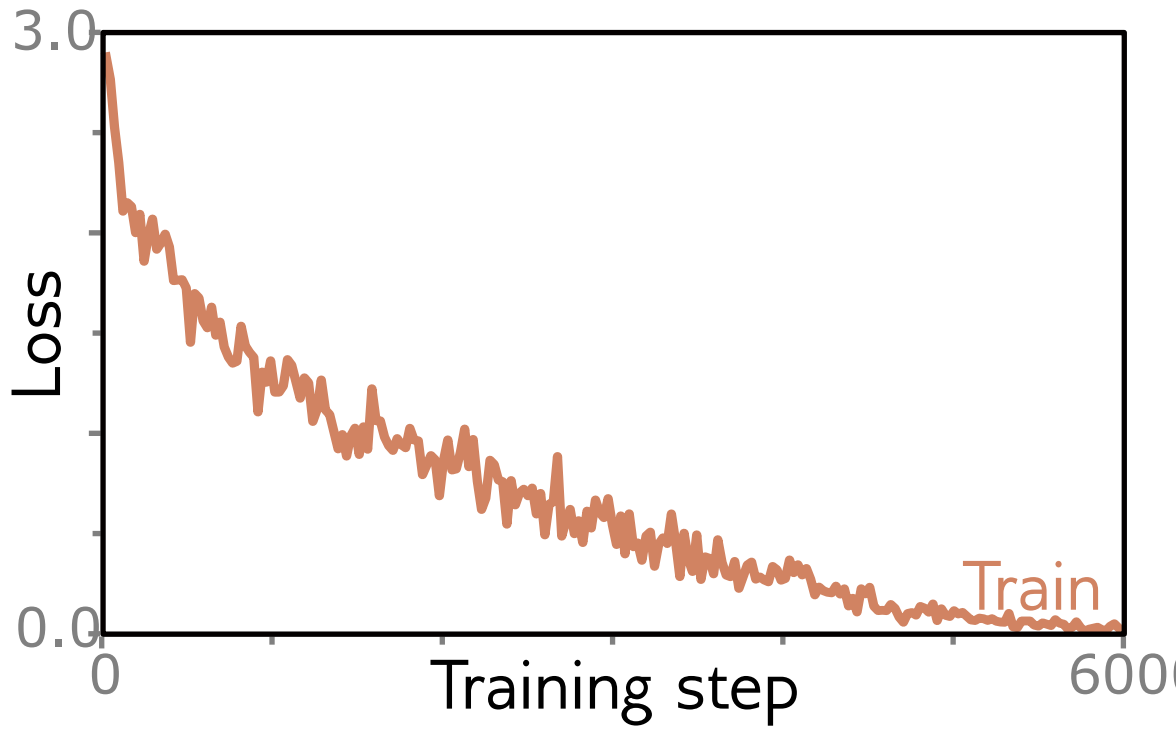
- 40 inputs
- 10 outputs
- 4000 training examples (~400 training examples per class)
- Two hidden layers
  - 100 hidden units each
- SGD with batch size 100, learning rate 0.1
- 6000 steps (60 Epochs)

Results

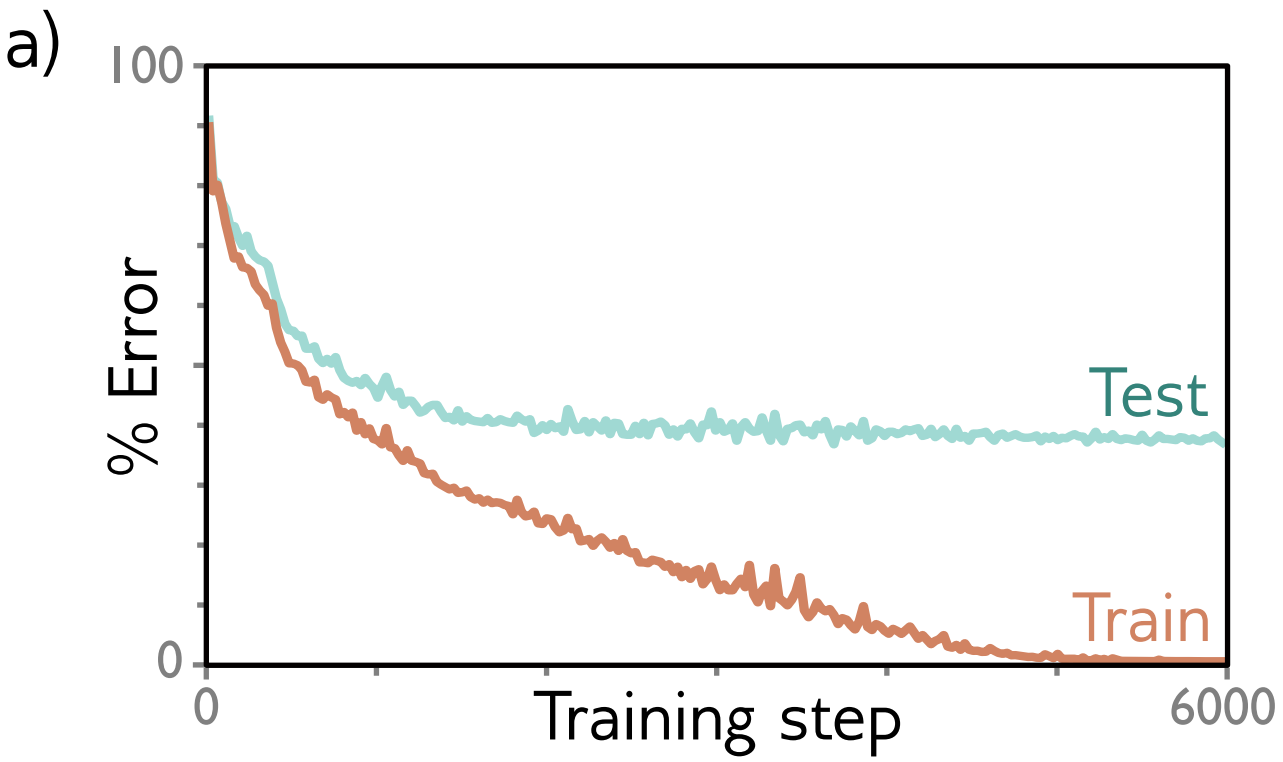
a)

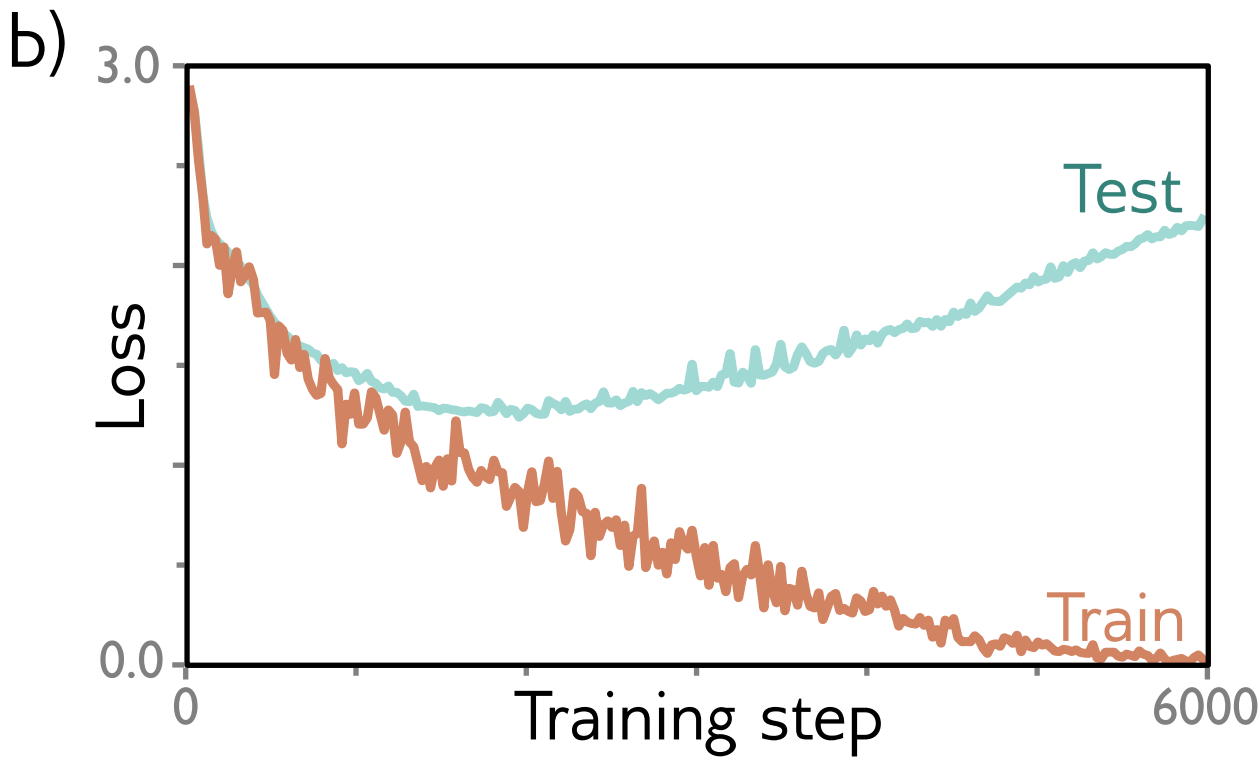

100
% Error
0
0
Training step
6000
Train

b)

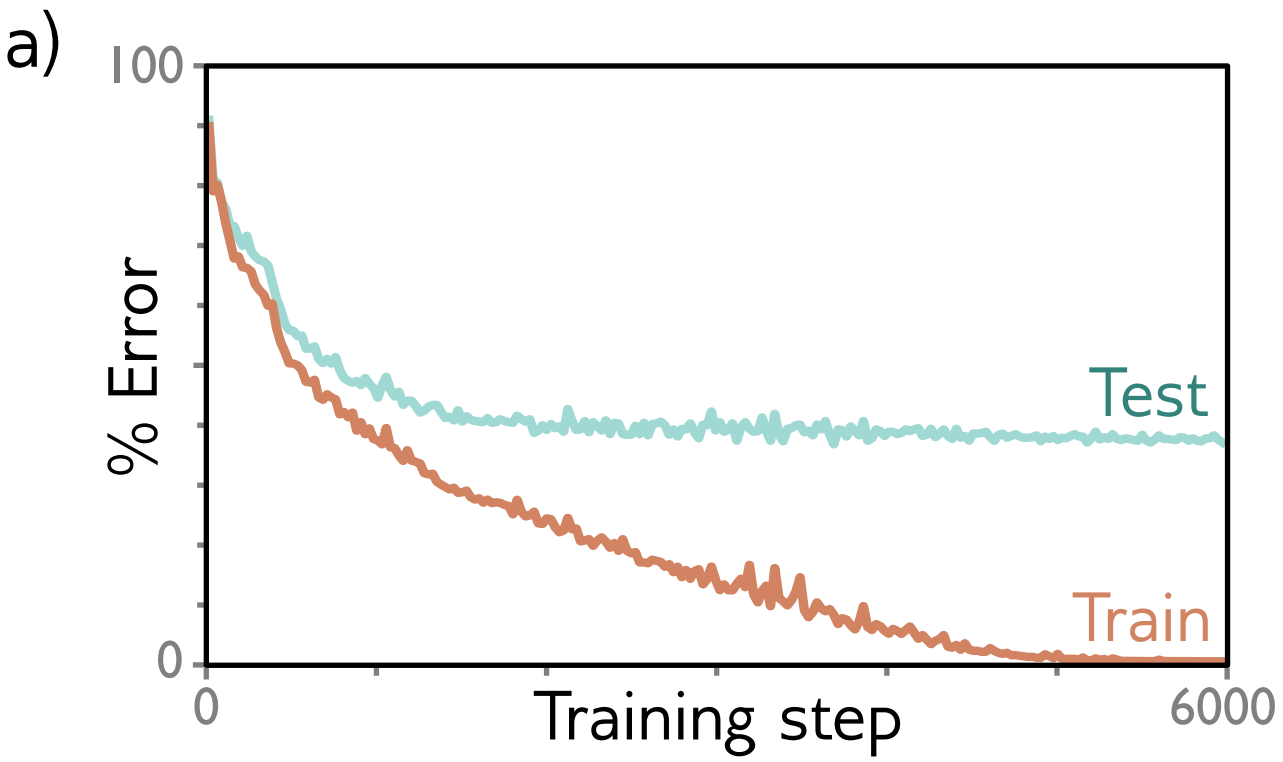

3.0
Loss
0.0
0
Training step
Train

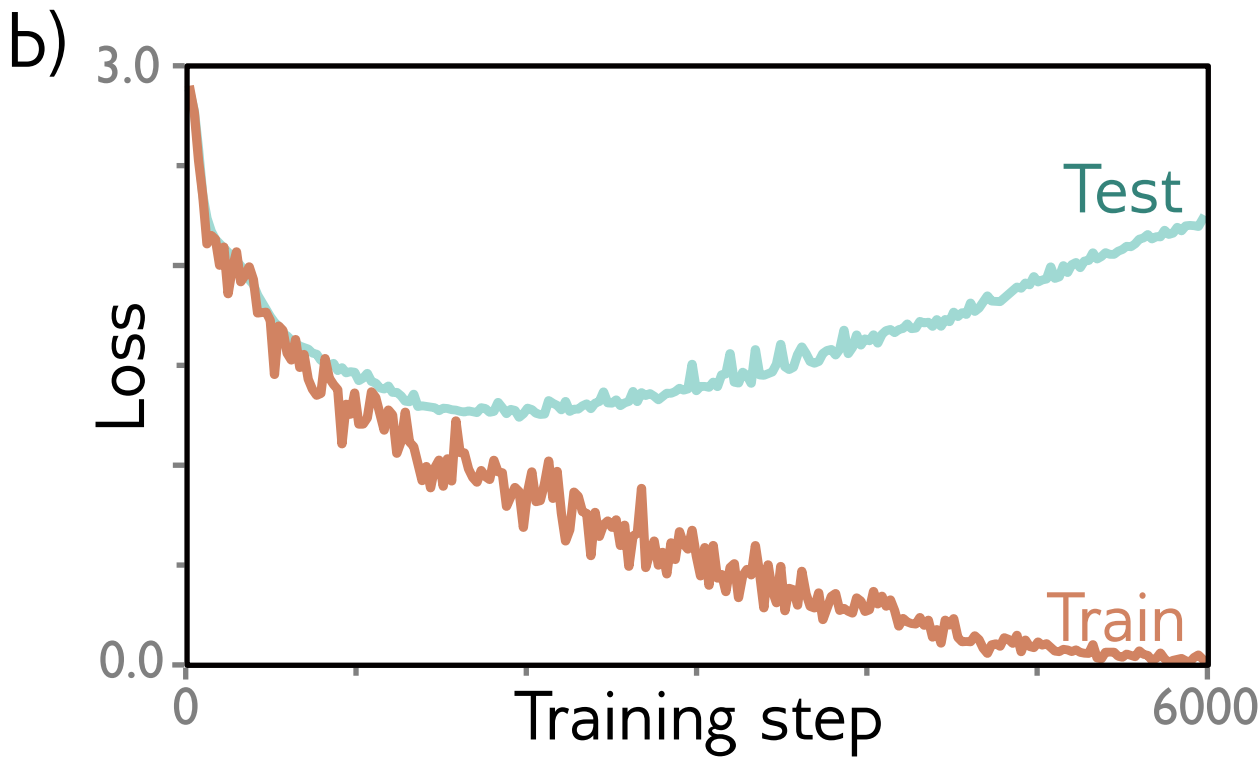
Need to use separate test data





Need to use separate test data



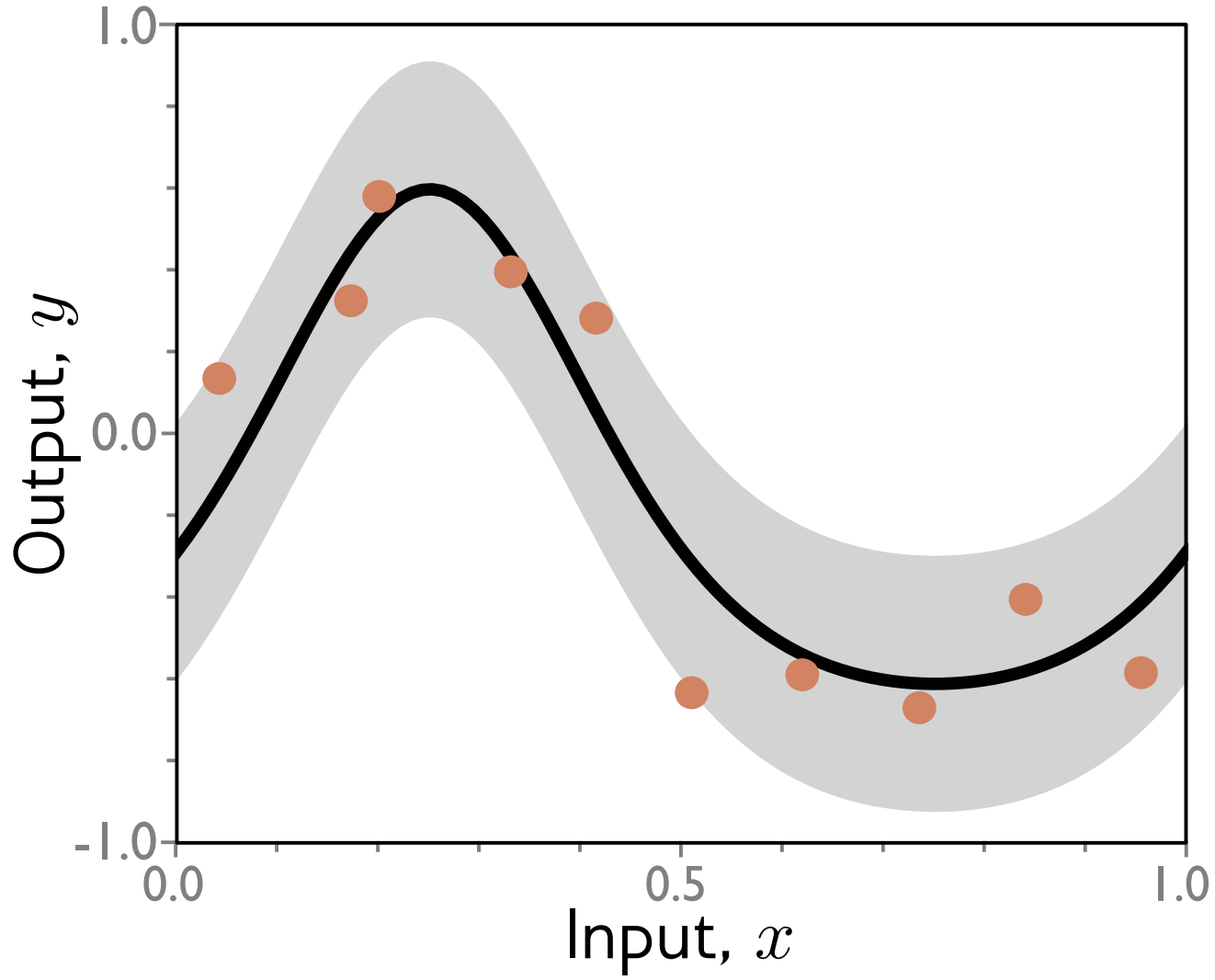


The model has not generalized well to the new data

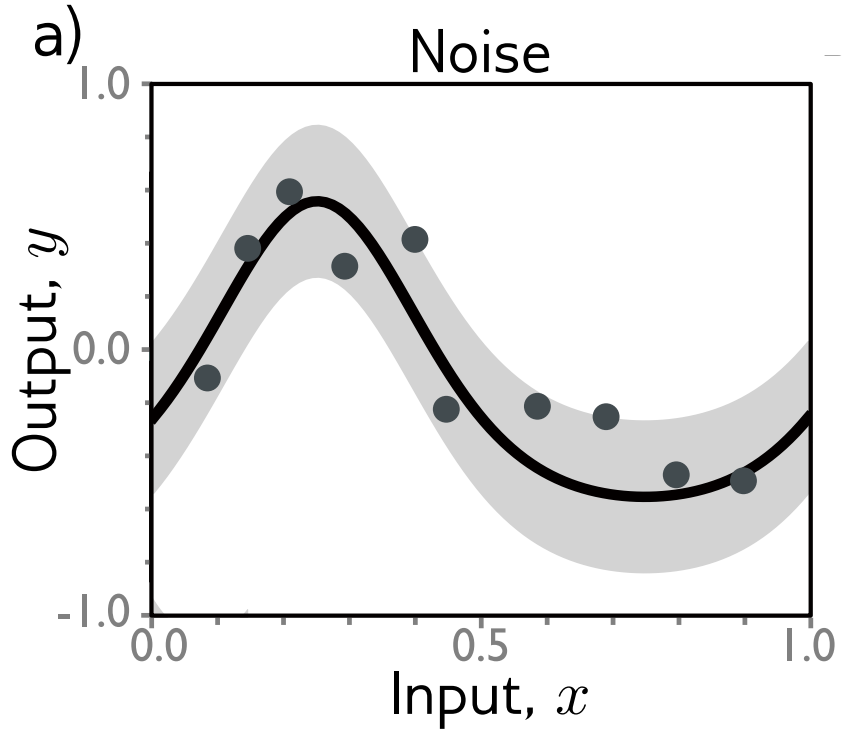
Measuring performance

- MNIST1D dataset model and performance
- Noise, bias, and variance
- Reducing variance
- Reducing bias & bias-variance trade-off
- Double descent
- Curse of dimensionality & weird properties of high dimensional space
- Choosing hyperparameters

Regression example



Noise, bias, and variance



- Noise in measurements
- Some variables not observed
- Data mislabeled

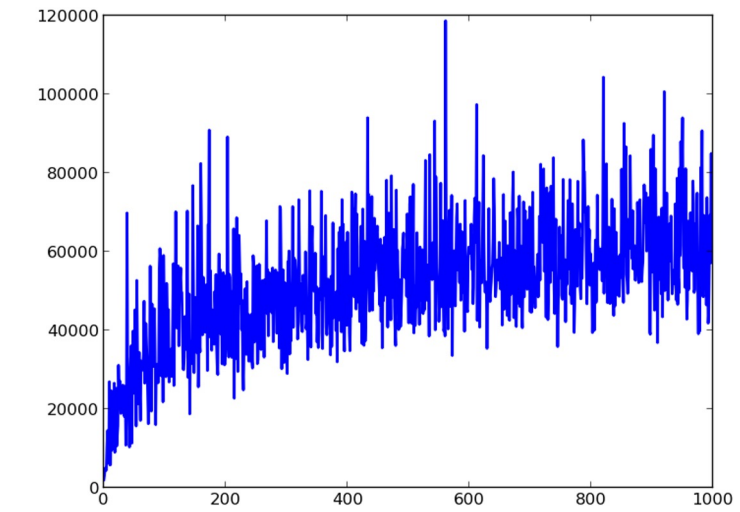
Noise is inherent uncertainty in the true mapping from input to output

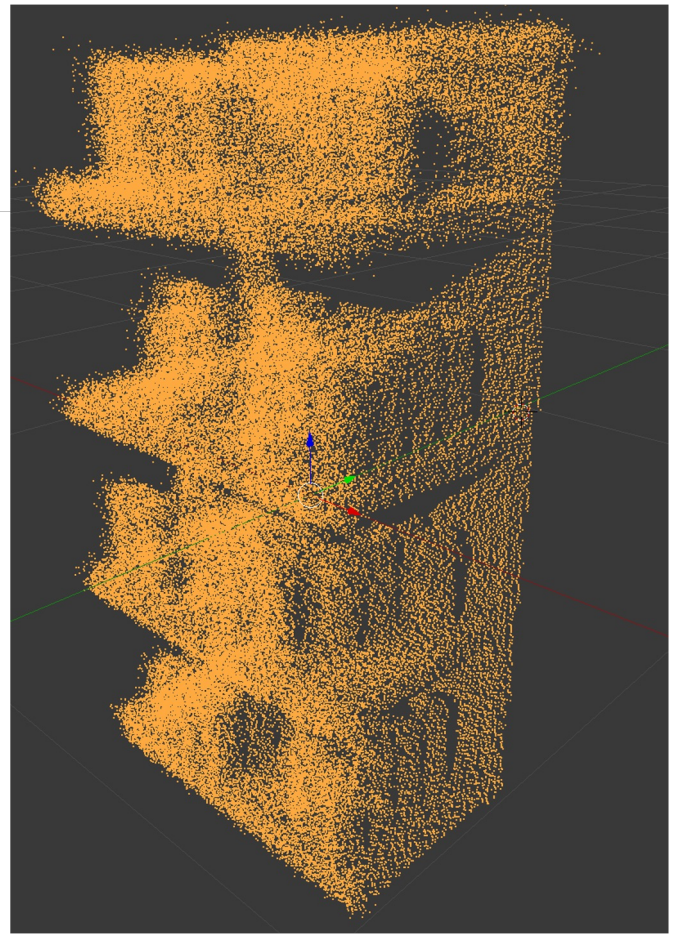
Noise = Aleatoric Uncertainty

Noise that are inherent in the **observations**

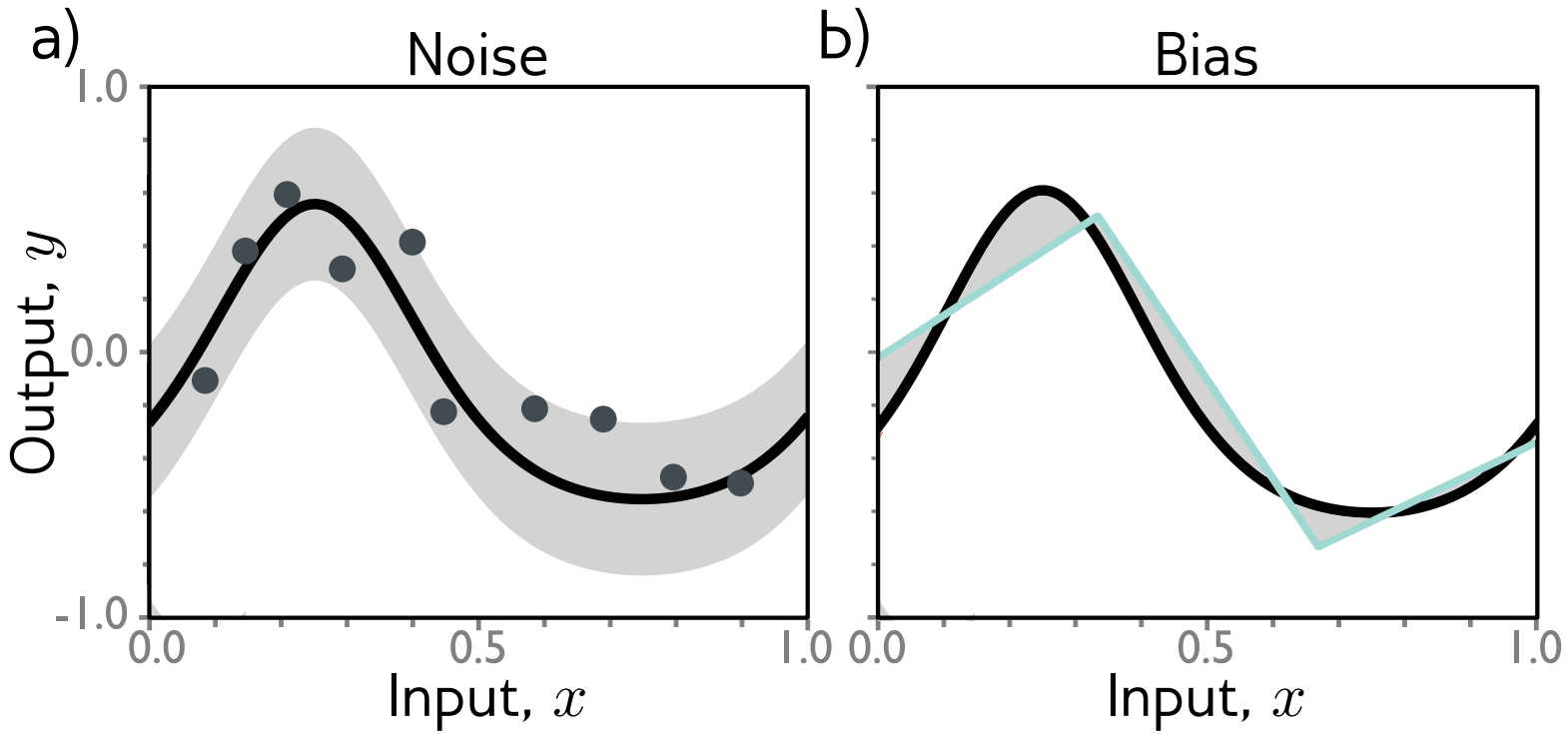
Cannot be reduced with more data

Example: a **sensor noise** or motion noise, resulting in uncertainty which cannot be reduced **even if more data were to be collected.**



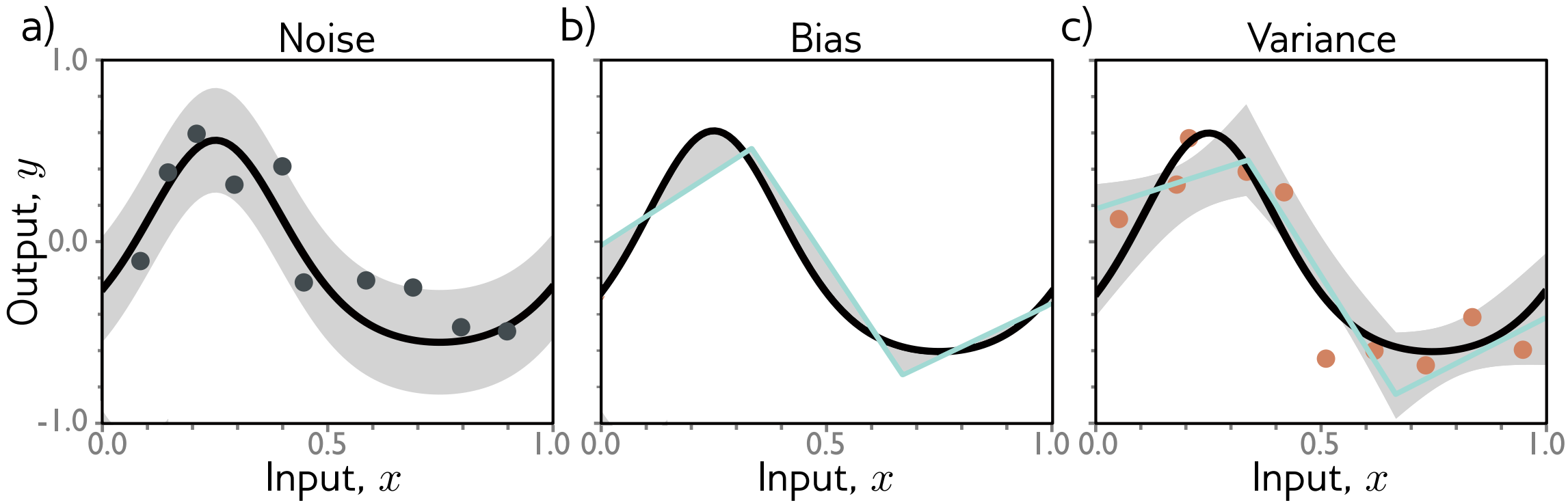


Noise, bias, and variance



Bias is systematic deviation from the mean of the function we are modeling due to limitations in our model

Noise, bias, and variance



Variance is the uncertainty in fitted model due to choice of training set

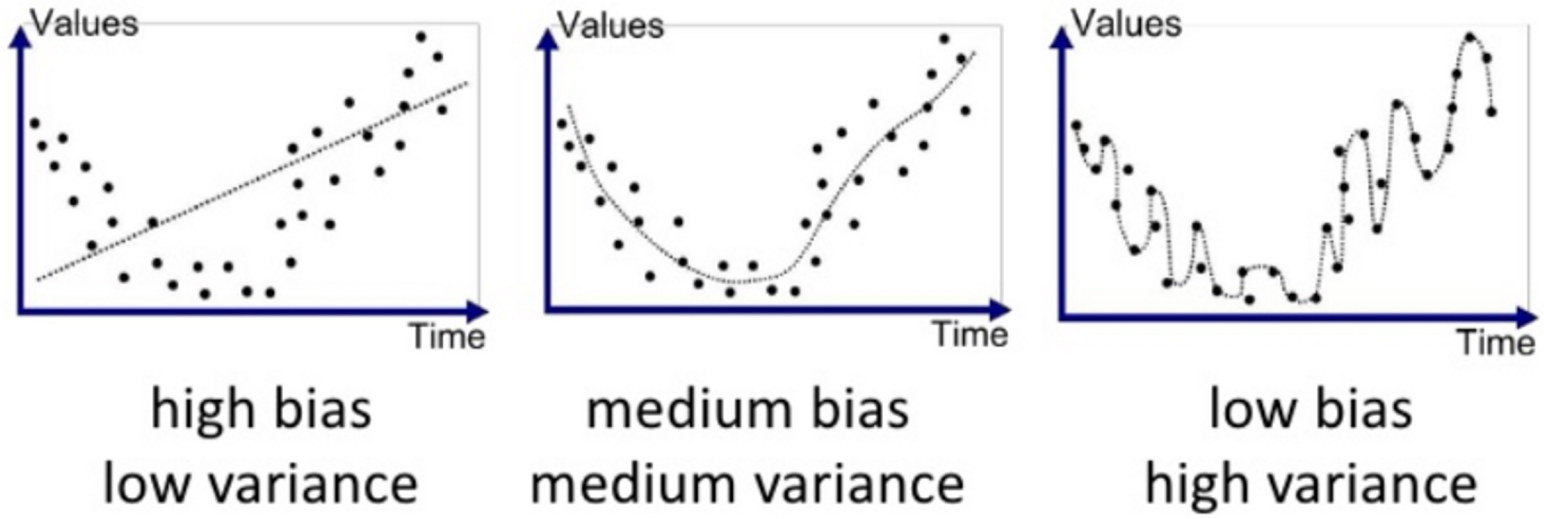
Bias + Variance = Epistemic Uncertainty

Uncertainty in the parameters of a model

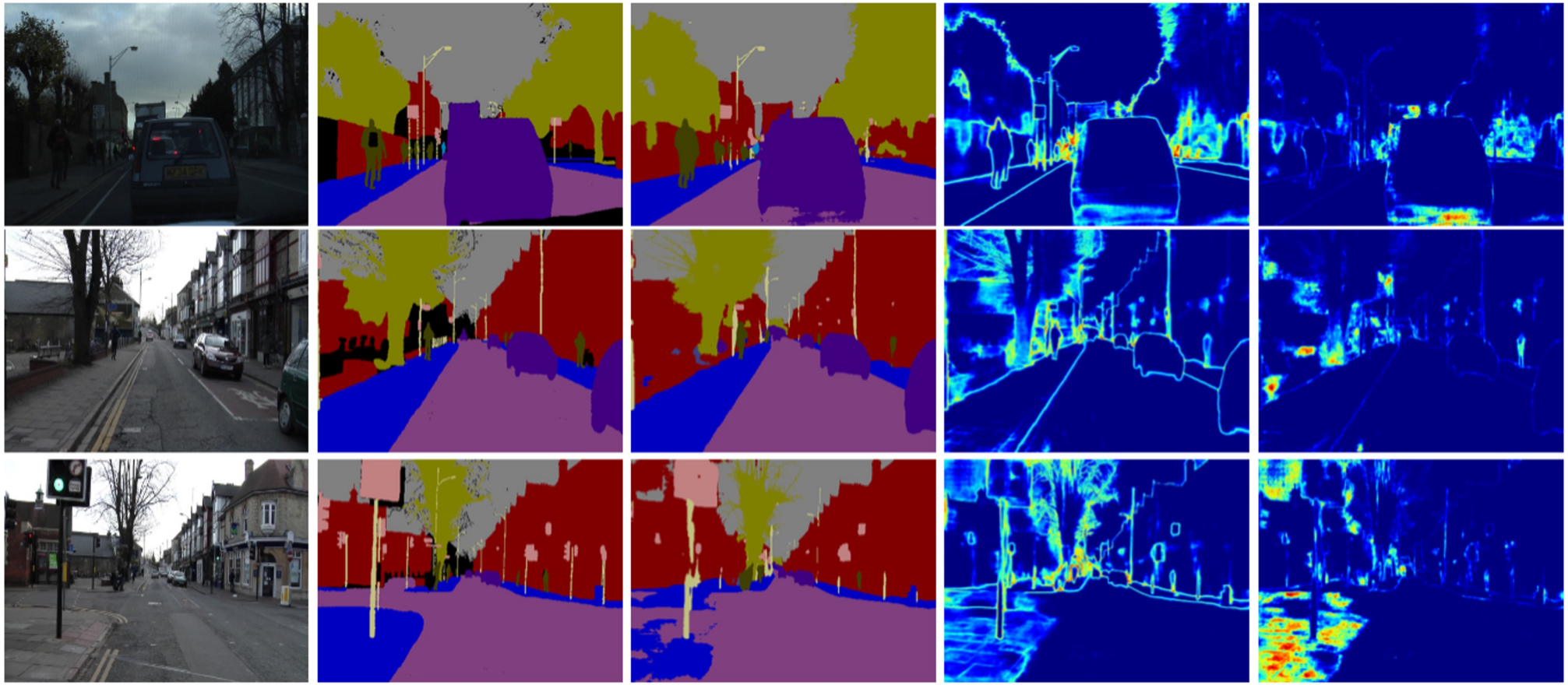
Captures our ignorance about which model generated our collected data

Expected to be high when the system encounters examples that differ from the training data (out-of-distribution)

Can (eventually) be reduced to zero given more data

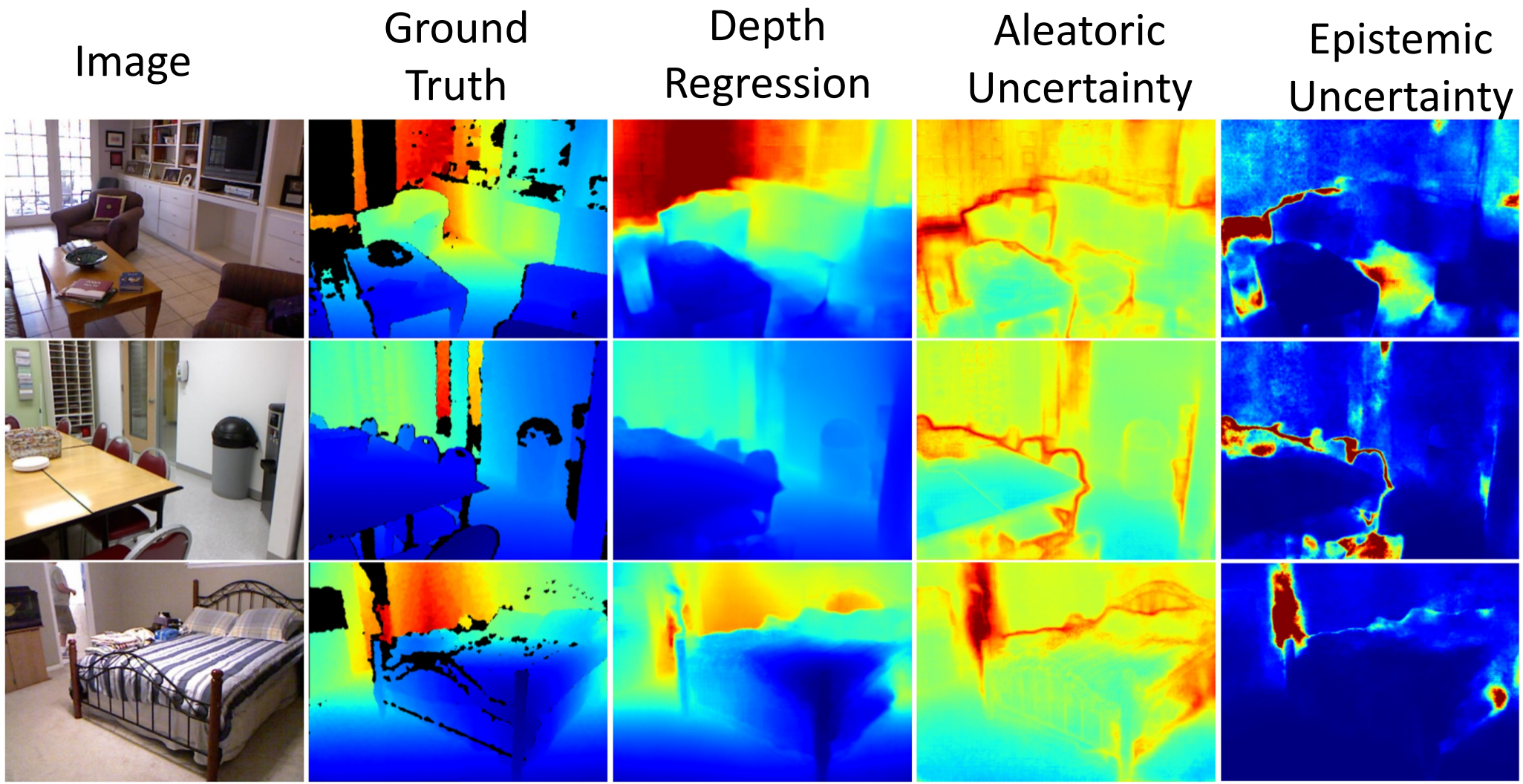


Epistemic and Aleatoric Uncertainty

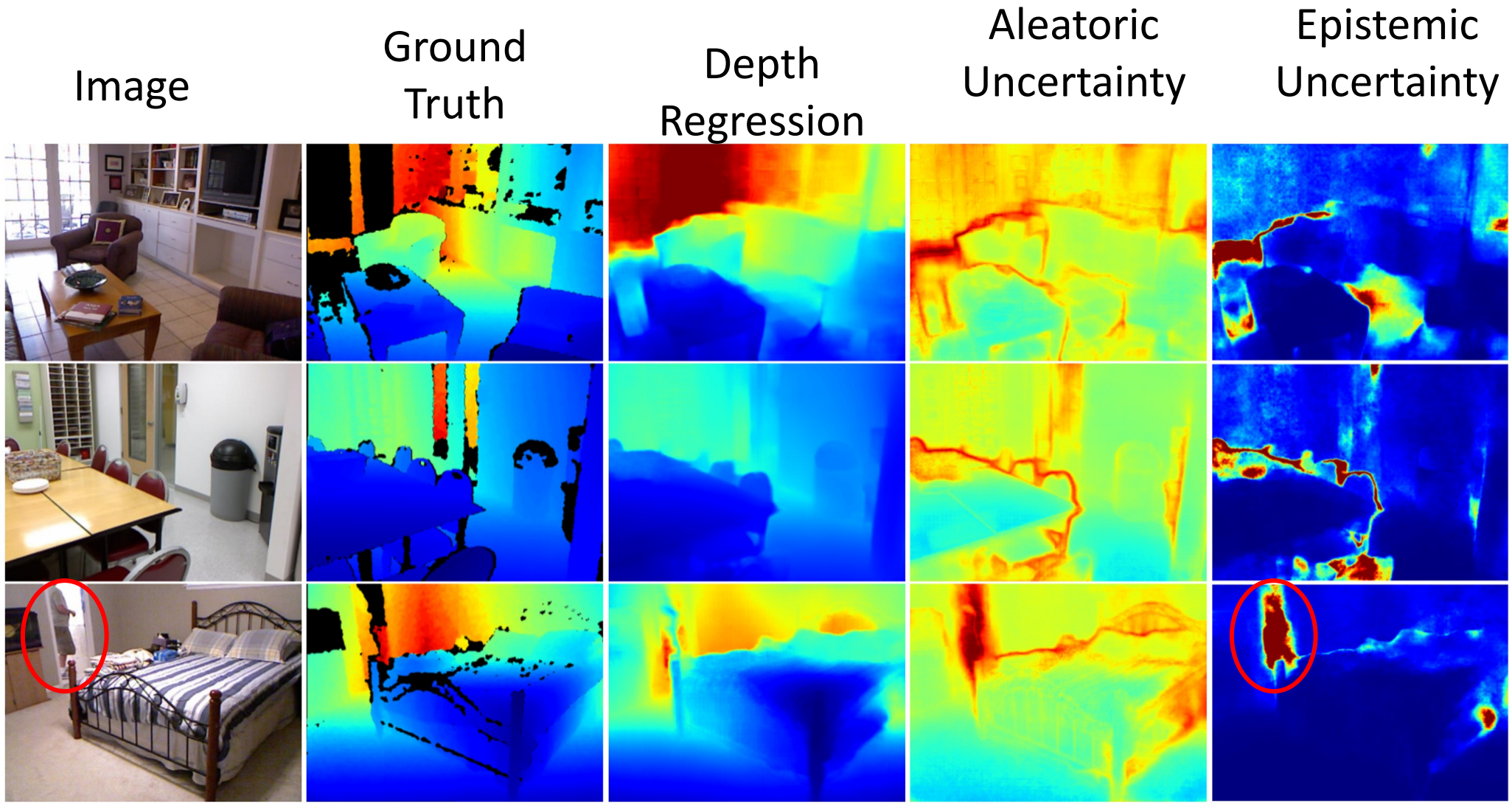


(a) Input Image (b) Ground Truth (c) Semantic Segmentation (d) Aleatoric Uncertainty (e) Epistemic Uncertainty

Epistemic and Aleatoric Uncertainty



Epistemic and Aleatoric Uncertainty



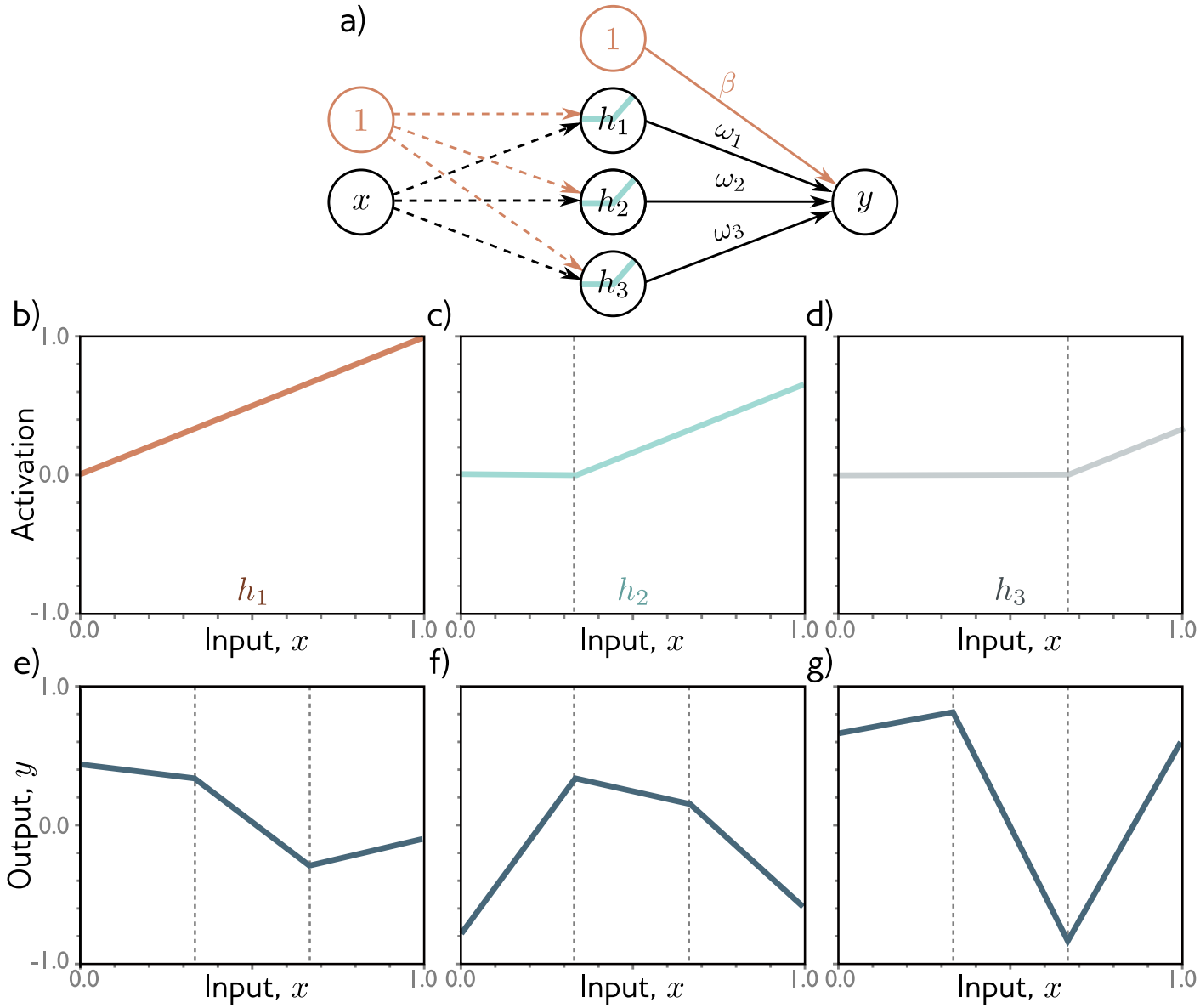
Few people in training data

Noise, bias, and variance

- Variance is the uncertainty in fitted model due to choice of training set
- Bias is systematic deviation from the mean of the function we are modeling due to limitations in our model
- Noise is inherent uncertainty in the true mapping from input to output

Toy model

- K hidden units
- First layer fixed so “joints” divide interval evenly
- Second layer trained
- But… now linear in **h**
  - so convex cost function
  - can find best solution in closed-form



Least squares regression only

$$L[x] = \left(\mathrm{f}[x, \boldsymbol{\phi}] - y[x]\right)^2$$

- We can show that:

$$\mathbb{E}_{\mathcal{D}}\Big[\mathbb{E}_{y}[L[x]]\Big] = \underbrace{\mathbb{E}_{\mathcal{D}}\Big[\left(\mathrm{f}[x, \boldsymbol{\phi}[\mathcal{D}]] - f_{\mu}[x]\right)^2\Big]}_{\text{variance}} + \underbrace{\left(f_{\mu}[x] - \mu[x]\right)^2}_{\text{bias}} + \underbrace{\sigma^2}_{\text{noise}}$$

Expectation over noise in training data

Expectation over noise in test data

Actual model

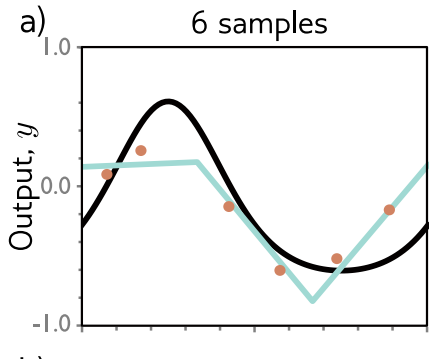
Best possible model if we had infinite data

True function

Measuring performance

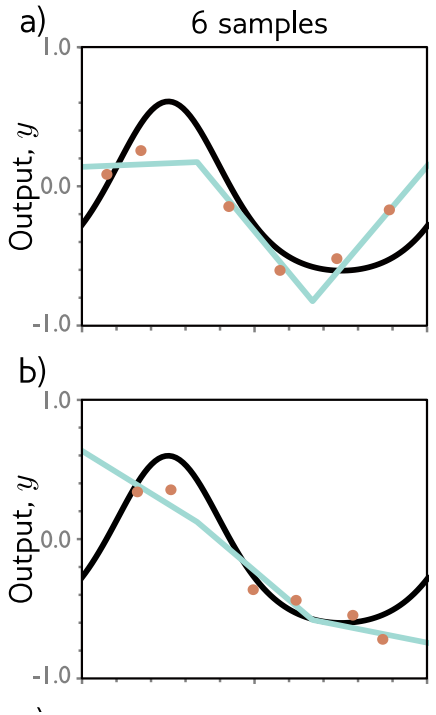
- MNIST1D dataset model and performance
- Noise, bias, and variance
- Reducing variance
- Reducing bias & bias-variance trade-off
- Double descent
- Curse of dimensionality & weird properties of high dimensional space
- Choosing hyperparameters

Variance



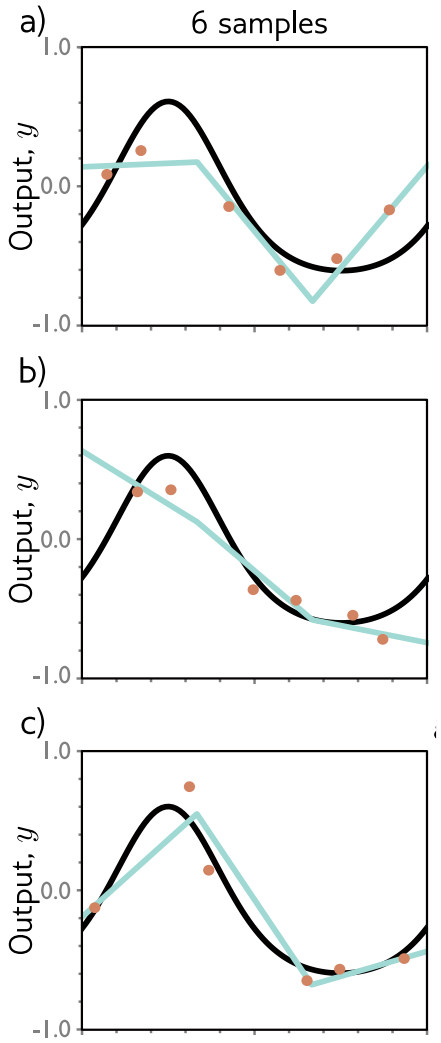
Draw sample 1

Variance



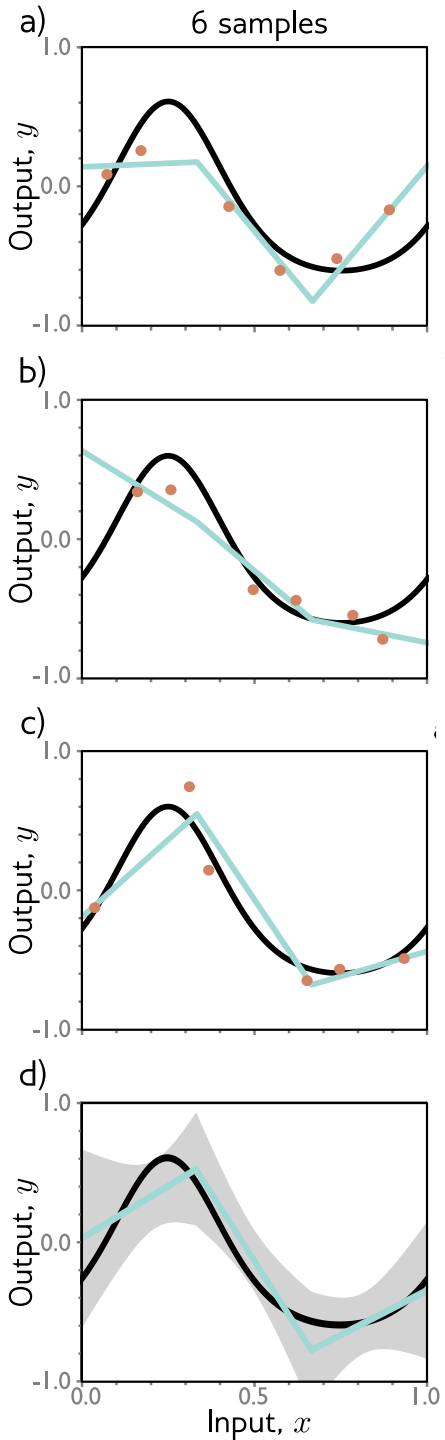
Draw sample 2

Variance



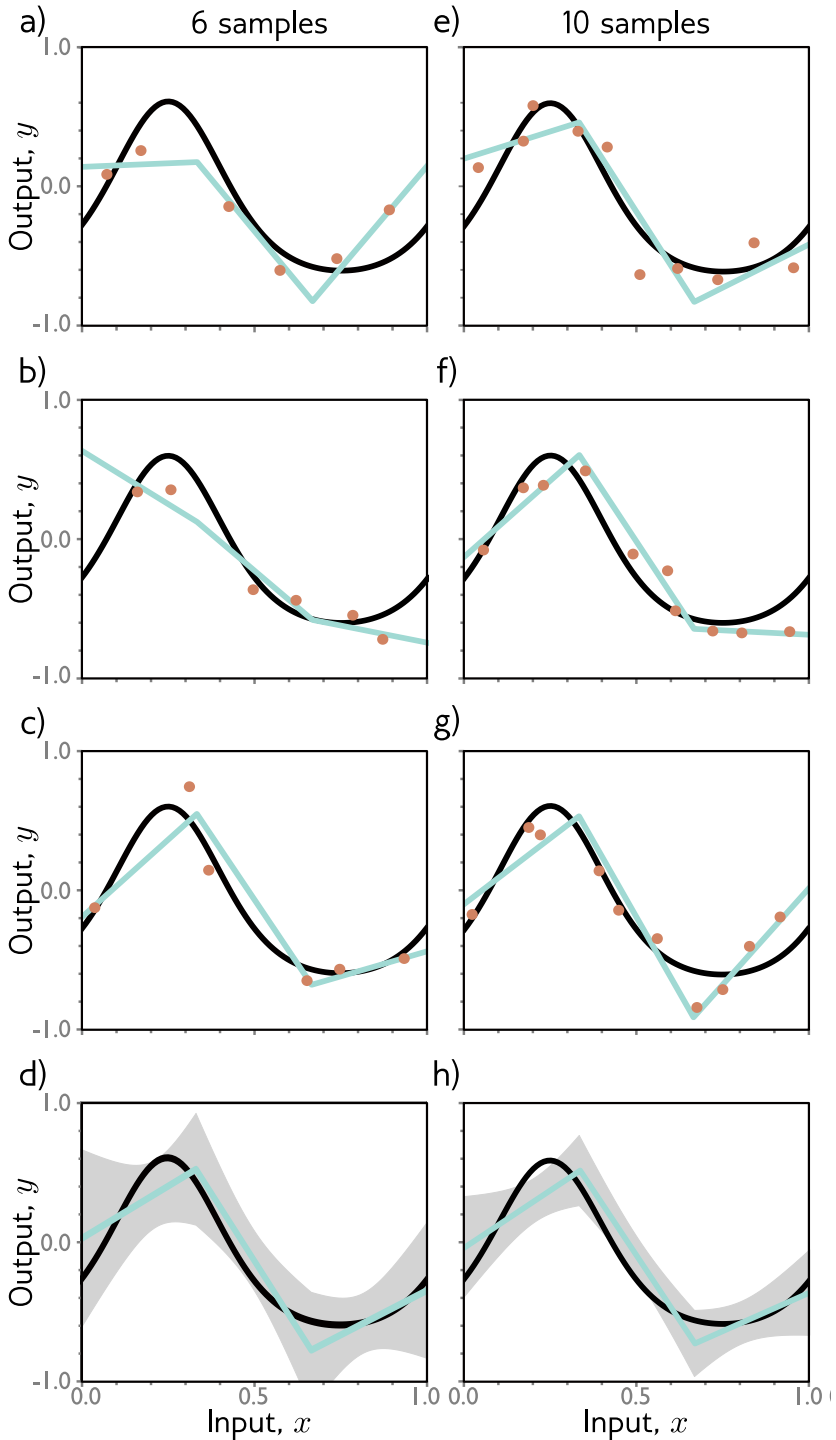
Draw sample 2

Variance

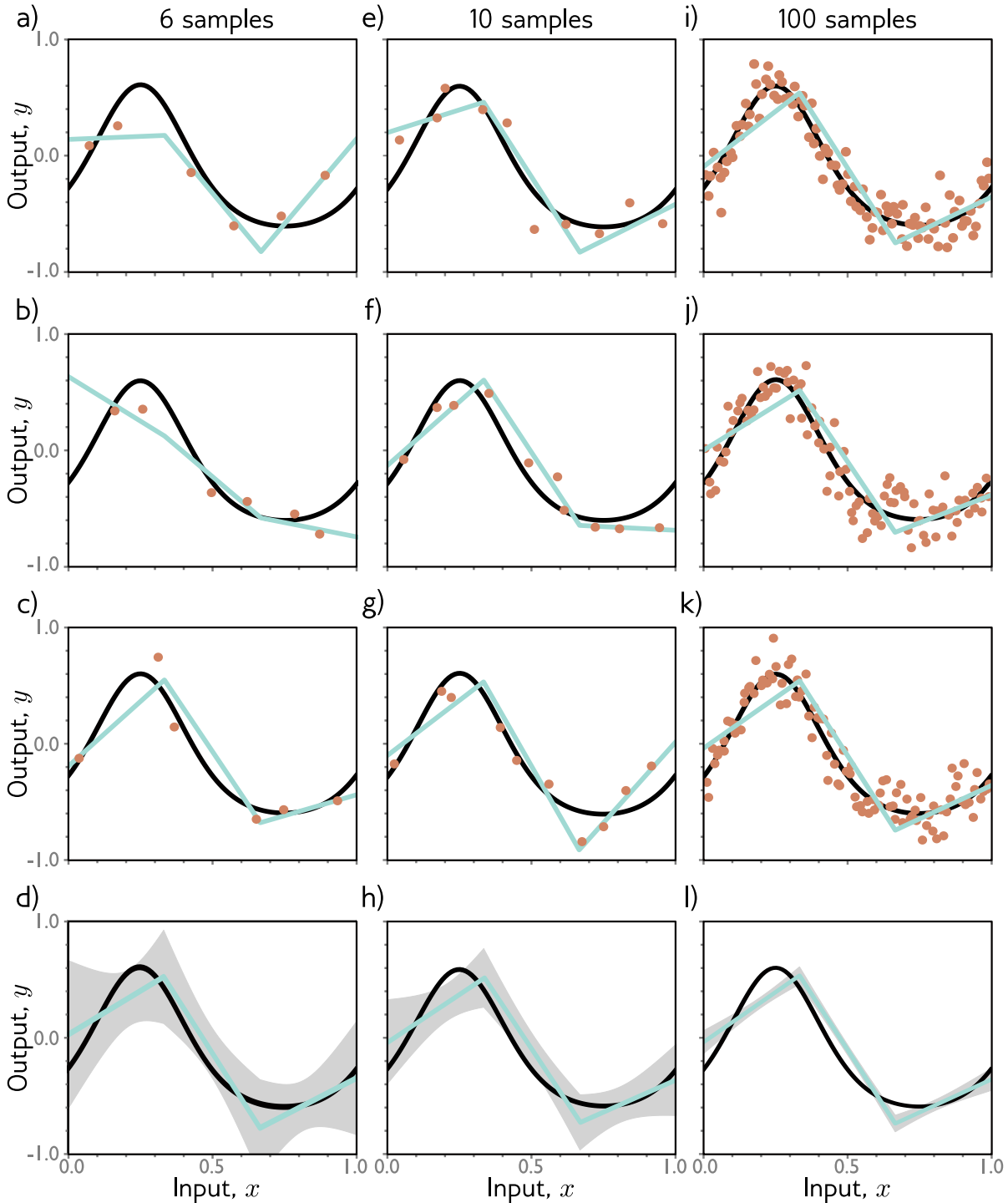


Variance over 3 draws

Variance


a)
6 samples
e)
10 samples
b)
f)
c)
g)
d)
h)
Output, y
1.0
0.0
-1.0
0.0
0.5
1.0
Input, x

Variance

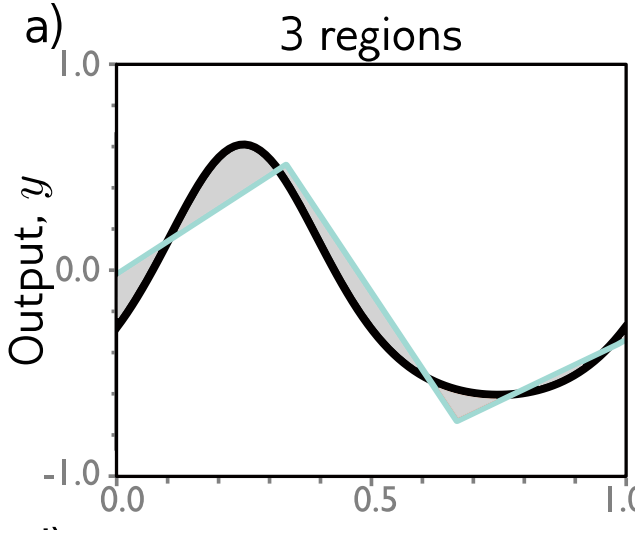


Can reduce variance by adding more samples

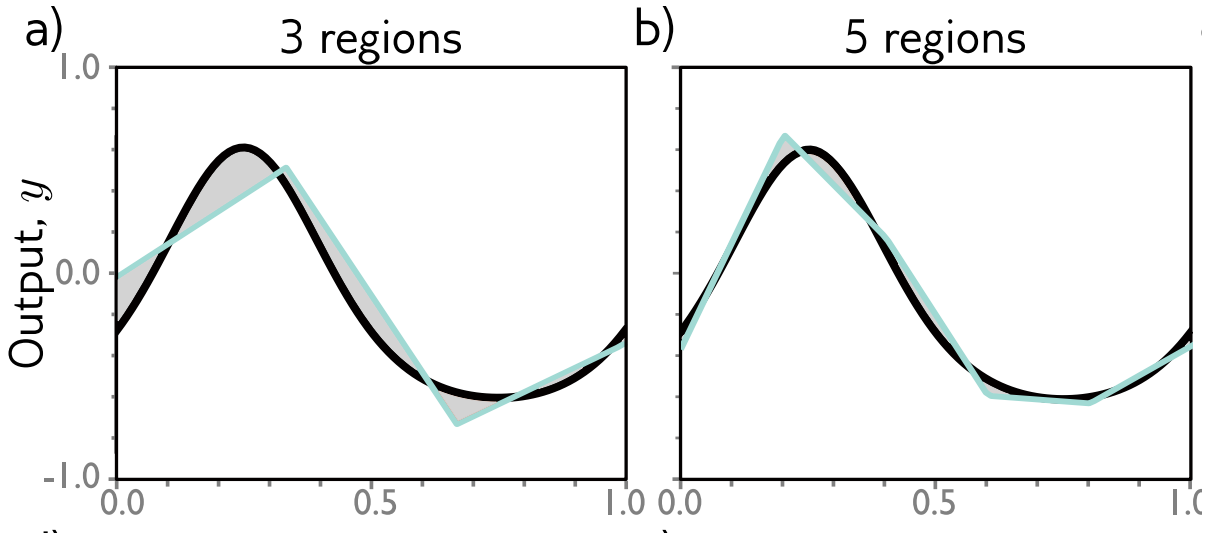
Measuring performance

- MNIST1D dataset model and performance
- Noise, bias, and variance
- Reducing variance
- Reducing bias & bias-variance trade-off
- Double descent
- Curse of dimensionality & weird properties of high dimensional space
- Choosing hyperparameters

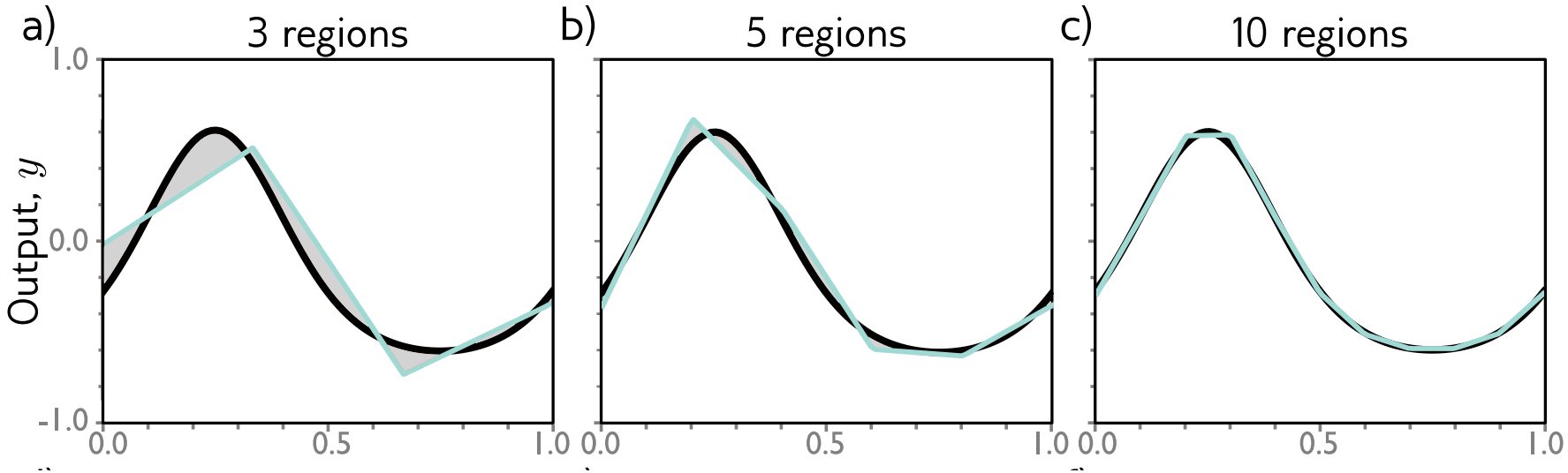
Reducing bias



Reducing bias

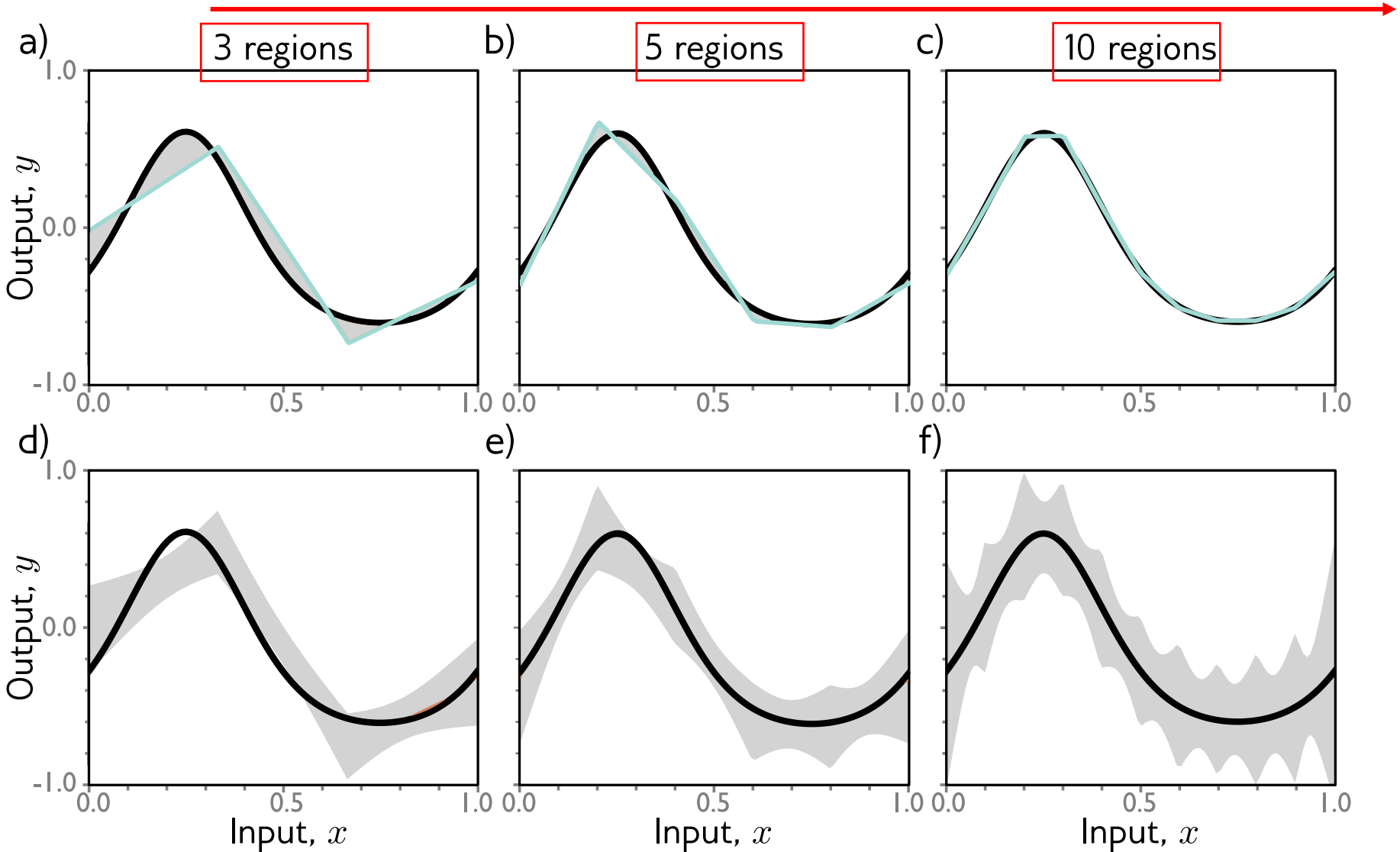


Reducing bias



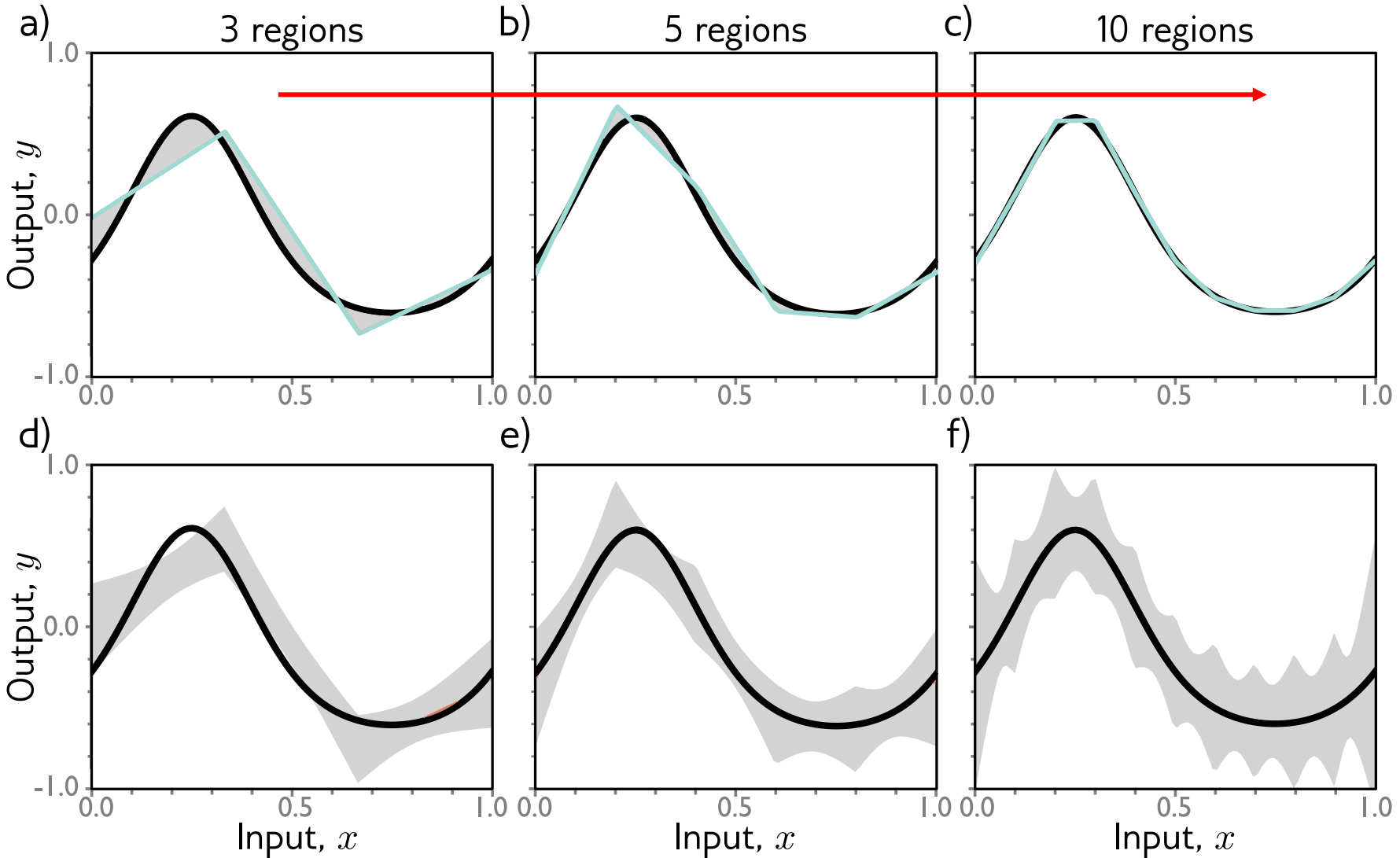
Reducing bias

Increasing Capacity



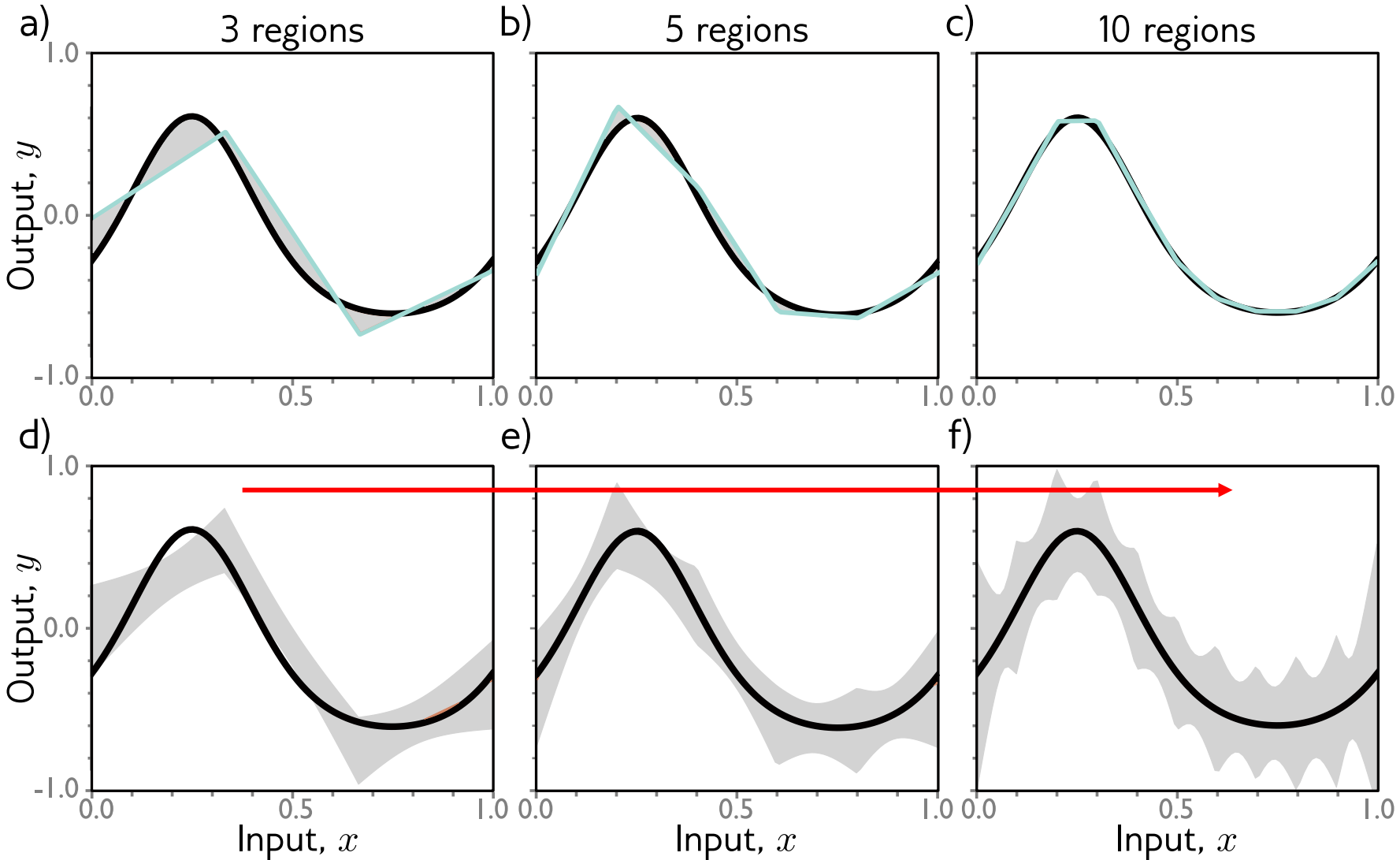
Reducing bias

Decreasing bias

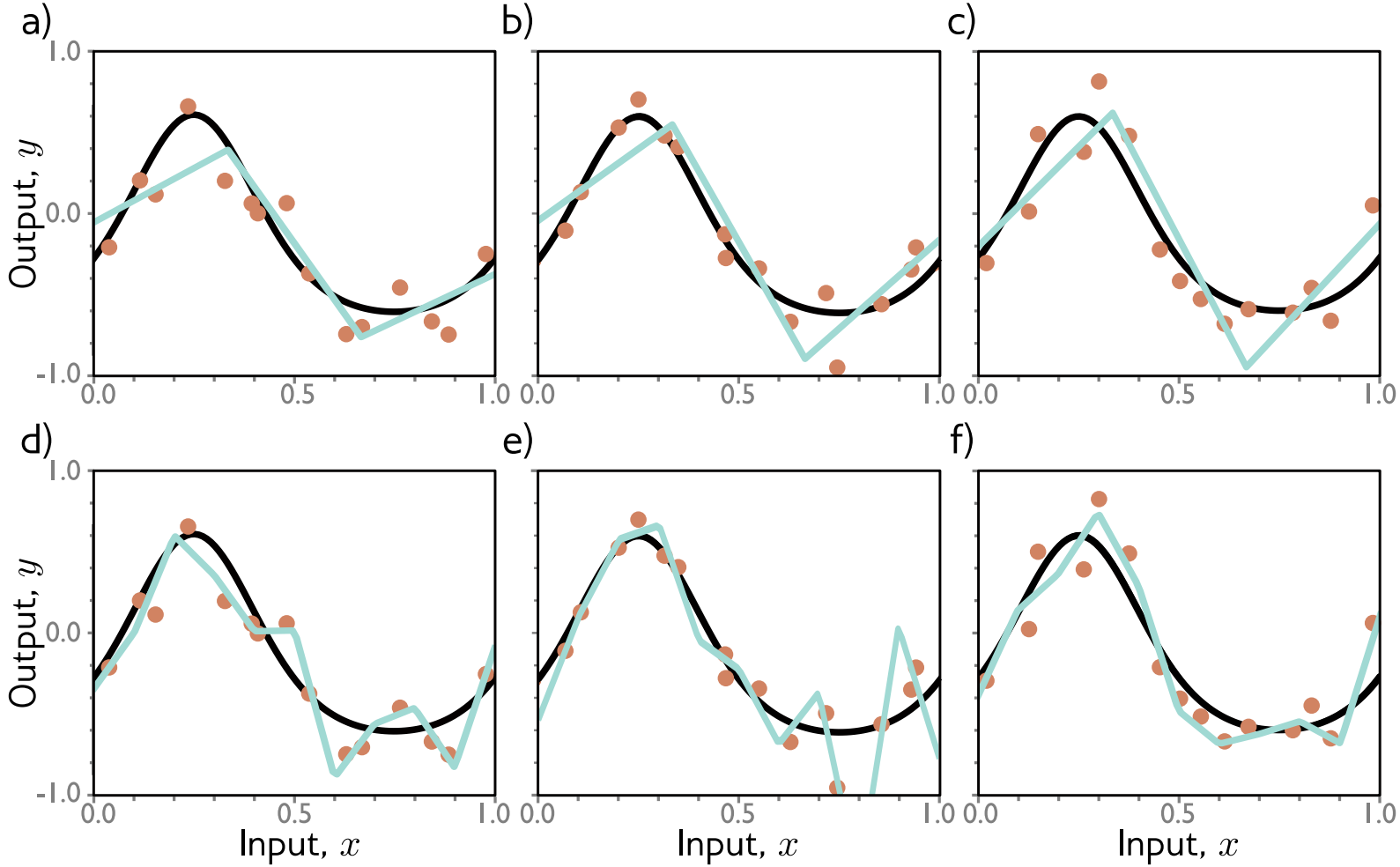


Reducing bias

Increasing Variance

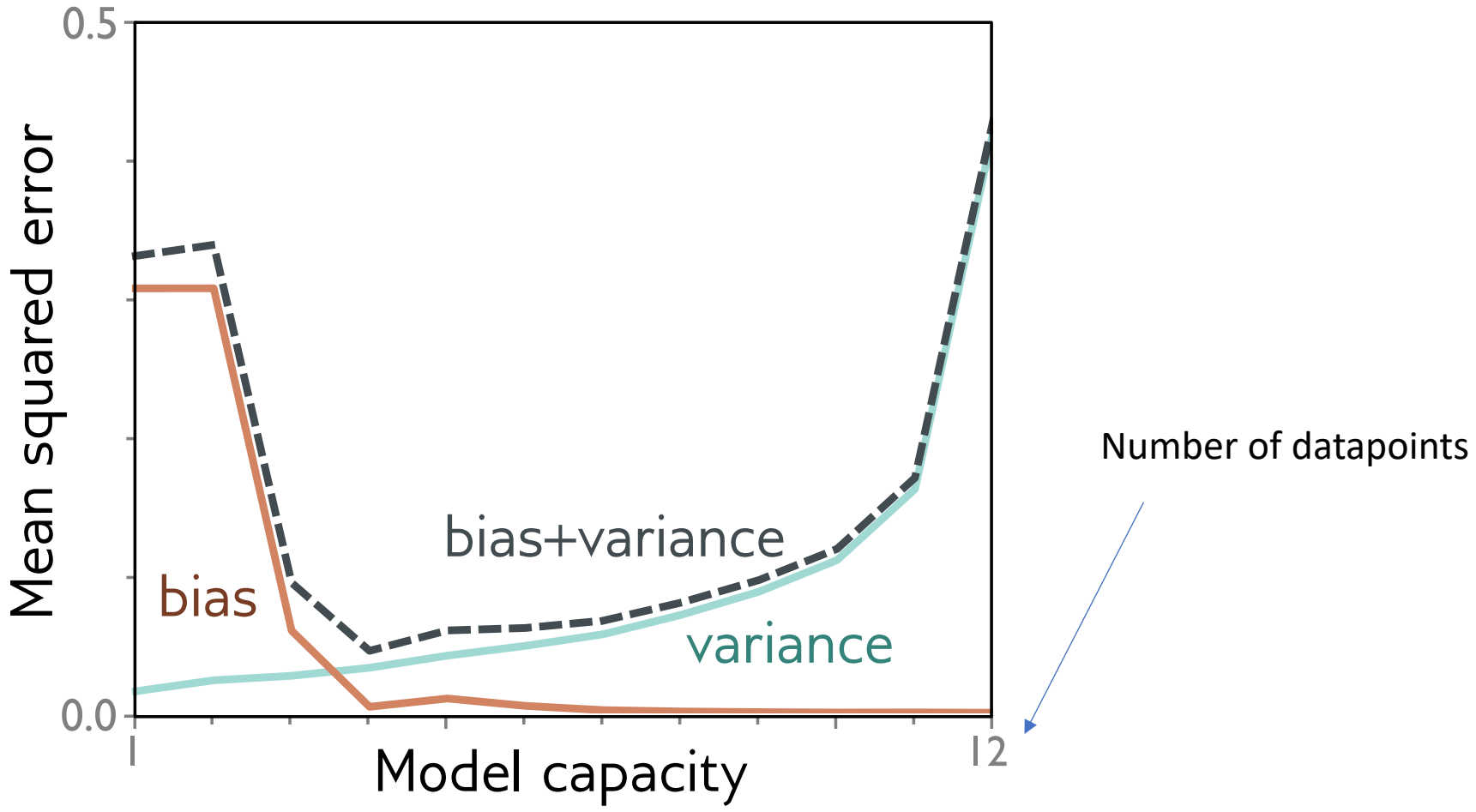


Why does variance increase? Overfitting



Describes the training data better, but not the true underlying function (black curve)

Bias and variance trade-off



Measuring performance

- MNIST1D dataset model and performance
- Noise, bias, and variance
- Reducing variance
- Reducing bias & bias-variance trade-off
- Double descent
- Curse of dimensionality & weird properties of high dimensional space
- Choosing hyperparameters

Questions?

Double Descent

"The discovery of double descent is recent, unexpected, and somewhat puzzling" – Prince, p.129

Double Descent

Interaction of two phenomena:

1) Test performance is worse and capacity equals data set size and can memorize exactly (interpolation threshold)

Double Descent

Interaction of two phenomena:

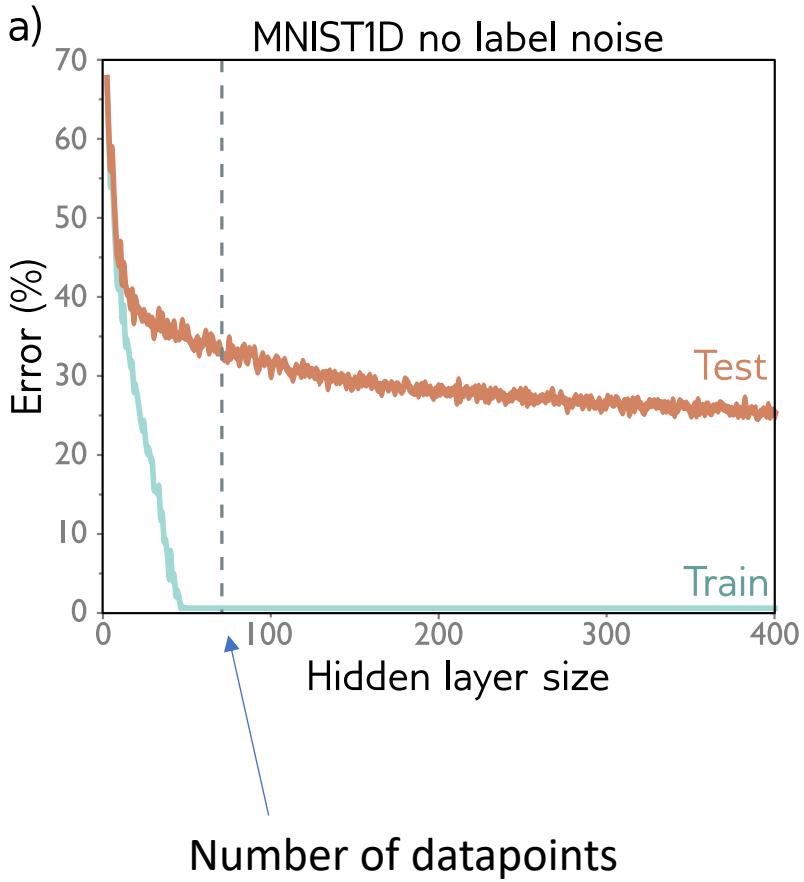
1) Test performance is worse and capacity equals data set size and can memorize exactly (interpolation threshold)
2) Test performance **improves** as we increase capacity beyond this point

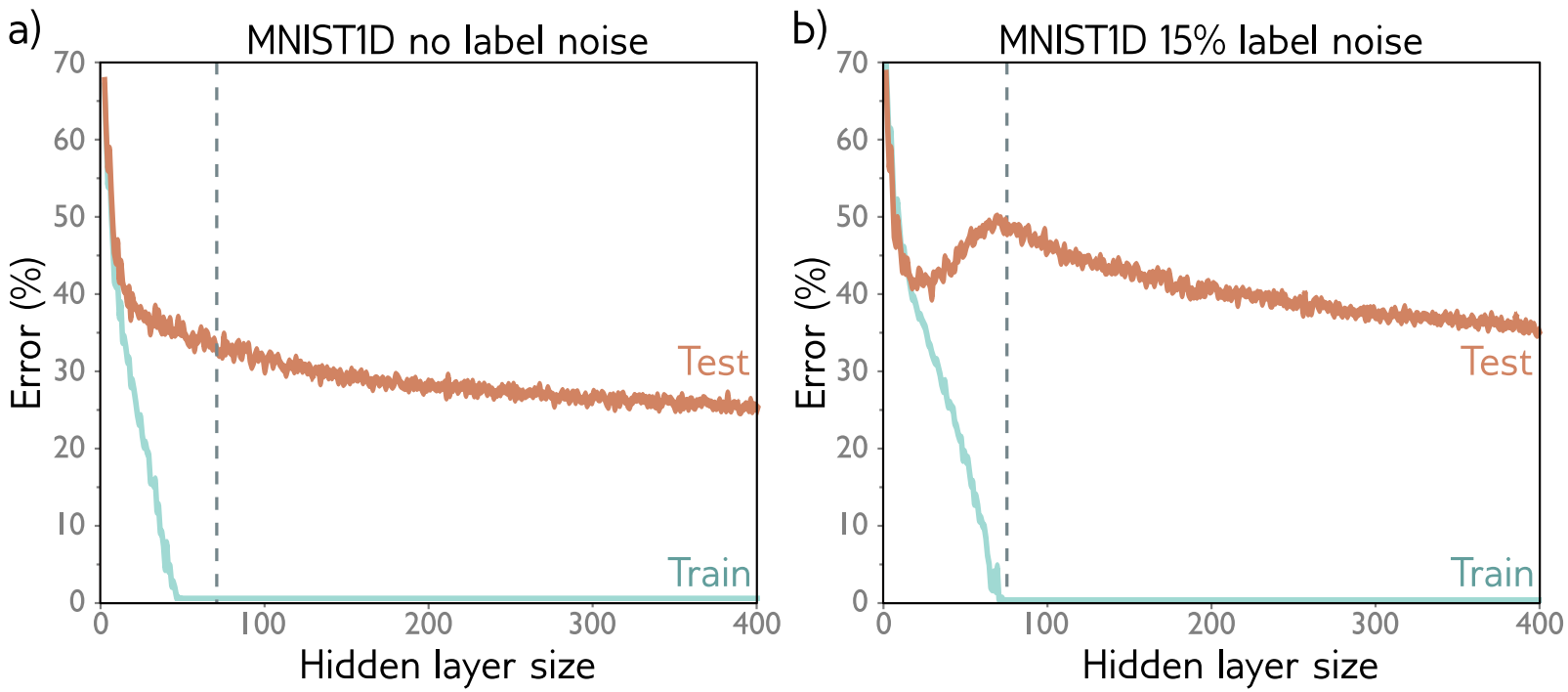
Double Descent

Interaction of two phenomena:

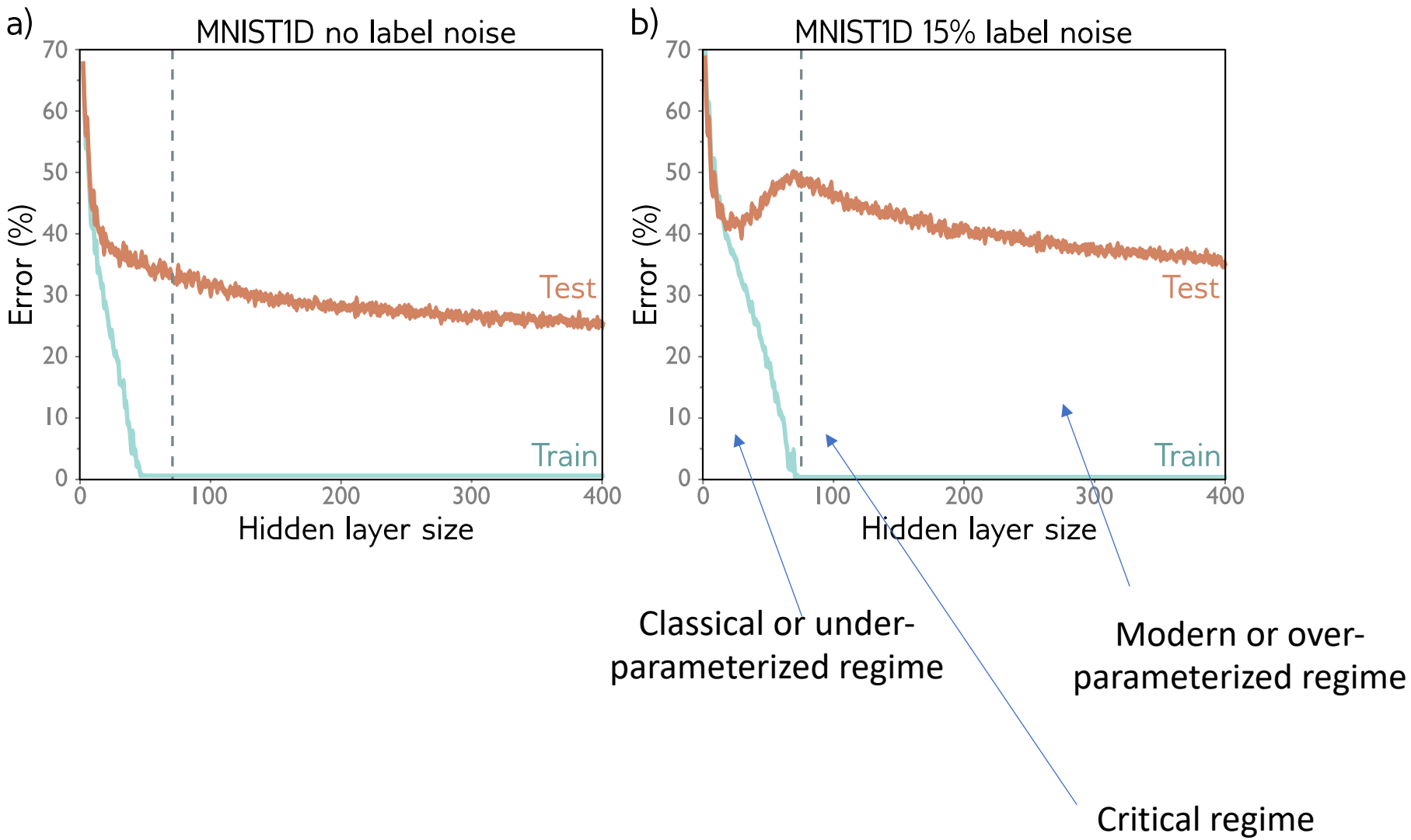
1) Test performance is worse and capacity equals data set size and can memorize exactly (interpolation threshold)
2) Test performance **improves** as we increase capacity beyond this point

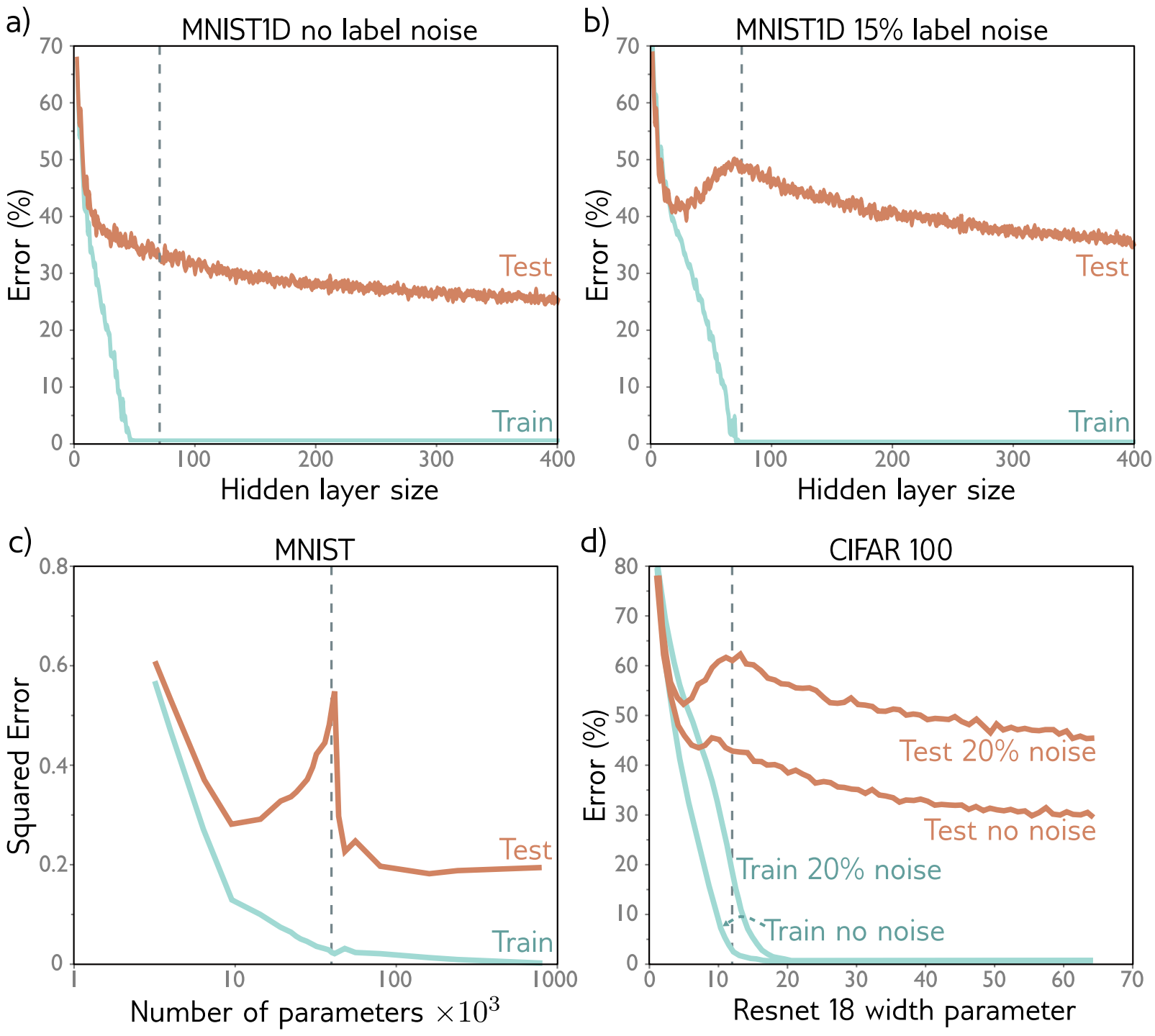
Contradicts theory of bias-variance trade-off

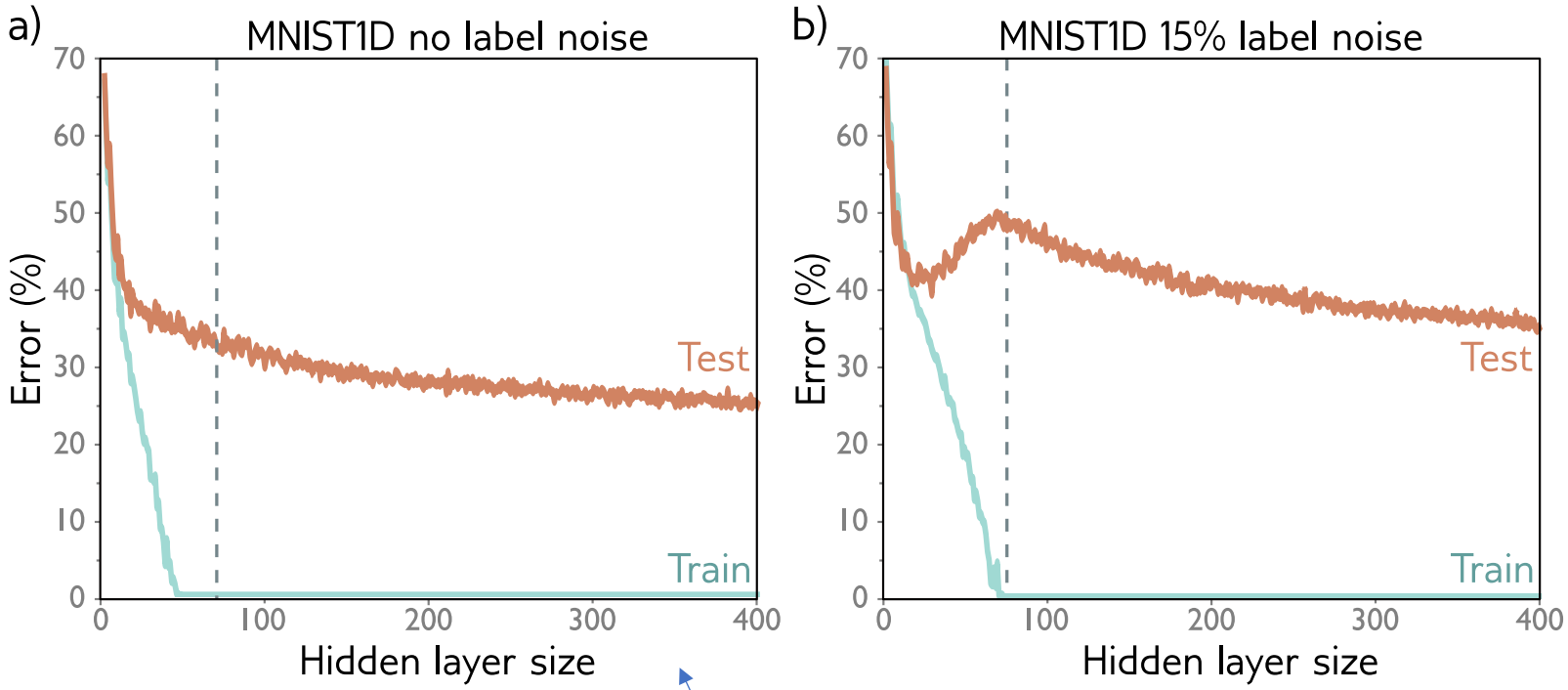

a)
MNIST1D no label noise
Error (%)
70
60
50
40
30
20
10
0
0
100
200
300
400
Test
Train
Hidden layer size
Number of datapoints



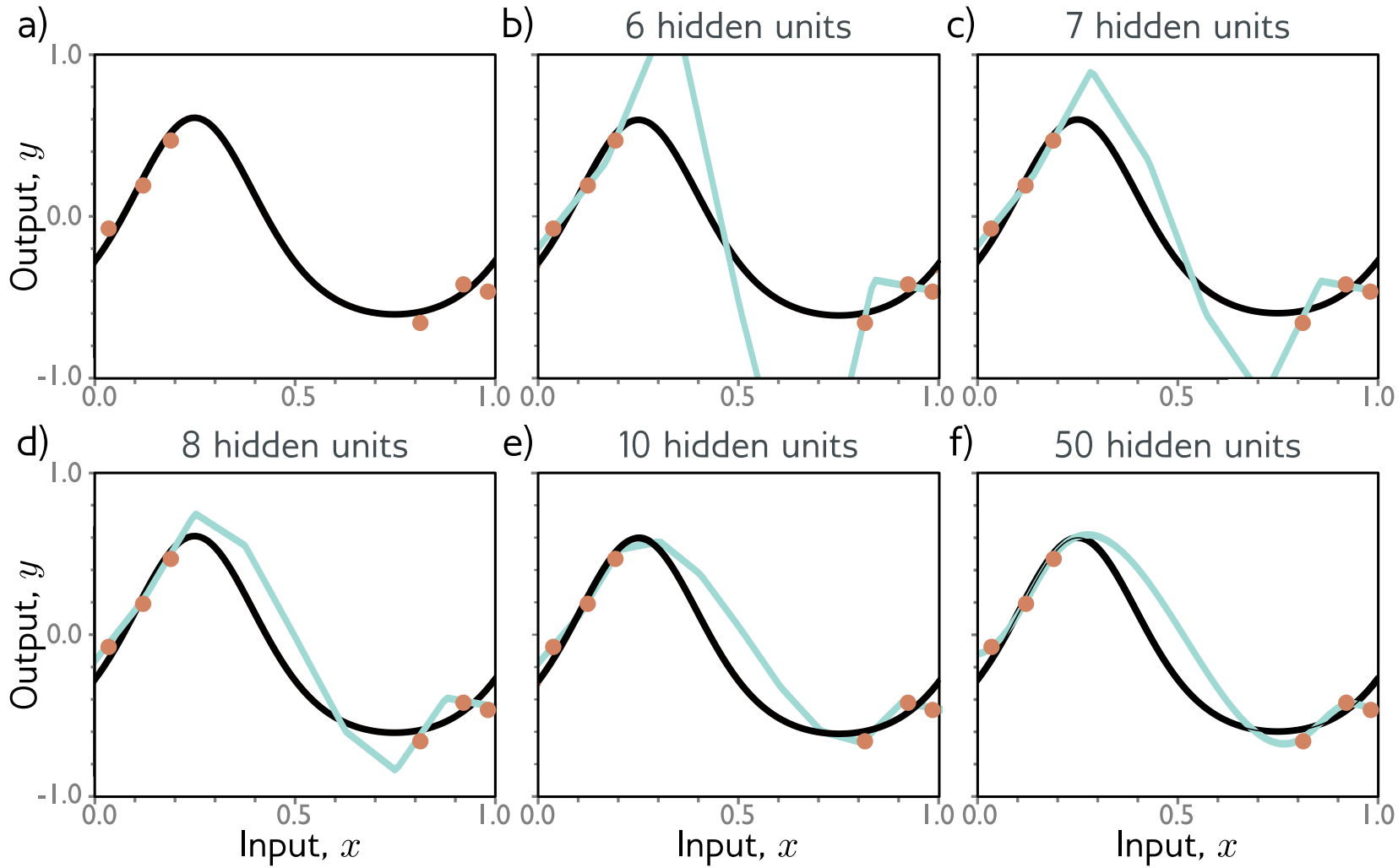
Double descent


a)
MNIST1D no label noise
Error (%)
Hidden layer size
Test
Train
b)
MNIST1D 15% label noise
Error (%)
Hidden layer size
Test
Train
Classical or under-parameterized regime
Modern or over-parameterized regime
Critical regime


a)
MNIST1D no label noise
Error (%)
Test
Train
Hidden layer size
b)
MNIST1D 15% label noise
Error (%)
Test
Train
Hidden layer size
c)
MNIST
Squared Error
Test
Train
Number of parameters $\times 10^3$
d)
CIFAR 100
Error (%)
Test 20% noise
Test no noise
Train 20% noise
Train no noise
Resnet 18 width parameter



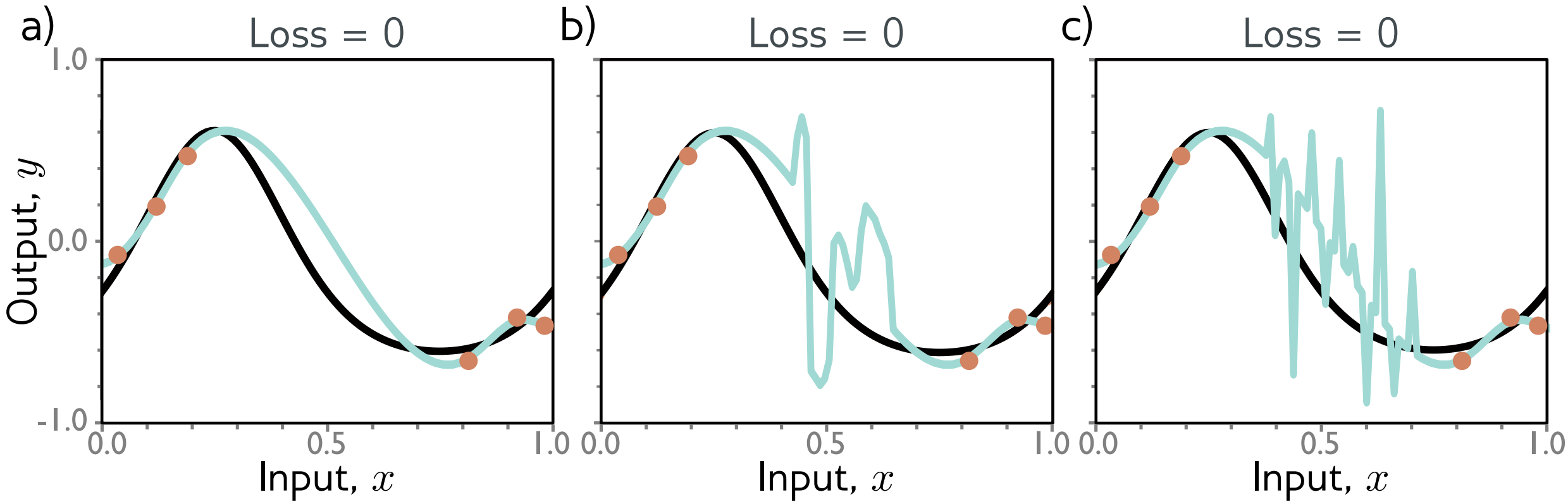
- Note that train data is very close to zero.
- Whatever is happening isn't happening at training data points
- Must be happening between the data points??



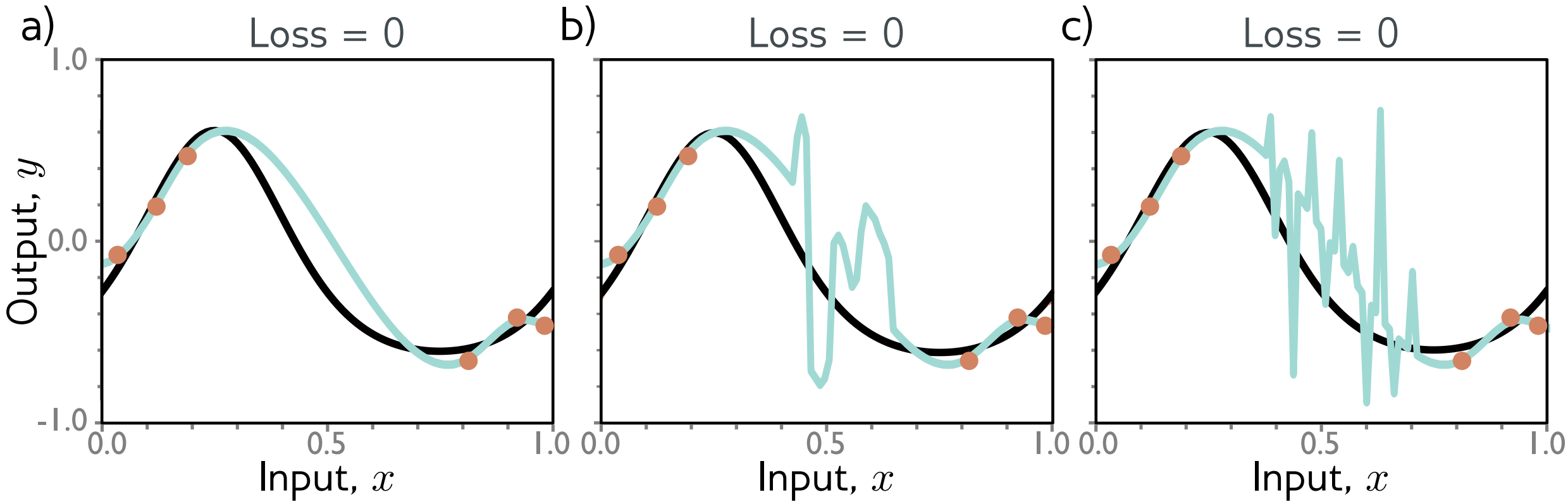
Potential explanation:

- can make smoother functions with more hidden units
- being smooth between the datapoints is a reasonable thing to do

But why?



- All of these solutions are equivalent in terms of loss.
- Why should the model choose the smooth solution?
- Tendency of model to choose one solution over another is inductive bias



Possible Explanations:

- Network initialization (He init) encourages smoothness (prior over weights)
  - sensible magnitudes that create smooth functions
- Training procedure is regularized enough to encourage smoothness
  - SGD, weight decay, batch- and layer-norm, dropout, etc…

Measuring performance

- MNIST1D dataset model and performance
- Noise, bias, and variance
- Reducing variance
- Reducing bias & bias-variance trade-off
- Double descent
- Curse of dimensionality & weird properties of high dimensional space
- Choosing hyperparameters

Curse of dimensionality

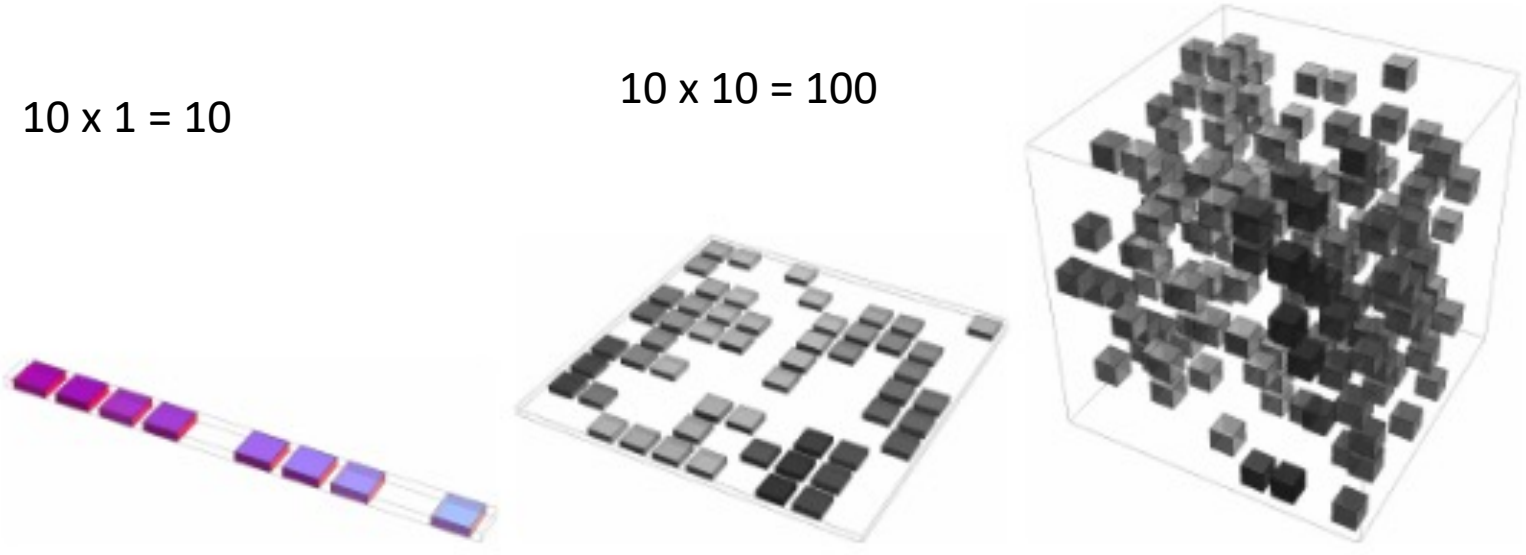
- 40-dimensional data
- 10,000 data points
- Consider quantizing each dimension into 10 bins
- 10^{40} bins
- 1 data point per 10^{35} bins
- The tendency of high-dimensional space to overwhelm the number of data points is called the curse of dimensionality

Curse of dimensionality

10 x 1 = 10

10 x 10 = 100

10 x 10 x 10 = 1000



As the number of dimensions grows, the number of possible “bins” grows exponentially. In order to maintain a fixed amount of coverage of bins (here 80%), the amount of data needed also grows exponentially

Measuring performance

- MNIST1D dataset model and performance
- Noise, bias, and variance
- Reducing variance
- Reducing bias & bias-variance trade-off
- Double descent
- Curse of dimensionality & weird properties of high dimensional space
- Choosing hyperparameters

Choosing hyperparameters

- Don’t know bias or variance
- Don’t know how much capacity to add
- How do we choose capacity in practice?
  - Or model structure
  - Or training algorithm
  - Or learning rate
- Third data set – validation set
  - Train models with different hyperparameters on training set
  - Choose best hyperparameters with validation set
  - Test once with test set

Assessing Generalization: Beyond the IID Assumption

IID Recap

In most of supervised learning problems, we make the assumption that our data are independent and identically distributed (IID)

This means that all samples are independent of each other, and follow the same distribution

This applies to both the **training data** and the **test data**

Gives us confidence that the outputs of our network will generalize

Independent

$$Y_i \mid X_i \perp\!\!\!\perp Y_j \mid x_j$$

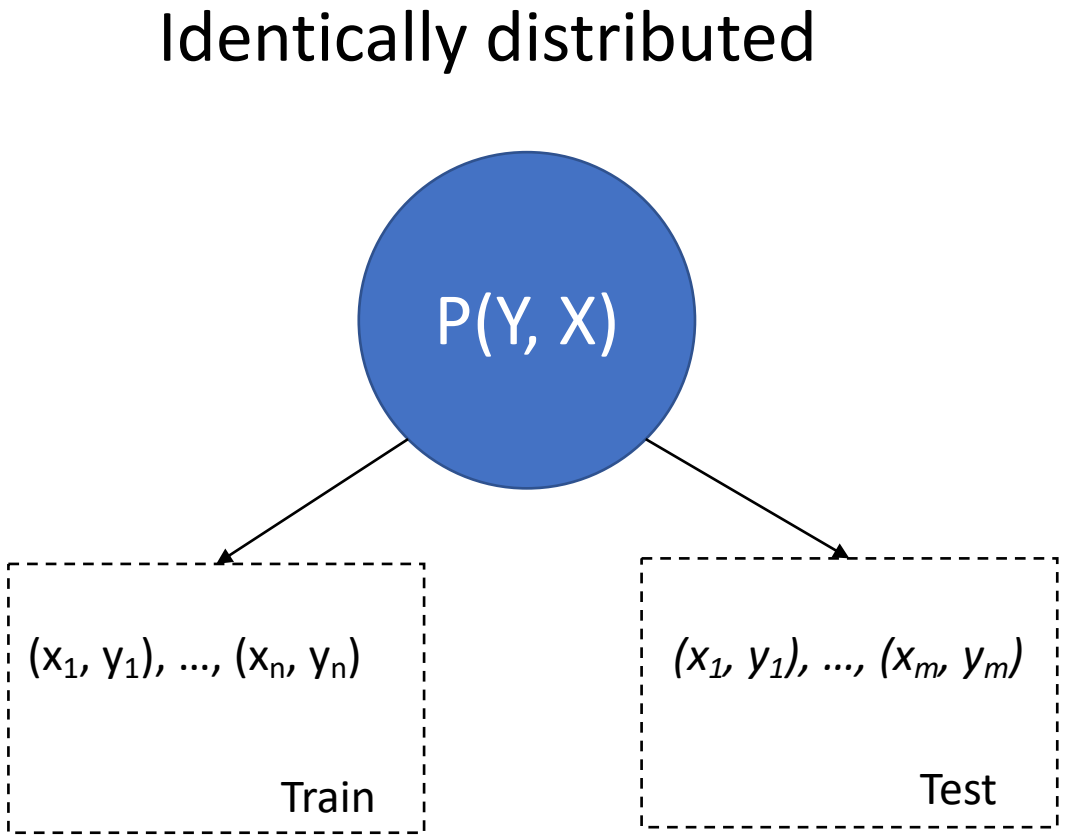
IID Recap

In most of supervised learning problems, we make the assumption that our data are independent and identically distributed (IID)

This means that all samples are independent of each other, and follow the same distribution

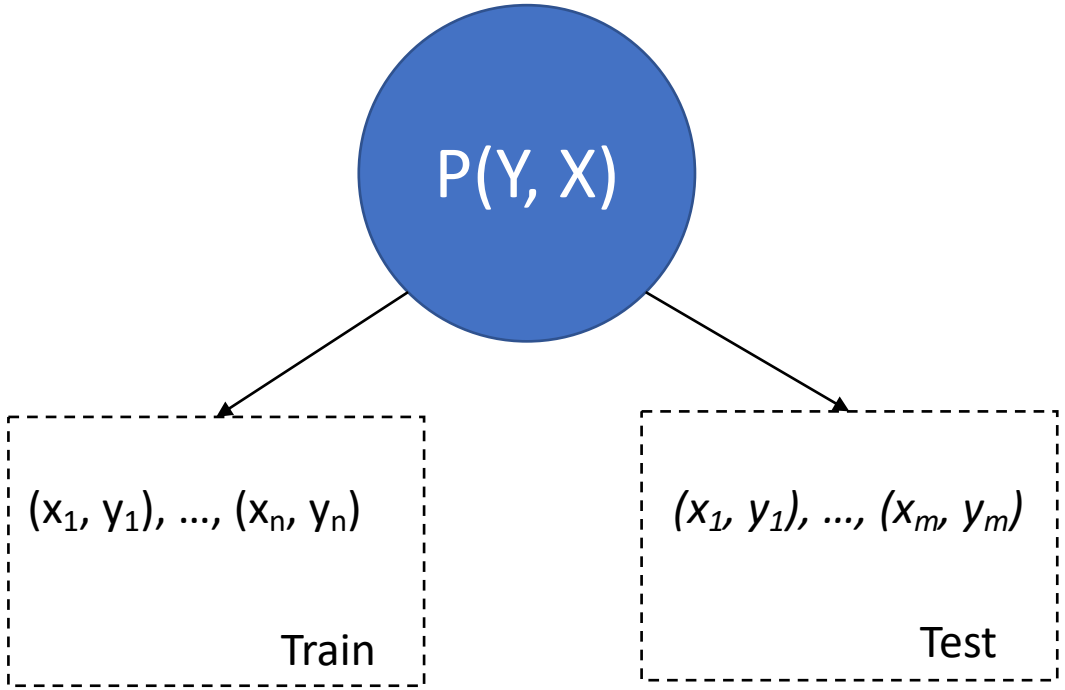
This applies to both the **training data** and the **test data**

Gives us confidence that the outputs of our network will generalize



IID Recap

Identically distributed

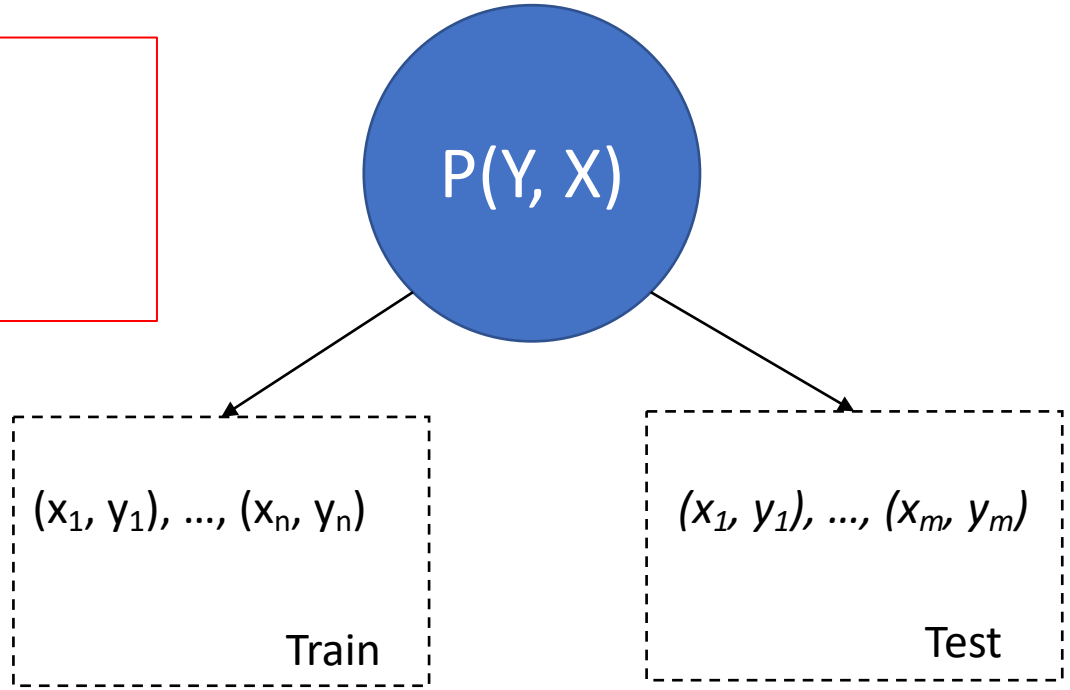


Allows us to learn model parameters via maximum likelihood

IID Recap

Identically distributed

$$\hat{\phi} = \underset{\phi}{\operatorname{argmax}} \left[\prod_{i=1}^{I} Pr(\mathbf{y}_i | \mathbf{f}[\mathbf{x}_i, \phi]) \right]$$



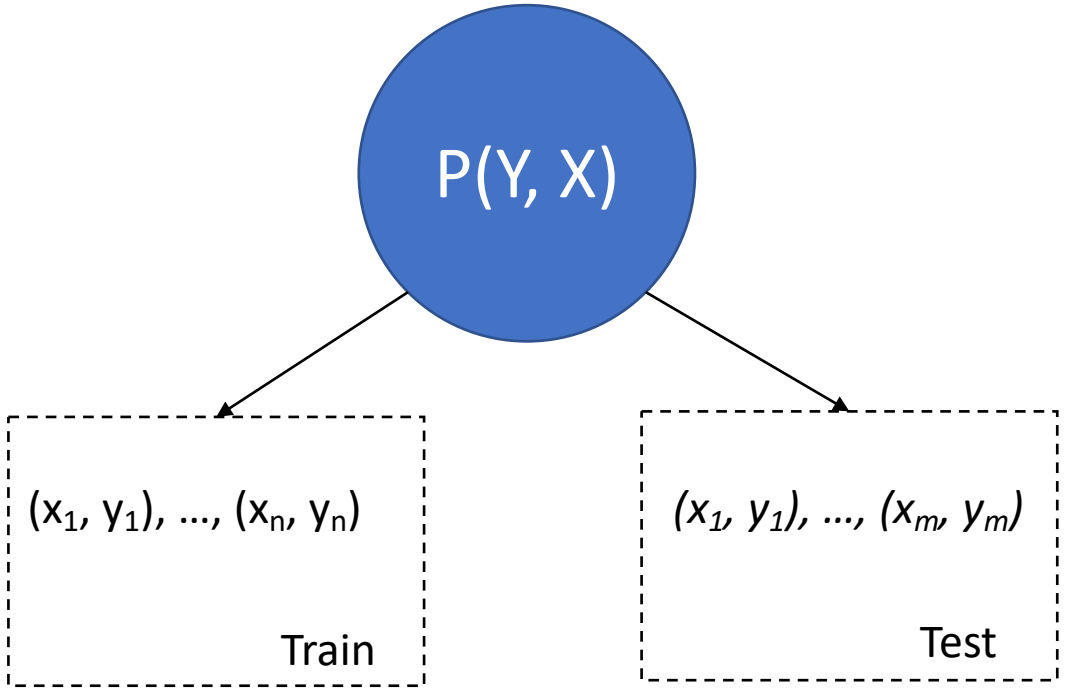
Product Rule of Probability:

If A and B are independent:
then P(A and B) = P(A)*P(B)

Allows us to learn model parameters via maximum likelihood

IID Recap

Identically distributed



Allows us to generalize to new data

Distribution Shift

Unfortunately, modern machine learning is not so clean…

We often observe shifts of the distribution between when when we train, and when the model is actually performing inference

Neural networks are actually quite bad at adapting to distributional changes

Distribution Shift

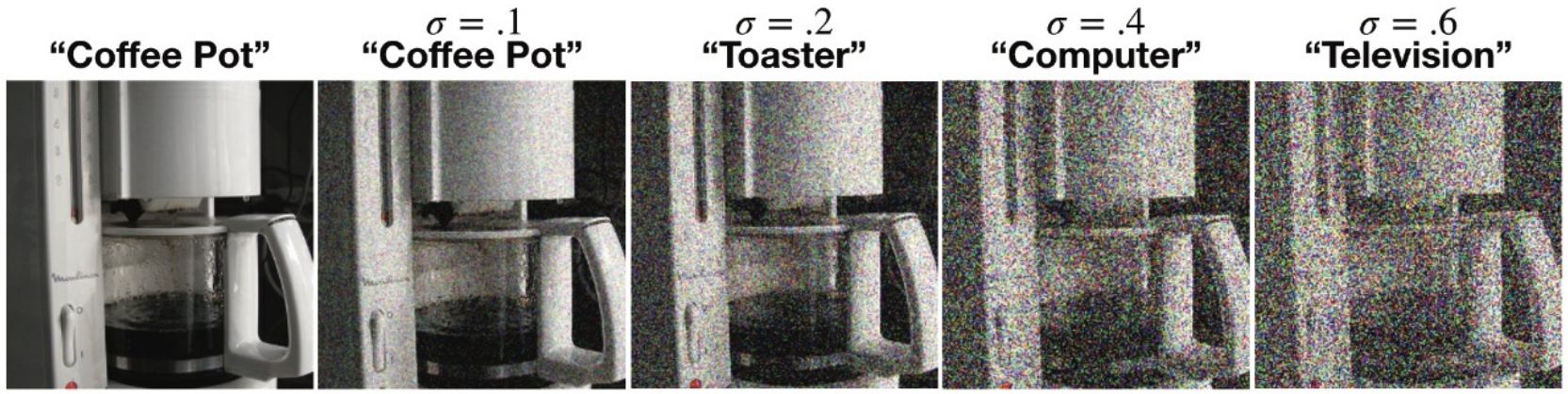


Figure 19.1: Effect of Gaussian noise of increasing magnitude on an image classifier. The model is a ResNet-50 CNN trained on ImageNet. From Figure 23 of [For+19]. Used with kind permission of Justin Gilmer.

Distribution Shift

(A) **Cow: 0.99**, Pasture: 0.99, Grass: 0.99, No Person: 0.98, Mammal: 0.98

(B) No Person: 0.99, Water: 0.98, Beach: 0.97, Outdoors: 0.97, Seashore: 0.97

(C) No Person: 0.97, **Mammal: 0.96**, Water: 0.94, Beach: 0.94, Two: 0.94

Distribution Shift

(A) **Cow: 0.99**, Pasture: 0.99, Grass: 0.99, No Person: 0.98, Mammal: 0.98

(B) No Person: 0.99, Water: 0.98, Beach: 0.97, Outdoors: 0.97, Seashore: 0.97

(C) No Person: 0.97, **Mammal: 0.96**, Water: 0.94, Beach: 0.94, Two: 0.94

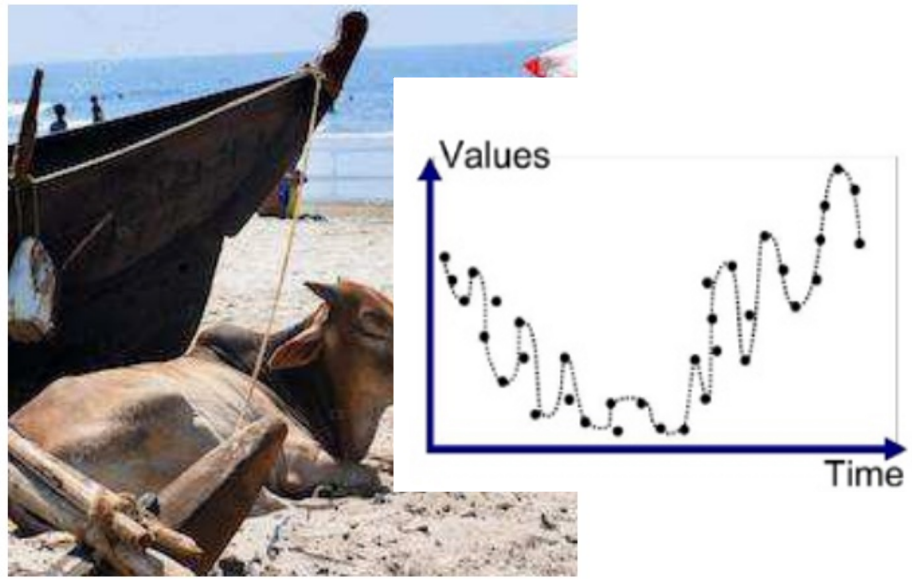
Presence of grass is a spurious correlation: predictive of examples in the test set, doesn't generalize

Distribution Shift

High dimensional analogue of this:

(A) **Cow: 0.99**, Pasture: 0.99, Grass: 0.99, No Person: 0.98, Mammal: 0.98

(B) No Person: 0.99, Water: 0.98, Beach: 0.97, Outdoors: 0.97, Seashore: 0.97



(C) No Person: 0.97, **Mammal: 0.96**, Water: 0.94, Beach: 0.94, Two: 0.94

Presence of grass is a spurious correlation: predictive of examples in the test set, doesn't generalize

Distribution Shift – Motivating Examples

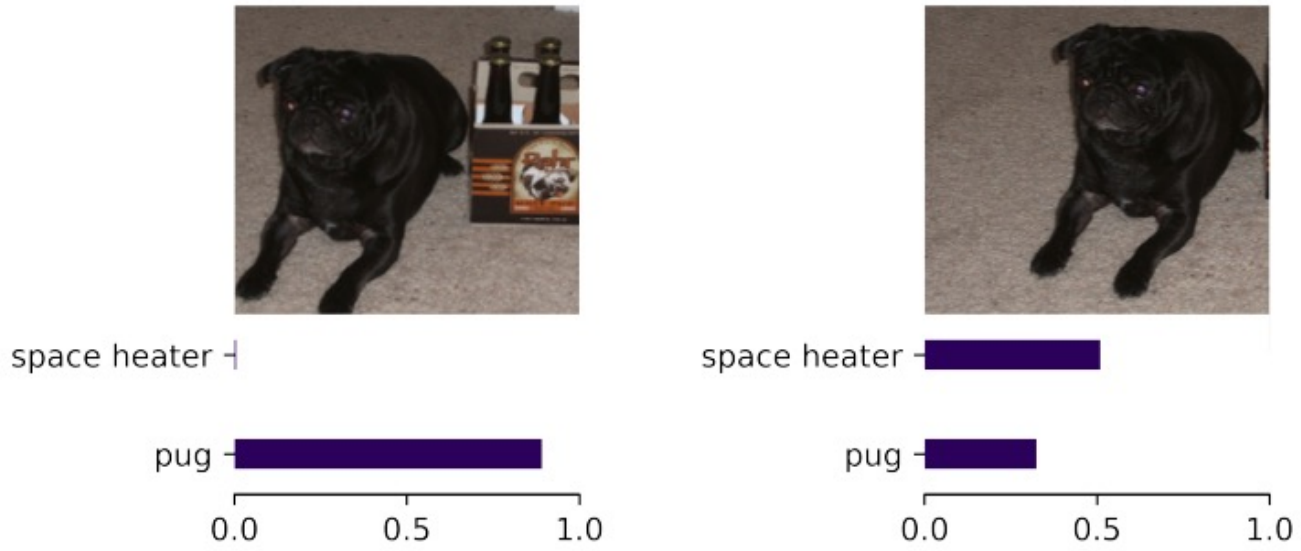


Figure 5. A correctly and an incorrectly classified sample for the BEiT-Model. The dog of class "pug" is fully visible in both images.

Causal Definition of Distribution Shift

Inputs *X* and outputs *Y*

Definition 1: Discriminative Model

Data generating process: X → Y

Example: Suppose *X* is a medical image and *Y* is segmentation created by a doctor. The image is causal of the segmentation. If we change X, then Y would also change Y

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = p_{\boldsymbol{\psi}}(\boldsymbol{x}) p_{\boldsymbol{w}}(\boldsymbol{y}|\boldsymbol{x})$$

Causal Definition of Distribution Shift

Inputs *X* and outputs *Y*

Definition 2: Generative Model

Data generating process: Y → X

Example: Suppose *X* is a medical image a *Y* is the ground truth disease state

Changing the disease state, Y, will change the appearance of the image

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = p_{\boldsymbol{\pi}}(\boldsymbol{y}) p_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{y})$$

Covariate Shift

Discriminative Model: $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = p_{\boldsymbol{\psi}}(\boldsymbol{x}) p_{\boldsymbol{w}}(\boldsymbol{y}|\boldsymbol{x})$

Consider if we change the distribution of the inputs $p_{\boldsymbol{\psi}}(\boldsymbol{x})$ from the source distribution (train) to the target distribution (test)

$$\boldsymbol{\psi}^s \neq \boldsymbol{\psi}^t$$

Covariate Shift

Discriminative Model: $p_{\theta}(x, y) = p_{\psi}(x)p_{w}(y|x)$

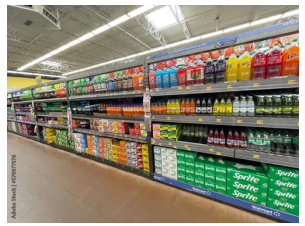
Consider if we change the distribution of the inputs $p_{\psi}(x)$ from the source distribution (train) to the target distribution (test)

$$\psi^s \neq \psi^t$$

Example: We train an object detector for products in marketing images and test on products on cluttered shelves

Source:

Target:



Concept Shift

Discriminative Model: $p_{\theta}(\boldsymbol{x}, \boldsymbol{y}) = p_{\psi}(\boldsymbol{x}) p_{w}(\boldsymbol{y}|\boldsymbol{x})$

Consider if we change the distribution of the conditional relationship between x and y, $p_w(\boldsymbol{y}|\boldsymbol{x})$ from the source distribution (train) to the target distribution (test)

$$\boldsymbol{w}^s \neq \boldsymbol{w}^t$$

Example: Different doctors use different rules and heuristics to label pixels of a medical image

Label Shift

Generative Model: $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = p_{\boldsymbol{\pi}}(\boldsymbol{y}) p_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{y})$

Consider if we change the distribution of the outputs, $p_{\boldsymbol{\pi}}(\boldsymbol{y})$ from the source distribution (train) to the target distribution (test)

$$\boldsymbol{\phi}^s \neq \boldsymbol{\phi}^t$$

Example: Suppose that the medical images about a disease are collected in an urban source environment, but the target environment is in rural areas

The prevalence of the disease may be different between the two

Manifestation Shift

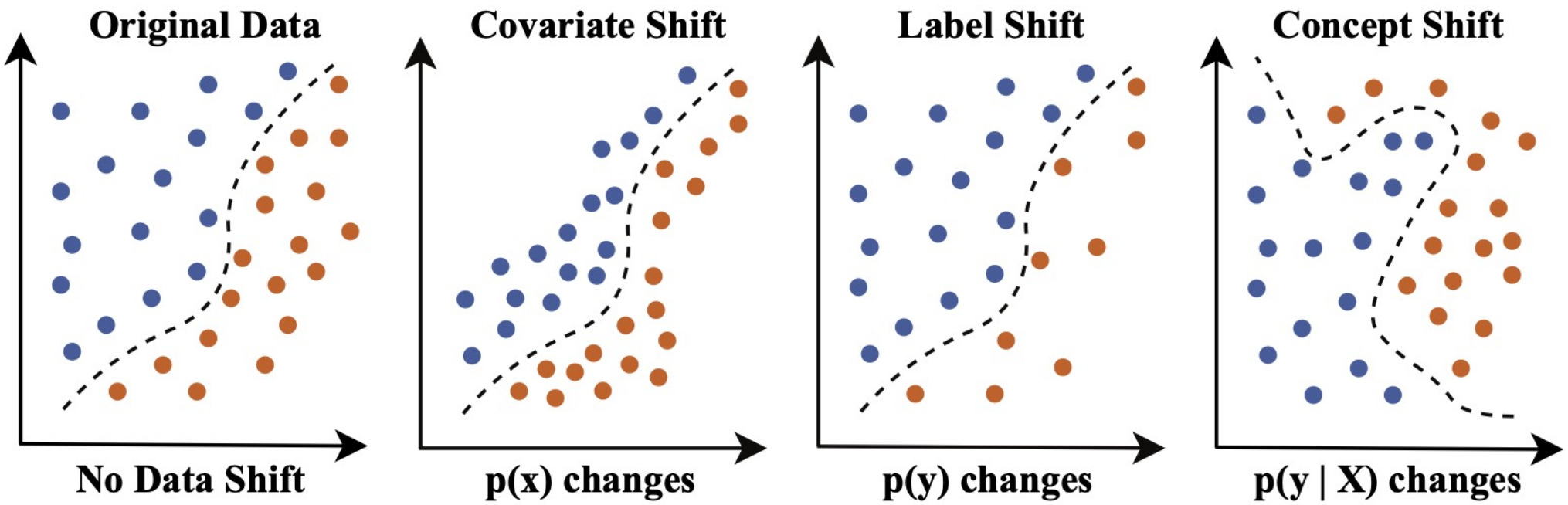
Generative Model: $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = p_{\boldsymbol{\pi}}(\boldsymbol{y}) p_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{y})$

Consider if we change the distribution of the inverse mapping, $p_{\boldsymbol{\phi}}(\boldsymbol{x}|\boldsymbol{y})$ from the source distribution (train) to the target distribution (test)

$$\boldsymbol{\phi}^s \neq \boldsymbol{\phi}^t$$

Example: The way that a disease, Y, manifests itself in image, X, changes maybe due to some other hidden variable, Z, such as age.

Different Types of Distribution Shift



Different Types of Distribution Shift

Name	Source	Target	Joint
Covariate/domain shift	$p(X)p(Y\|X)$	$q(X)p(Y\|X)$	Discriminative
Concept shift	$p(X)p(Y\|X)$	$p(X)q(Y\|X)$	Discriminative
Label (prior) shift	$p(Y)p(X\|Y)$	$q(Y)p(X\|Y)$	Generative
Manifestation shift	$p(Y)p(X\|Y)$	$p(Y)q(X\|Y)$	Generative

Table 19.1: The 4 main types of distribution shift.

Selection Bias

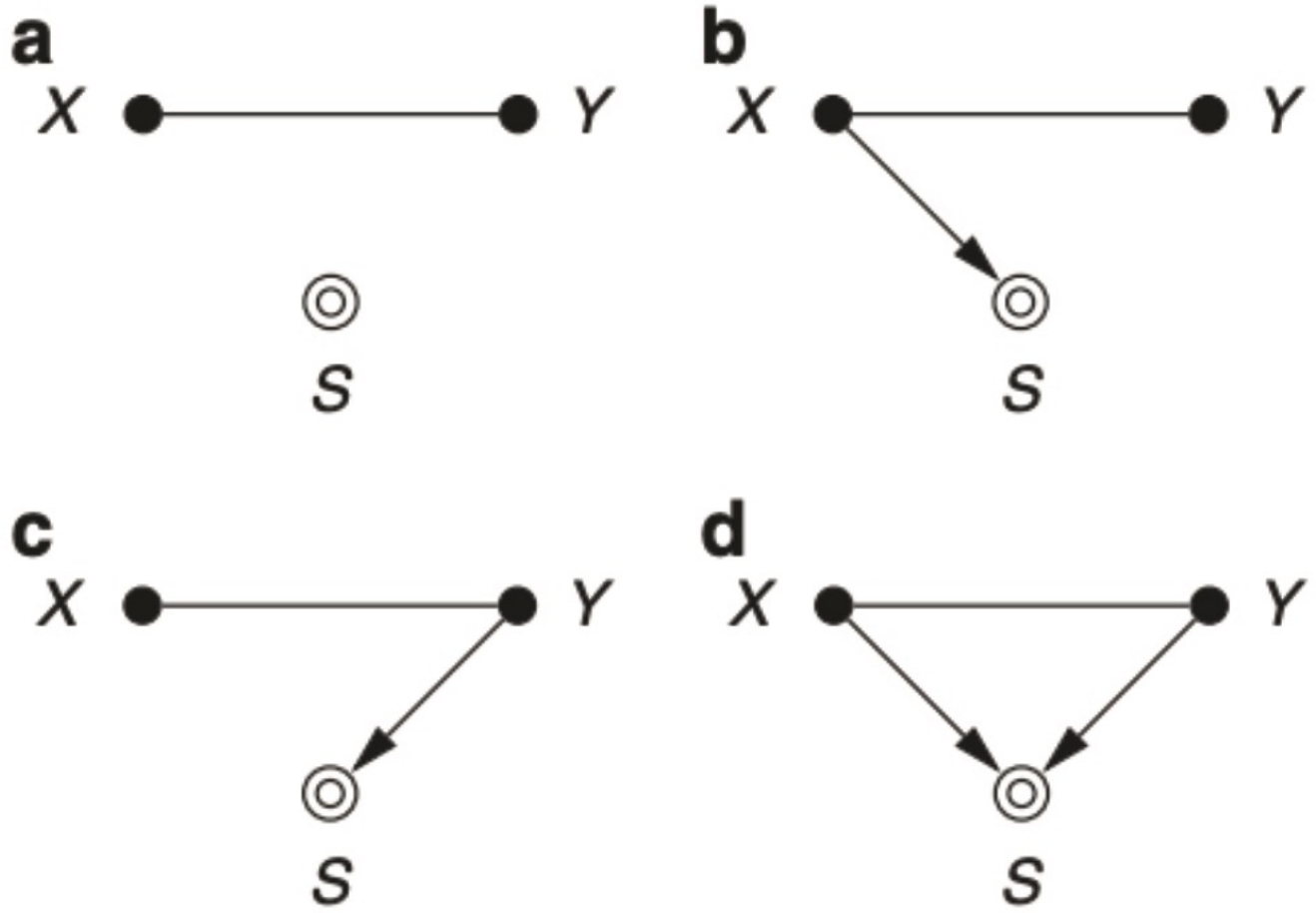
How we collect the data (train and test) can impact how our model performances

- Suppose we want train an image classifier for a production software system.
- During training we filter to clean, non-blurry images
- At test time, we don’t have this constraint

The selection mechanism for the data induces **covariate shift**

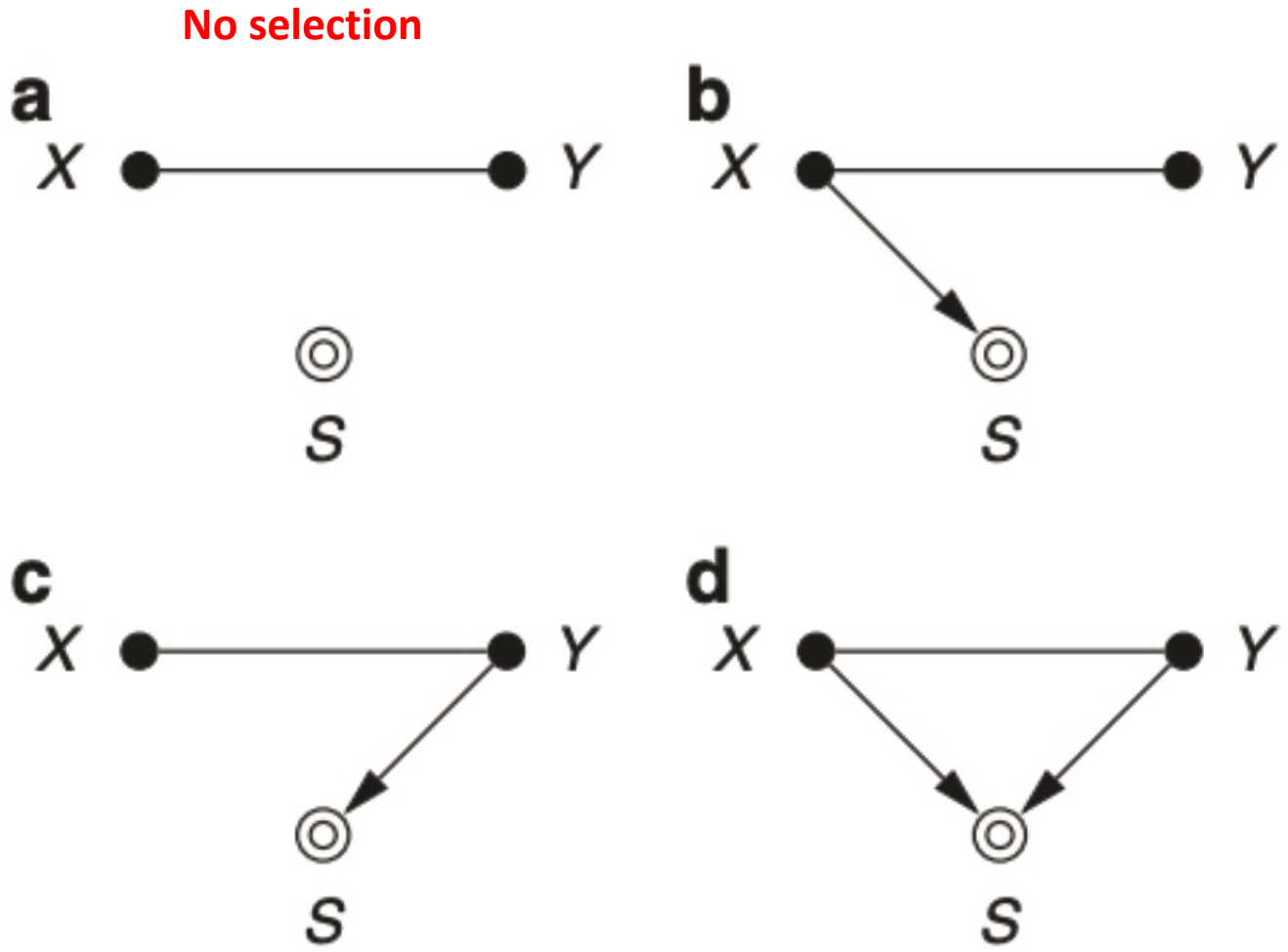
Selection Bias

Input: X
Label: Y,
S: Selection Variable {0, 1}



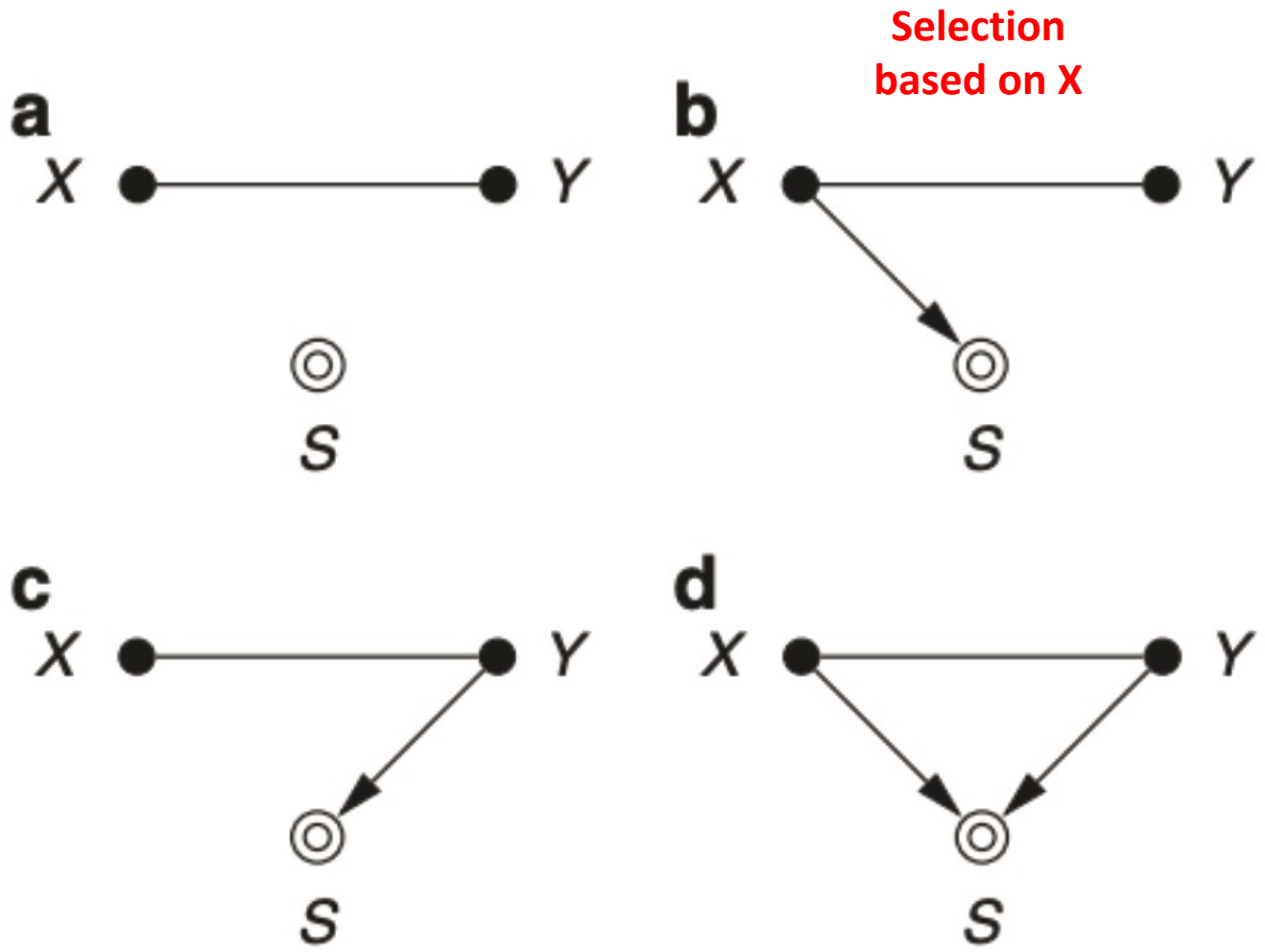
Selection Bias

Input: X
Label: Y,
S: Selection Variable {0, 1}



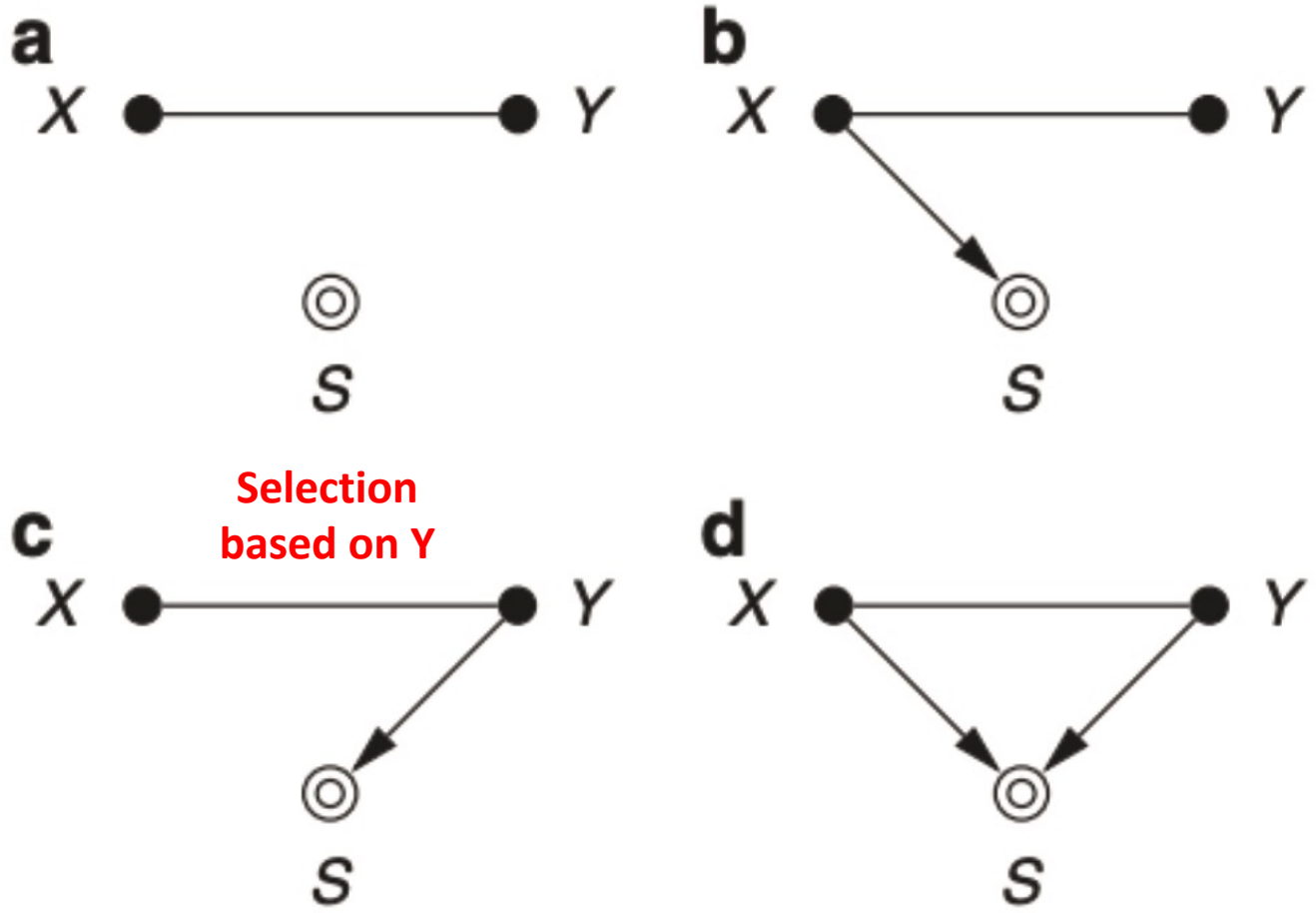
Selection Bias

Input: X
Label: Y,
S: Selection Variable {0, 1}



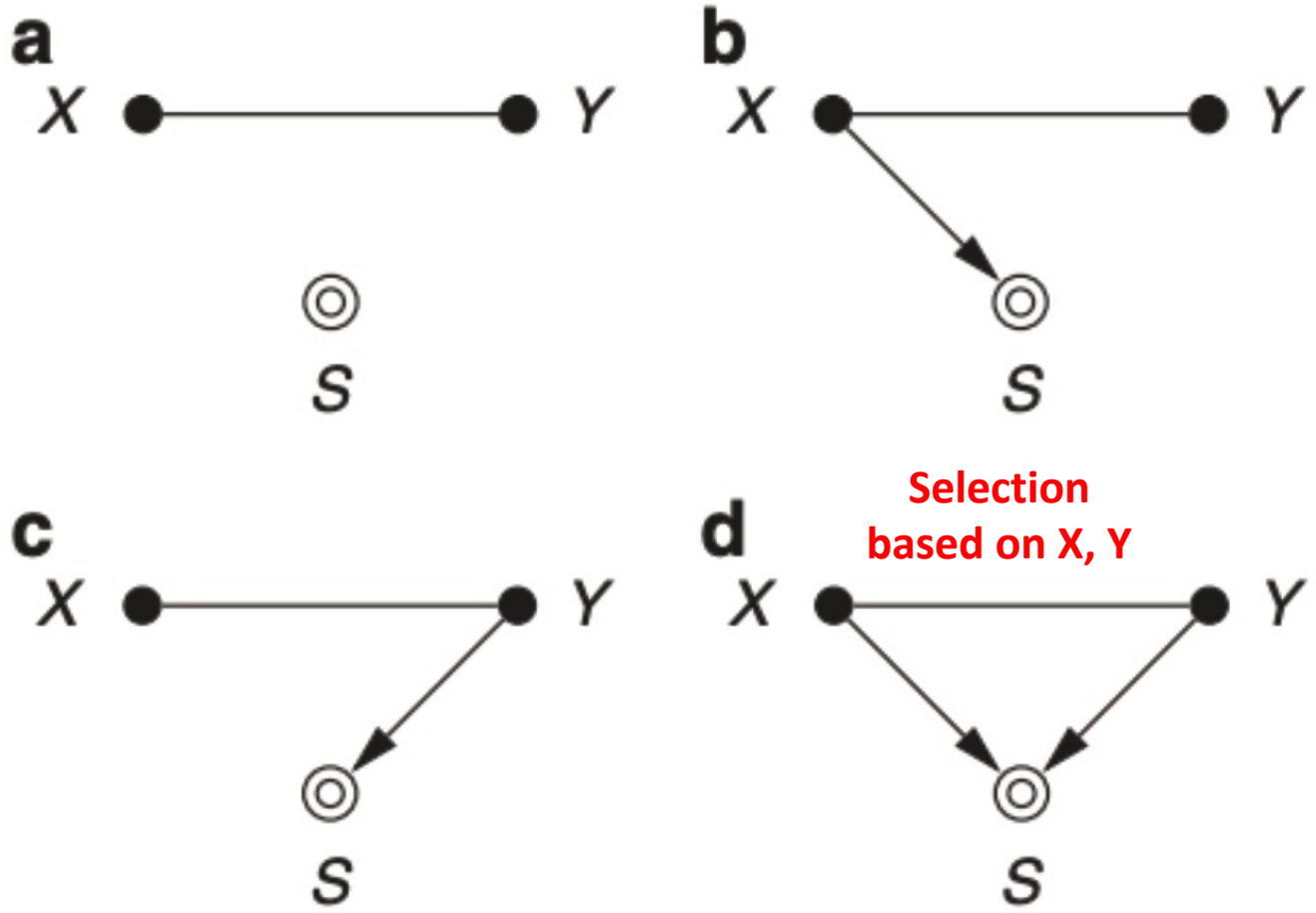
Selection Bias

Input: X
Label: Y,
S: Selection Variable {0, 1}



Selection Bias

Input: X
Label: Y,
S: Selection Variable {0, 1}



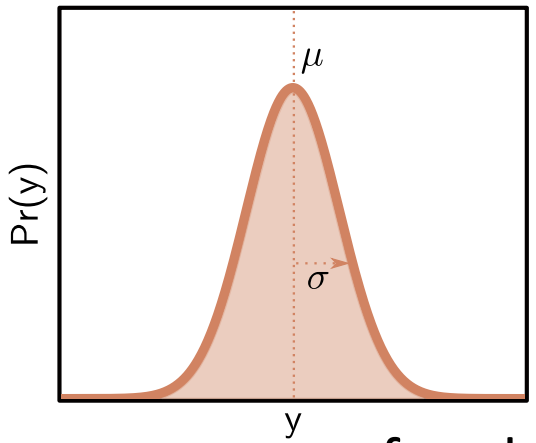
Detecting Shifts

- A variety of ideas exist to detect out-of-distribution (OOD).
- One good idea is to use networks that output a probability distribution, rather than models that output a single point prediction

How can a model predict a probability distribution?

1. Pick a known distribution (e.g., normal distribution) to model output y with parameters $\boldsymbol{\theta}$

   e.g., the normal distribution $\boldsymbol{\theta} = \{\mu, \sigma^2\}$



2. Use model to predict parameters of probability distribution $\boldsymbol{\theta}$

- ~~$y_i = f(x_i ; \varphi)$~~
- $\mu_i, \sigma_i = f(x_i ; \varphi)$

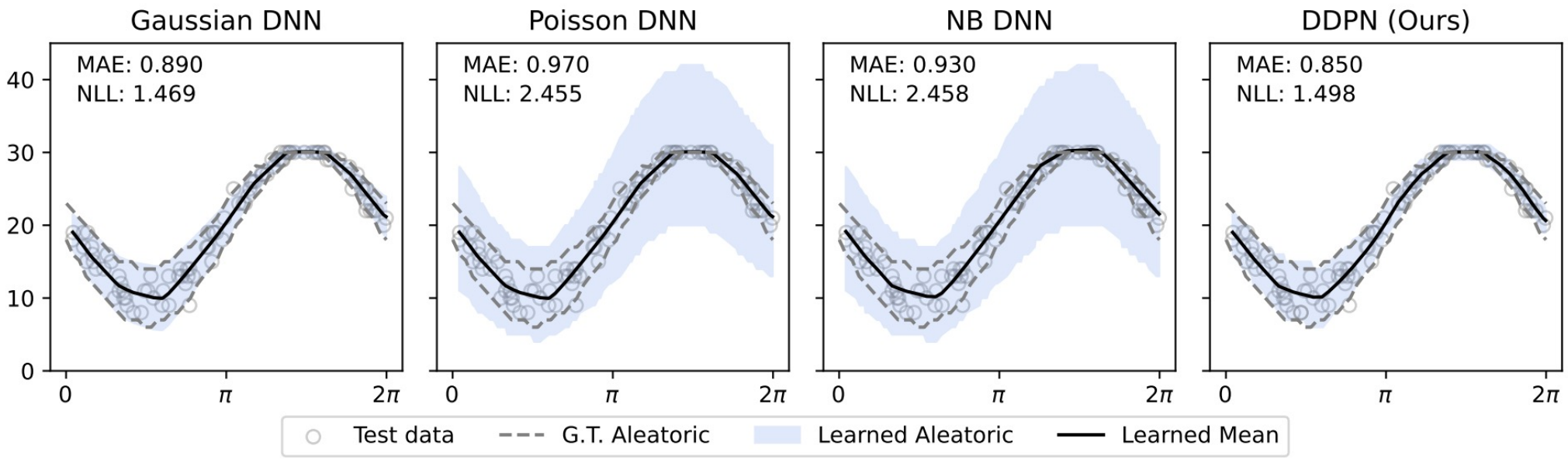
Heteroscedastic regression

- Previously assumed that the noise σ^2 is the same everywhere.
- But we could make the noise a function of the data *x.*
- Build a model with two outputs:

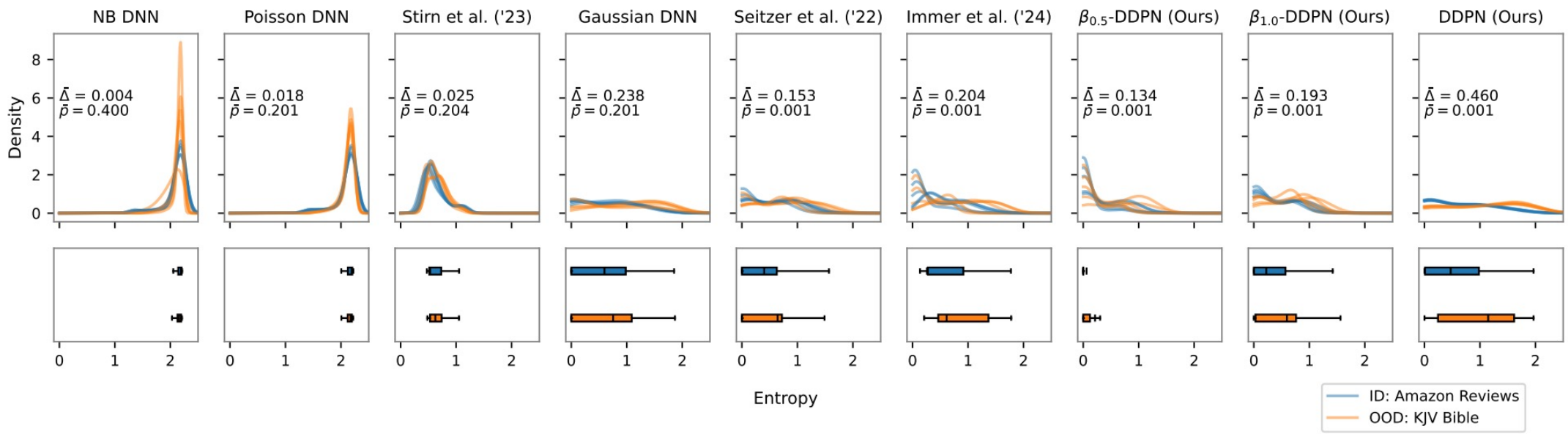
$$\mu = \mathrm{f}_1[\mathbf{x}, \boldsymbol{\phi}]$$
$$\sigma^2 = \mathrm{f}_2[\mathbf{x}, \boldsymbol{\phi}]^2$$

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\mathrm{argmin}} \left[-\sum_{i=1}^{I} \log \left[\frac{1}{\sqrt{2\pi \mathrm{f}_2[\mathbf{x}_i, \boldsymbol{\phi}]^2}} \right] - \frac{(y_i - \mathrm{f}_1[\mathbf{x}_i, \boldsymbol{\phi}])^2}{2\mathrm{f}_2[\mathbf{x}_i, \boldsymbol{\phi}]^2} \right]$$

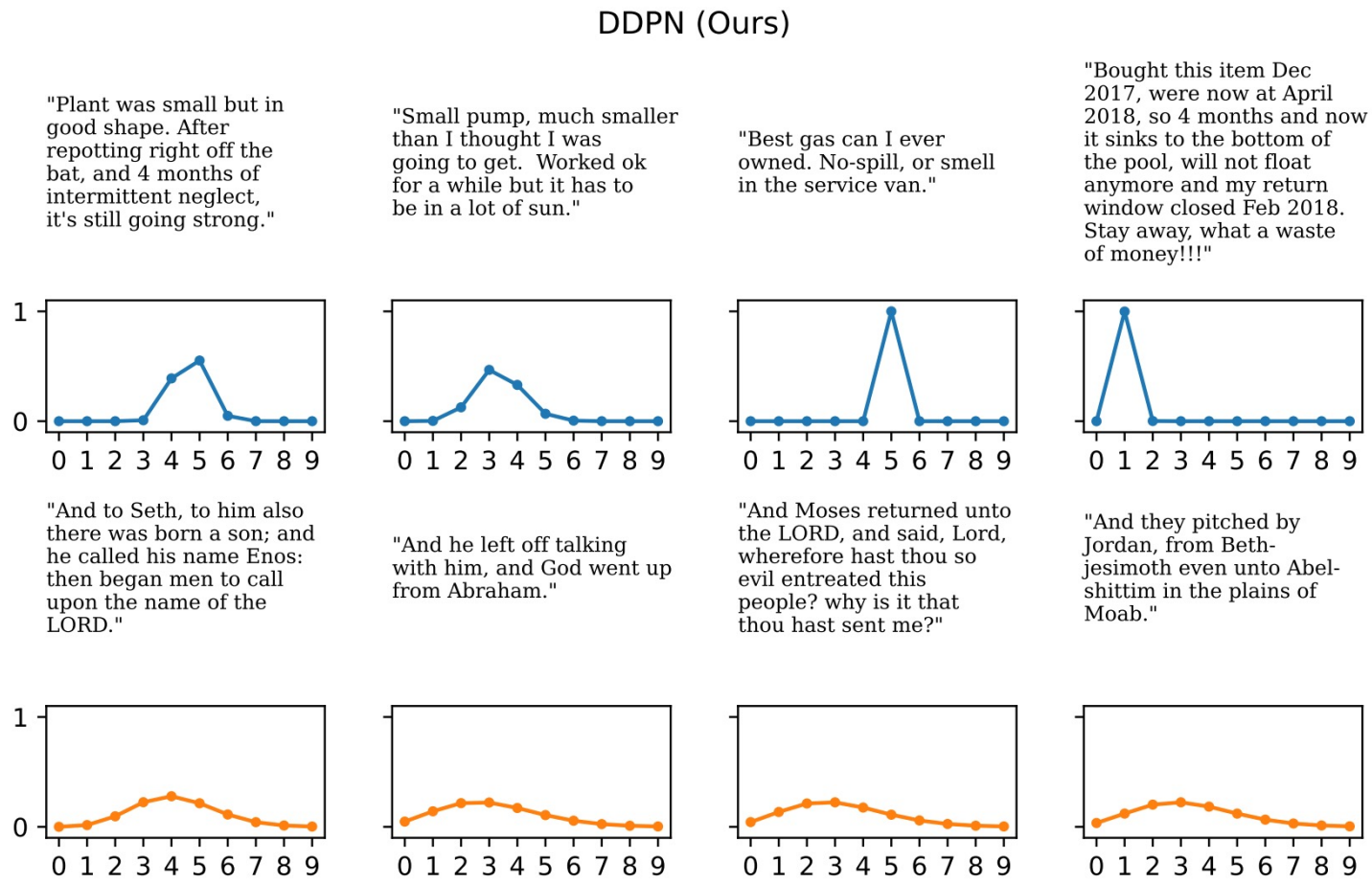
Detecting Shifts



Entropy of Probabilistic Models



Entropy of Probabilistic Models



Questions?