Model Capacity, Regularization, and Hyperparameters

Danna Gurari
University of Colorado Boulder
Spring 2025

https://dannagurari.colorado.edu/course/neural-networks-and-deep-learning-spring-2025/

Review

- Last lecture:
  - Motivation: effective gradients for learning
  - Initializing parameters
  - Initializing data
  - Following the gradient (optimization)
  - Programming tutorial

- Assignments (Canvas):
  - Problem set 2 due on Tuesday

- Questions?

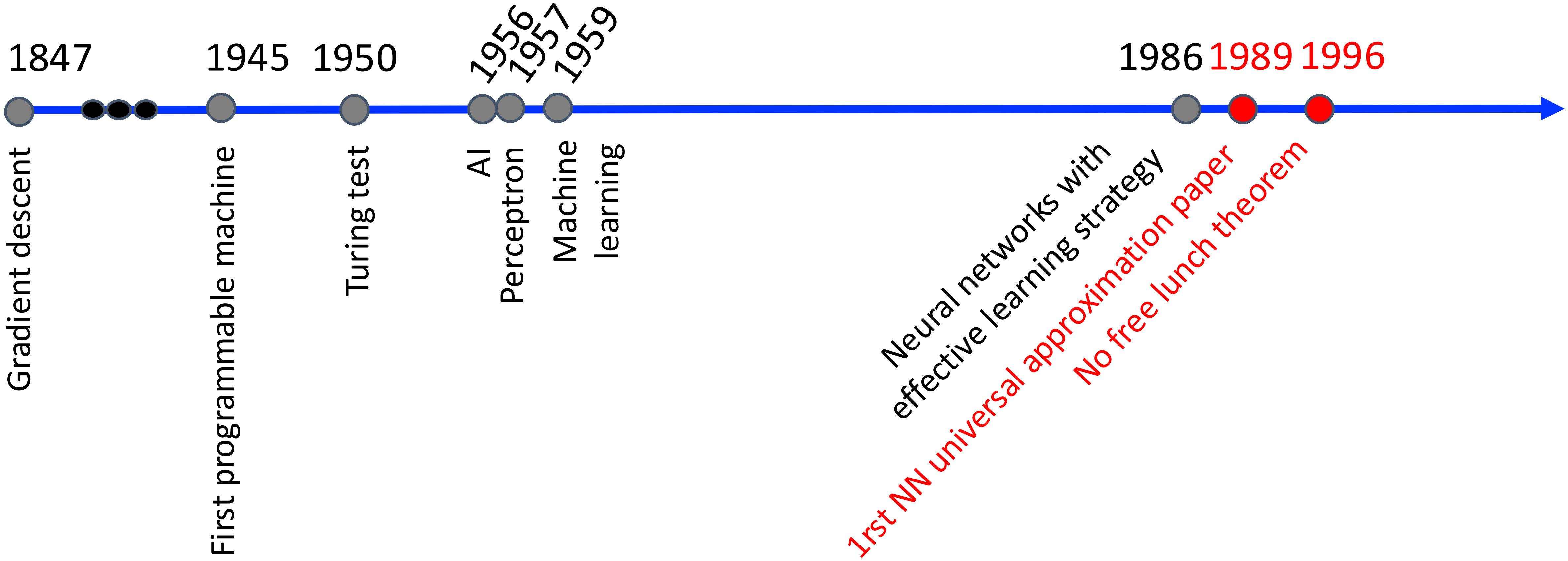
Today’s Topics

- Model capacity: how it affects learning
- Regularization: learning methods for improving model generalization
- Hyperparameter selection: tuning to improve model performance
- Programming tutorial

Today’s Topics

- Model capacity: how it affects learning
- Regularization: learning methods for improving model generalization
- Hyperparameter selection: tuning to improve model performance
- Programming tutorial

Historical Context: Motivating Theory



Hornik, Stinchcombe and White. Multilayer feedforward networks are universal approximators. Neural Networks, 1989

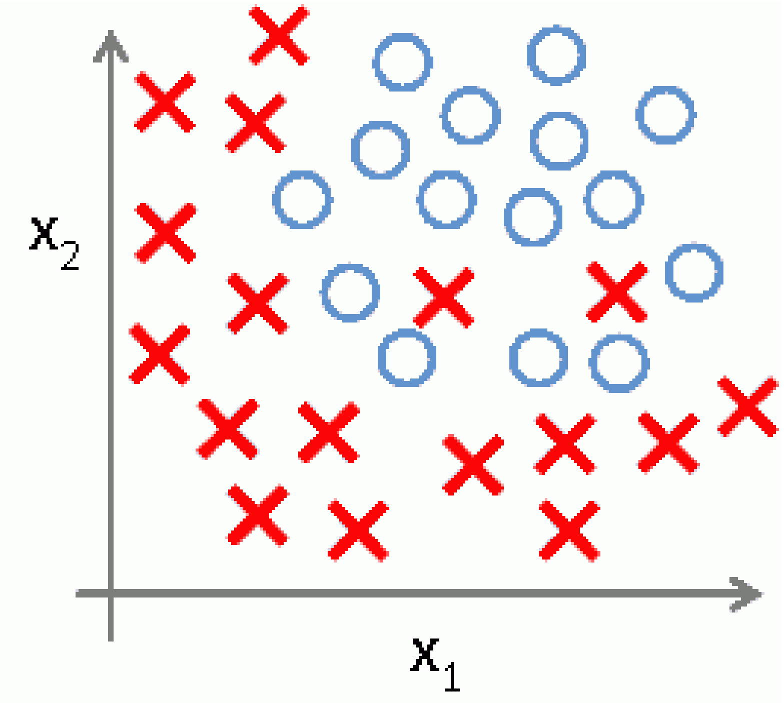
What model design should we use?

“no free lunch theorem… no machine learning algorithm is universally any better than any other.” - Ch. 5.2.1 of Goodfellow book

“The universal approximation theorem means that regardless of what function we are trying to learn, we know that a large MLP [multilayer perceptron] will be able to *represent* this function.” - Ch. 6.4.1 of Goodfellow book

Model Capacity: Recall Class Exercise

- Model-based approach: separate x from o



Class volunteer:
1) Draw a straight line (linear equation)
2) Draw a parabola (quadratic equation)
3) Draw any curve

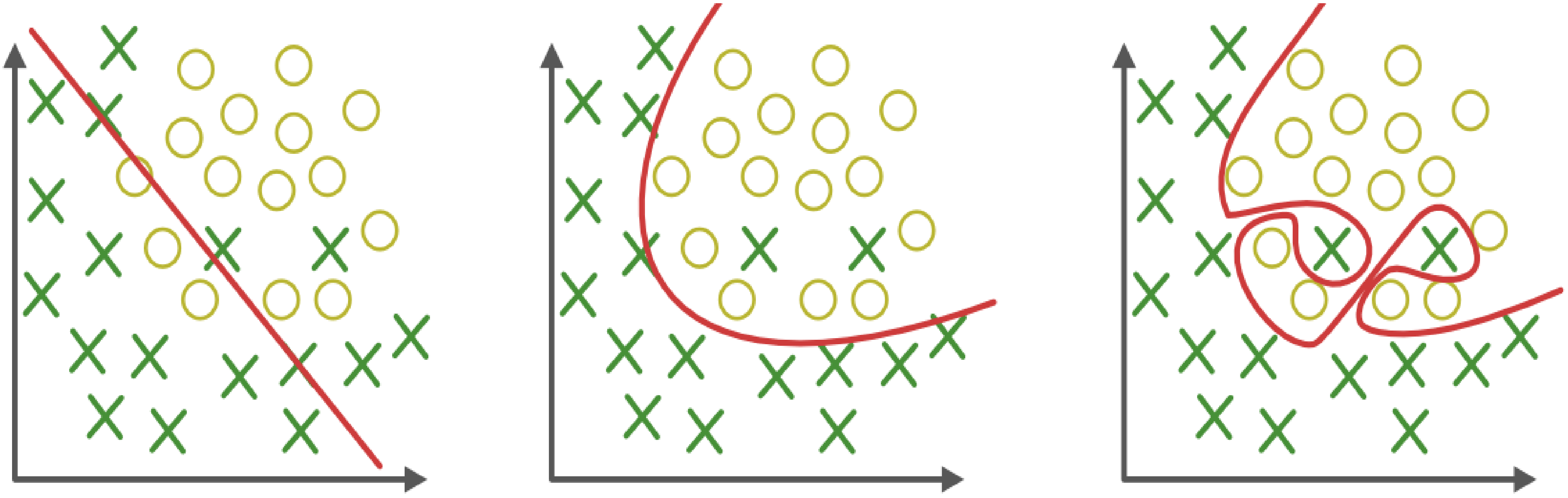
Models with increasing representational capacity

https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76

Model Capacity

Which model would you choose to separate x from o?

(a) (b) (c)

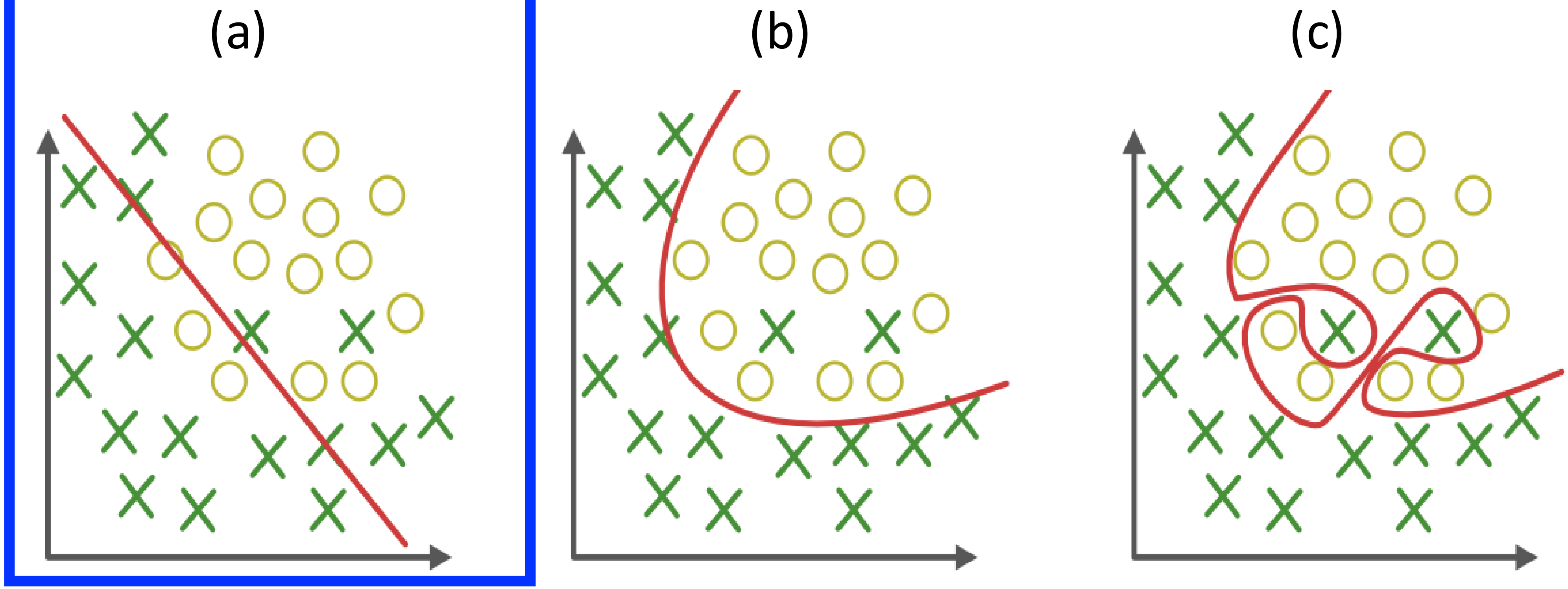


https://towardsdatascience.com/underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6fe4a8a49dbf

Model Capacity

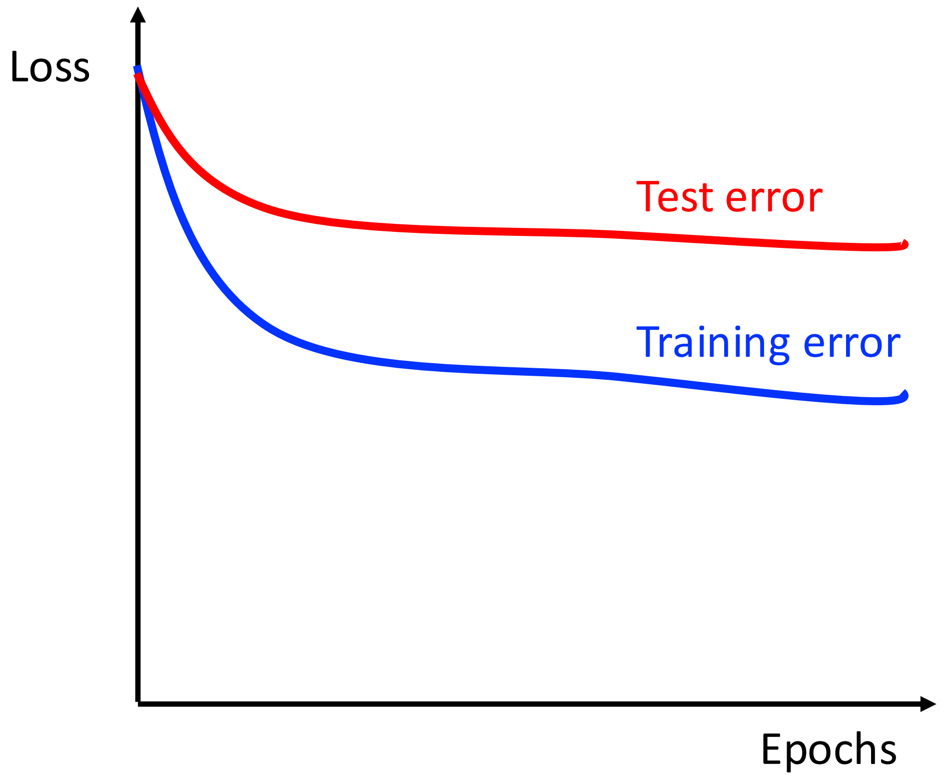
Underfits: too simple to explain the data

Overfits: too complex to generalize to a test set



https://towardsdatascience.com/underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6fe4a8a49dbf

Model Capacity: Detecting Underfitting



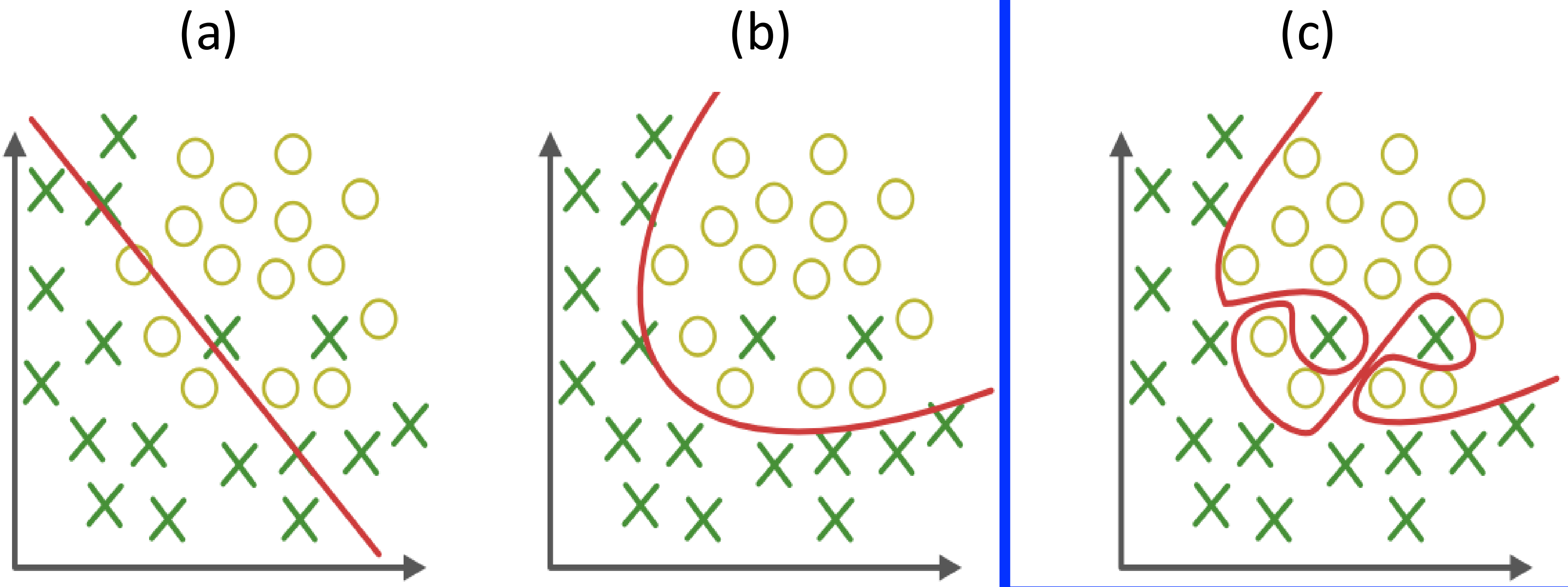
Analyze **learning curves** for models tested on training data

- What happens to training data error as number of training steps increases?
  - Error remains high
- What happens to test data error as number of training steps increases?
  - Error remains high

Model Capacity

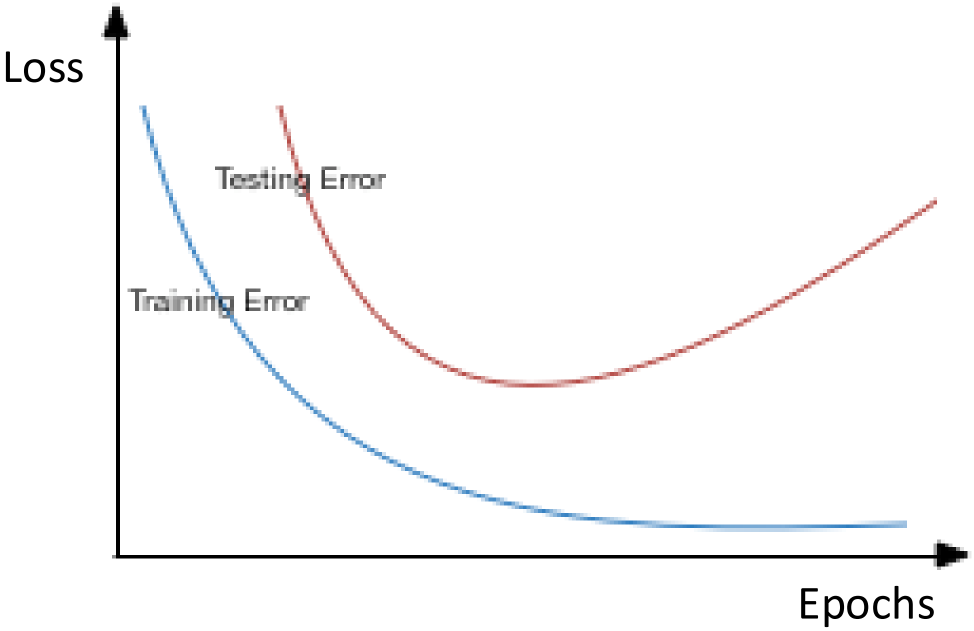
Underfits: too simple to explain the data

Overfits: too complex to generalize to a test set



https://towardsdatascience.com/underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6fe4a8a49dbf

Model Capacity: Detecting Overfitting



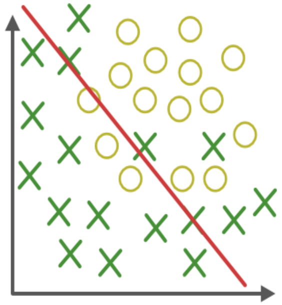
Analyze **learning curves** for model on training data and test data

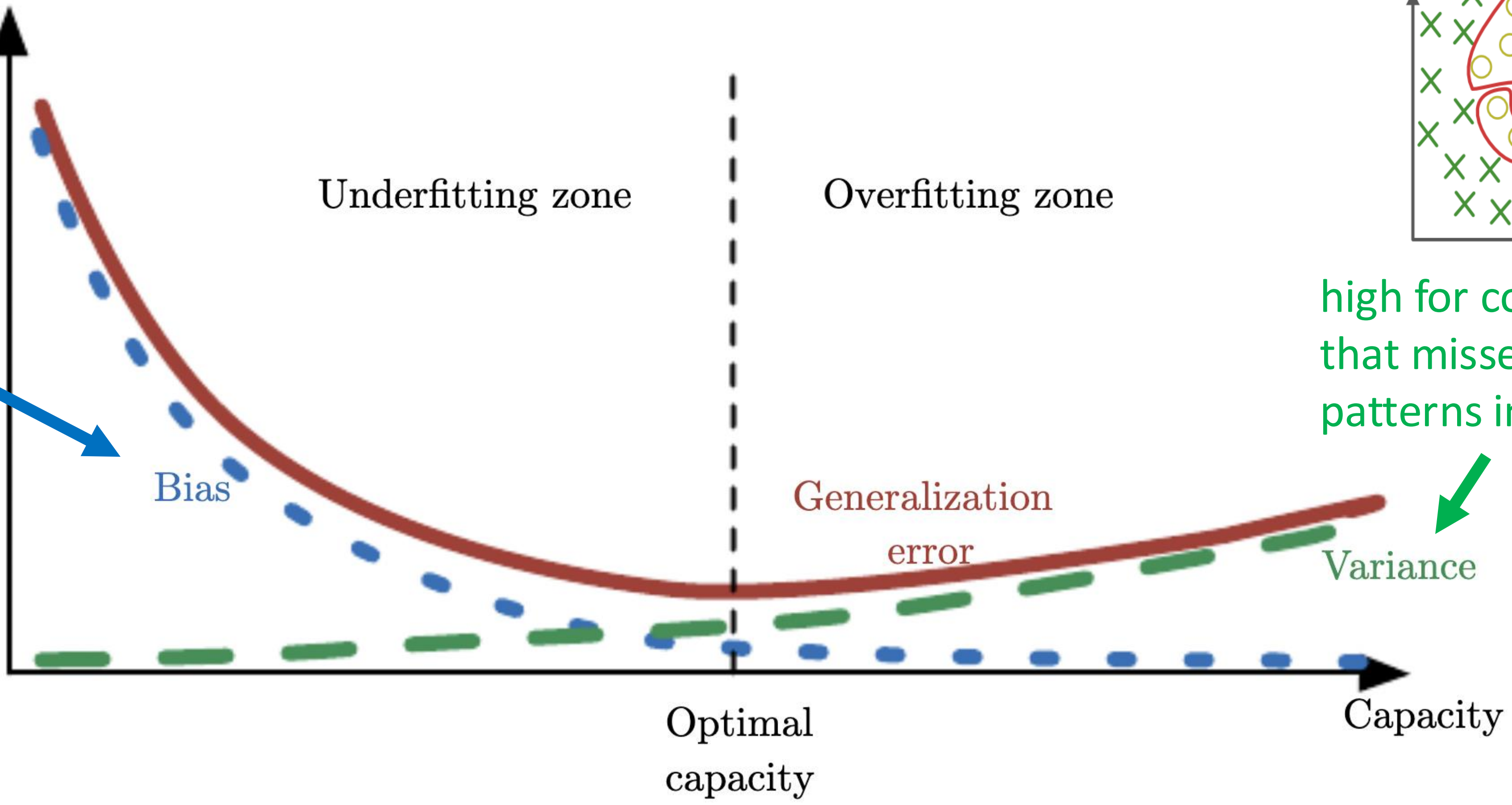
- What happens to training data error as number of training steps increases?
  - Error shrinks
- What happens to test data error as number of training steps increases?
  - Error shrinks and then grows
- Why does training error ***shrink*** and test error ***grow***?
  - Models ***noise*** in the training data (i.e., "overfitting") at the expense of knowledge that generalizes
- What can cause noise in a dataset?
  - e.g., incorrect data entry/labeling, hardware measurement error (caution: some outliers are data points we want models to learn)

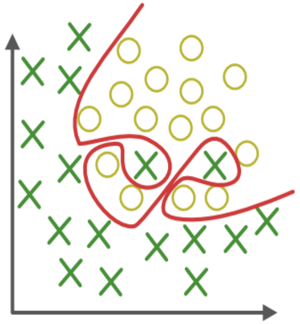
https://chatbotslife.com/regularization-in-deep-learning-f649a45d6e0

Model Capacity: Overfitting vs Underfitting

high for simple model that assumes simpler patterns in the data (not related to “bias” parameter):





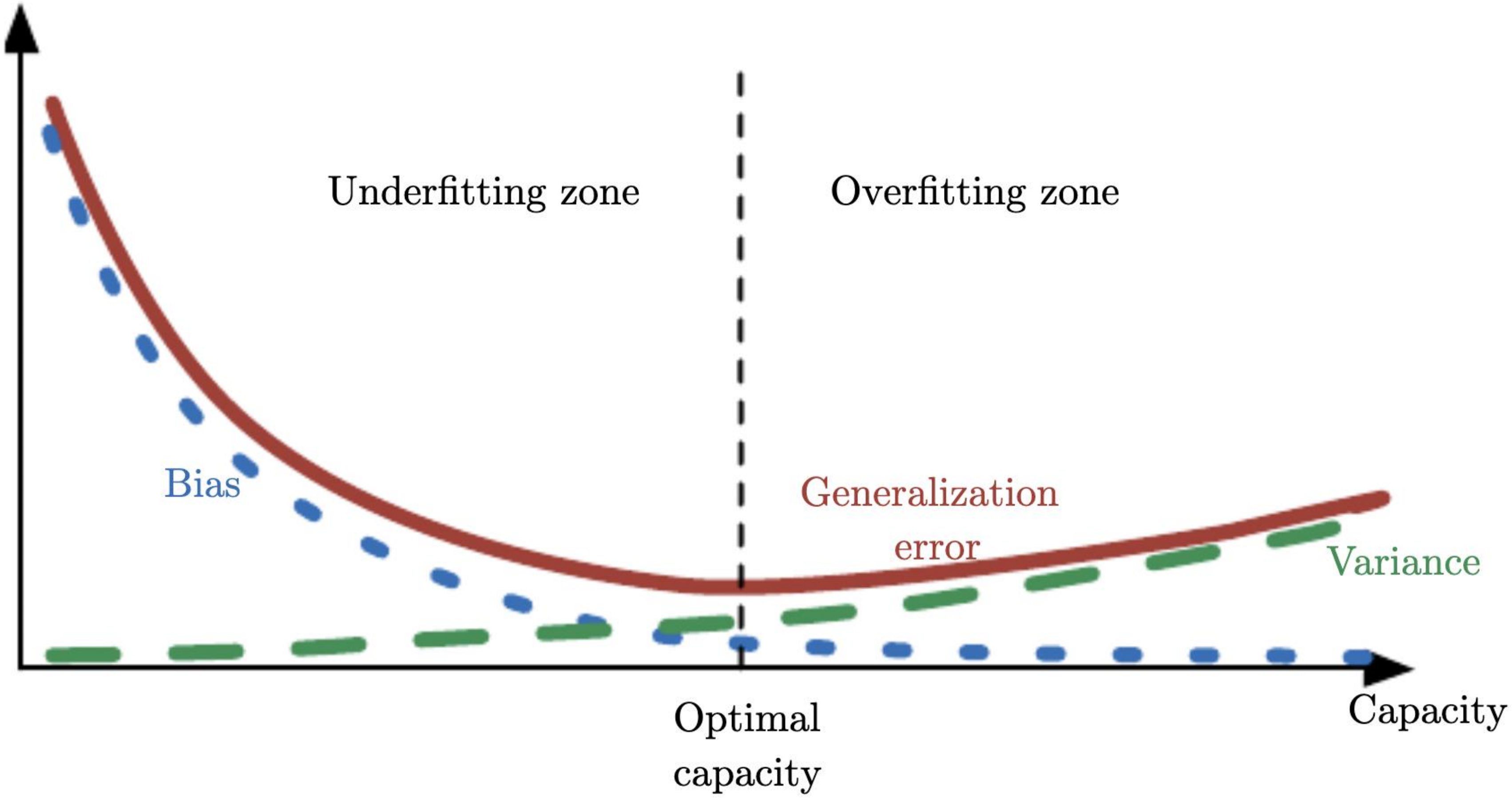


high for complex model that misses simpler patterns in the data

Often discussed with respect to a **bias-variance** trade-off

Ian Goodfellow, Yoshua Bengio, and Aaron Courville; Deep Learning, 2016

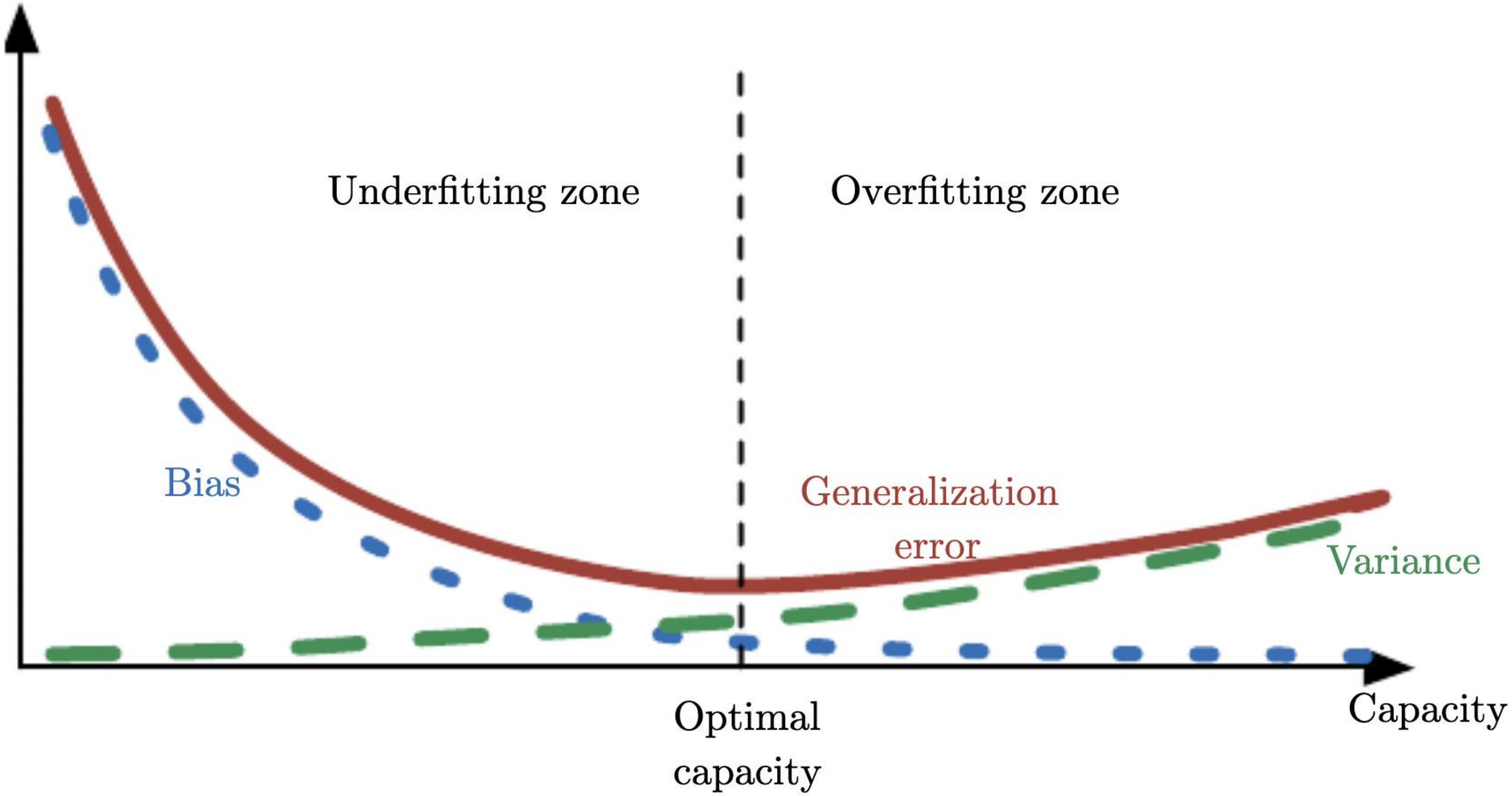
Model Capacity: Overfitting vs Underfitting



Can shift between underfitting and overfitting by adding/removing parameters

Ian Goodfellow, Yoshua Bengio, and Aaron Courville; Deep Learning, 2016

Model Capacity: Overfitting vs Underfitting

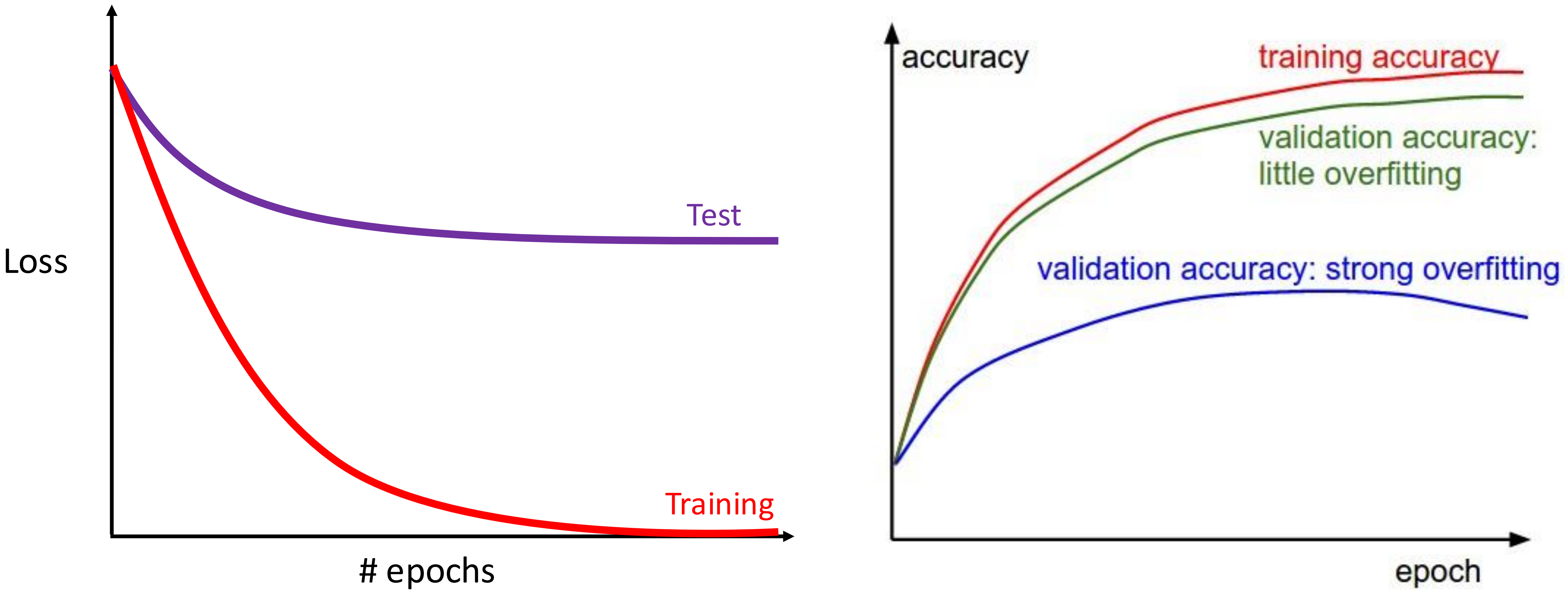


Often, we err on over-parameterized models capable of overfitting and then regularize them

Ian Goodfellow, Yoshua Bengio, and Aaron Courville; Deep Learning, 2016

During Training, You Should Ask Yourself: What Does the Observed Performance Mean?

- Loss and performance curves can signal how well training is going



https://cs231n.github.io/neural-networks-3/#update

Today’s Topics

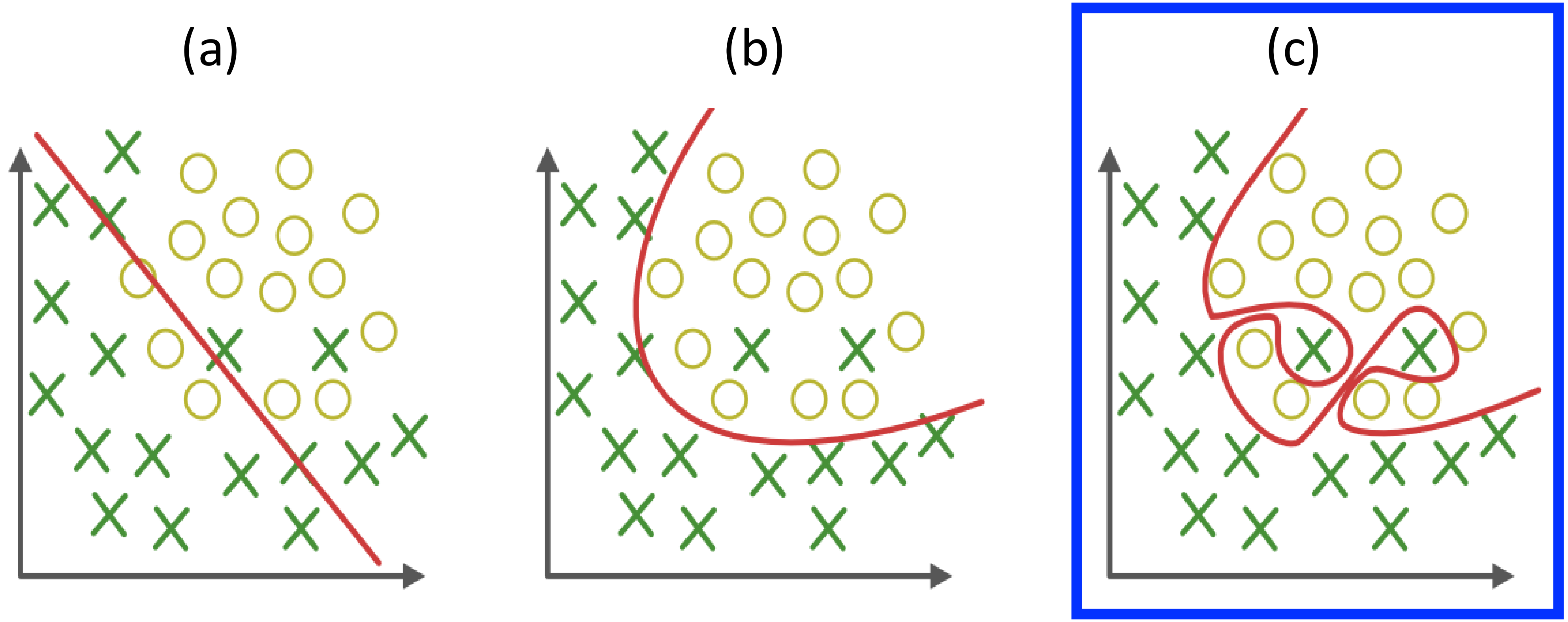
- Model capacity: how it affects learning
- Regularization: learning methods for improving model generalization
- Hyperparameter selection: tuning to improve model performance
- Programming tutorial

What is Regularization?

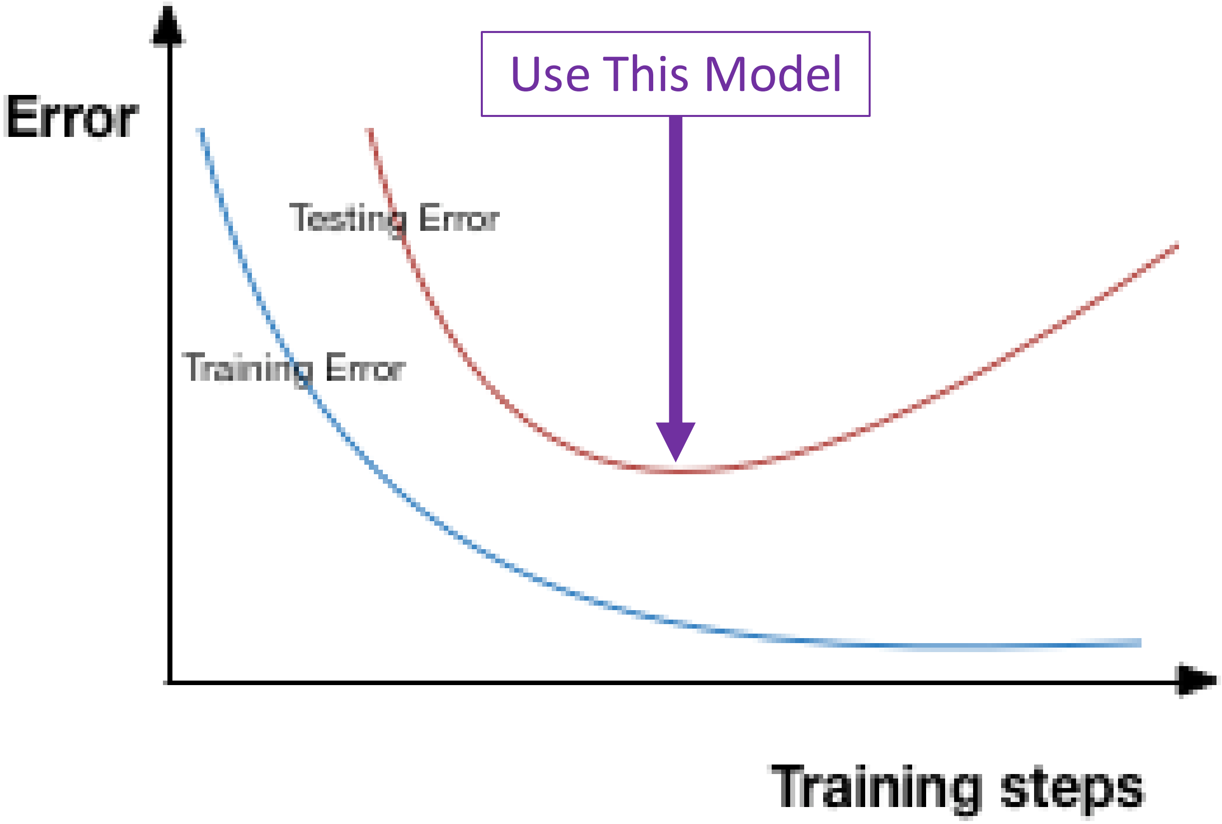
- “any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.”- Ch. 5.2 of Goodfellow book

Regularization techniques will be a focus for much of the course

e.g., Regularize Over-Parameterized Model



Example 1: Early Stopping During Training

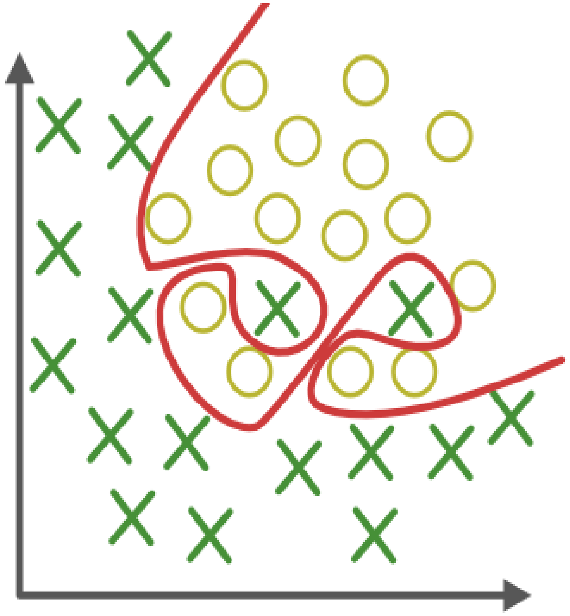


https://chatbotslife.com/regularization-in-deep-learning-f649a45d6e0

Example 2: Parameter Norm Penalty

Smooth a model's decision boundaries by incentivizing against large weights

(models can learn to represent noise with correlated large positive and negative weights that usually just cancel each other out)



https://towardsdatascience.com/underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6fe4a8a49dbf

Example 2: Parameter Norm Penalty

Add penalty term to objective function; e.g., when *minimizing* sum of squared errors

- L2 norm: penalize squared weight values

$$Error = \boxed{\sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2} + \alpha \boxed{\sum_{j=1}^{m} w_j^2}$$

- L1 norm: penalize absolute weight values

$$Error = \boxed{\sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2} + \alpha \boxed{\sum_{j=1}^{m} |w_j|}$$

- *Note: penalizes only weight (not biases) and can apply per layer or globally*

Analogy: Belt for Big Pants

Example 2: Parameter Norm Penalty

Add penalty term to objective function; e.g., when *minimizing* sum of squared errors

- L2 norm: penalize squared weight values

$$Error = \boxed{\sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2} + \boxed{\alpha} \boxed{\sum_{j=1}^{m} w_j^2}$$

- L1 norm: penalize absolute weight values

$$Error = \boxed{\sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2} + \boxed{\alpha} \boxed{\sum_{j=1}^{m} |w_j|}$$

Analogy: Belt for Big Pants

- Hyperparameter determines contribution of norm penalty term (e.g., belt tightness)

Example 2: Parameter Norm Penalty

Add penalty term to objective function; e.g., when *minimizing* **s**um of **s**quared **e**rrors

- L2 norm: penalize squared weight values

$$Error = \boxed{\sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2} + \boxed{\alpha} \boxed{\sum_{j=1}^{m} w_j^2}$$

- L1 norm: penalize absolute weight values

$$Error = \boxed{\sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2} + \boxed{\alpha} \boxed{\sum_{j=1}^{m} |w_j|}$$

Intuitively, larger alpha values prioritizes having weights closer to 0 instead of minimizing sum of squared errors

- Hyperparameter determines contribution of norm penalty term (e.g., belt tightness)

Example 2: Parameter Norm Penalty

Add penalty term to objective function; e.g., when *minimizing* **s**um of **s**quared **e**rrors

- L2 norm: penalize squared weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2 + \alpha \boxed{\sum_{j=1}^{m} w_j^2}$$

- L1 norm: penalize absolute weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2 + \alpha \boxed{\sum_{j=1}^{m} |w_j|}$$

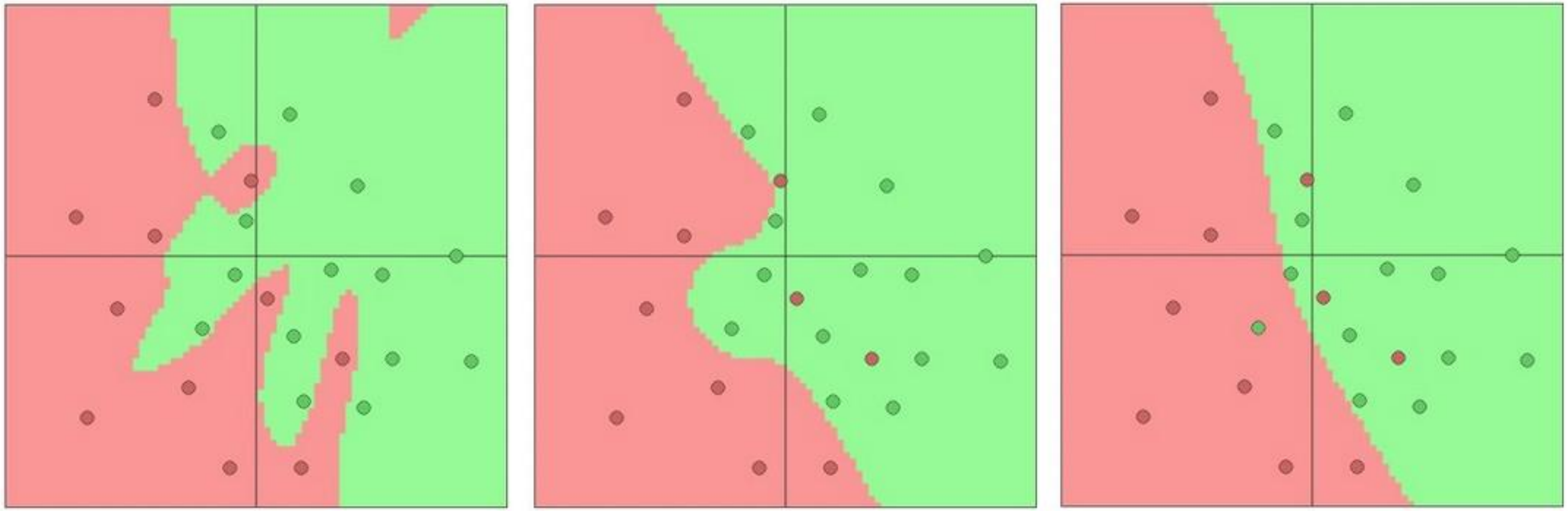
- Only change for learning is need gradient for norm penalty (shown in assigned reading)

Example 2: Parameter Norm Penalty

Which model had the largest value for alpha (i.e., norm penalty contribution), given that this value was the only difference when training the models?

$$Error = \sum_{i=1}^{n}(y^{(i)} - \widehat{y}^{(i)})^2 + \boxed{\alpha} \sum_{j=1}^{m} w_j^2$$

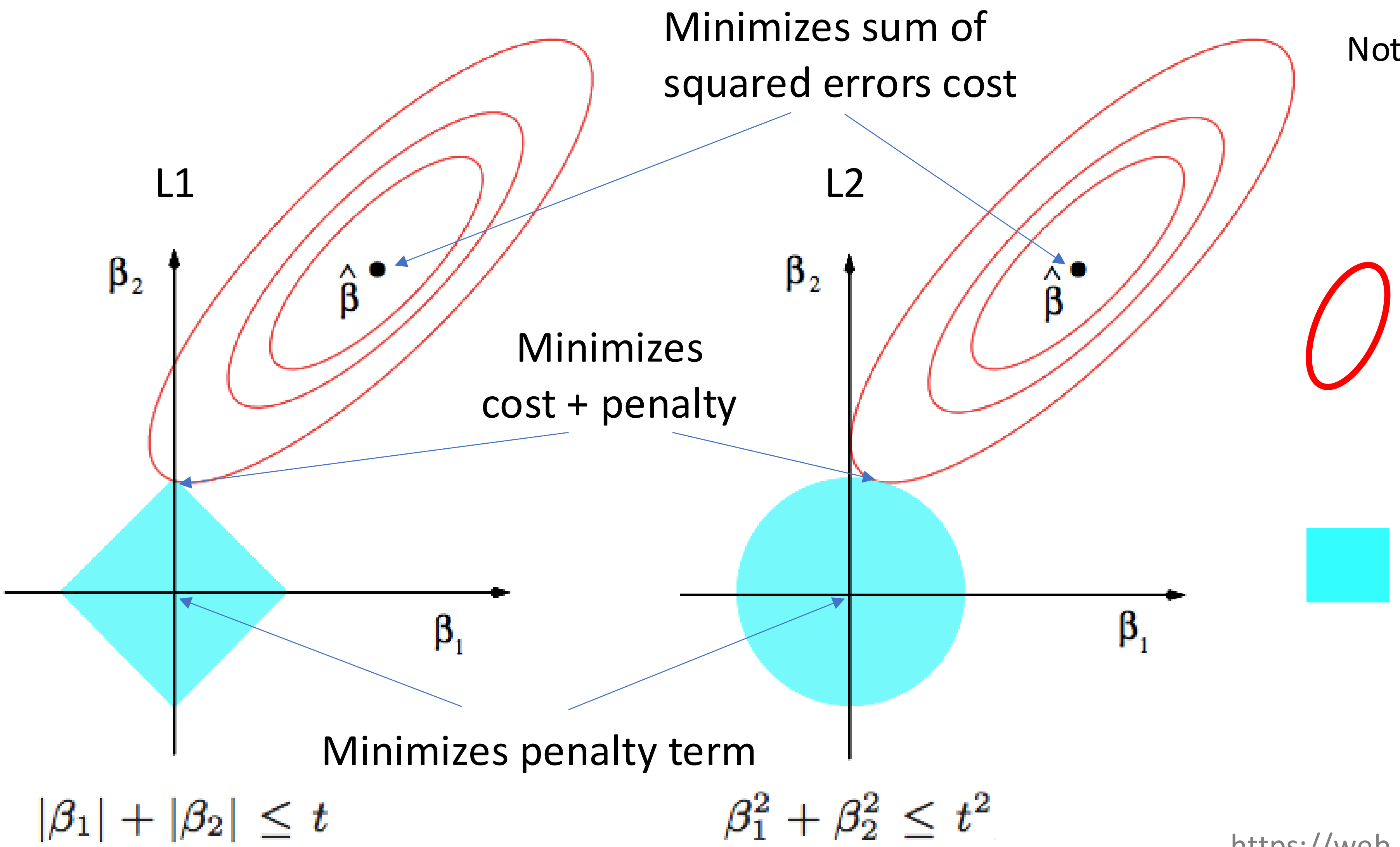
(a) (b) (c)



https://cs231n.github.io/neural-networks-1/

Example 2: Parameter Norm Penalty (Geometric Interpretation in 2D)

Note: L2 commonly used in practice



Contour of least square error function

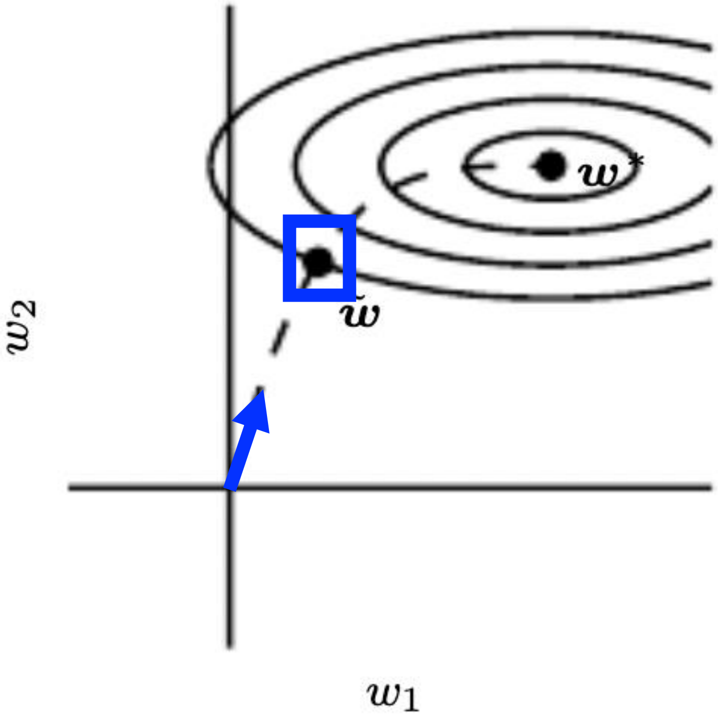
Contour of regularization constraint functions shown below both plots

https://web.stanford.edu/~hastie/Papers/ESLII.pdf

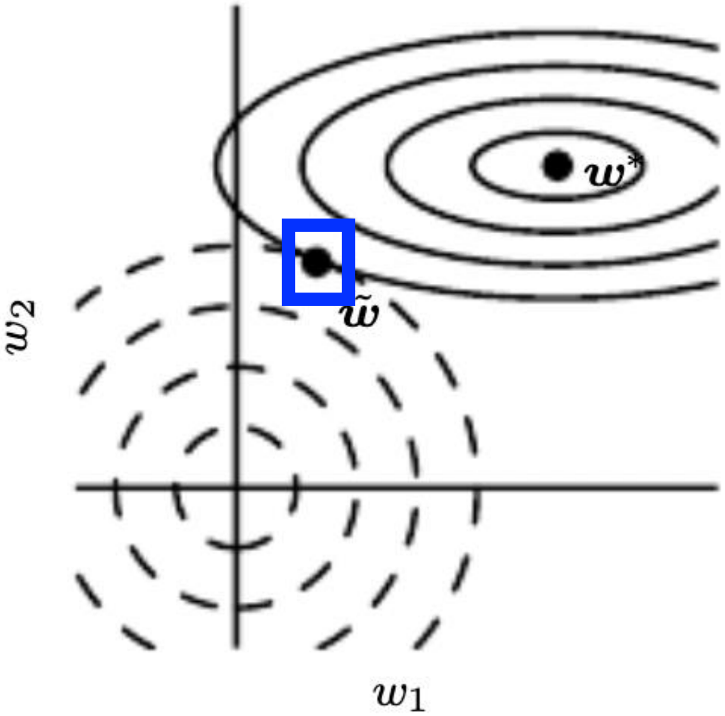
Early Stopping vs Parameter Norm Penalties

Similar behavior when model parameters initialized around origin; e.g.,

Early Stopping



Parameter Norm Penalty (L2)



https://www.deeplearningbook.org/contents/regularization.html

Today’s Topics

- Model capacity: how it affects learning
- Regularization: learning methods for improving model generalization
- Hyperparameter selection: tuning to improve model performance
- Programming tutorial

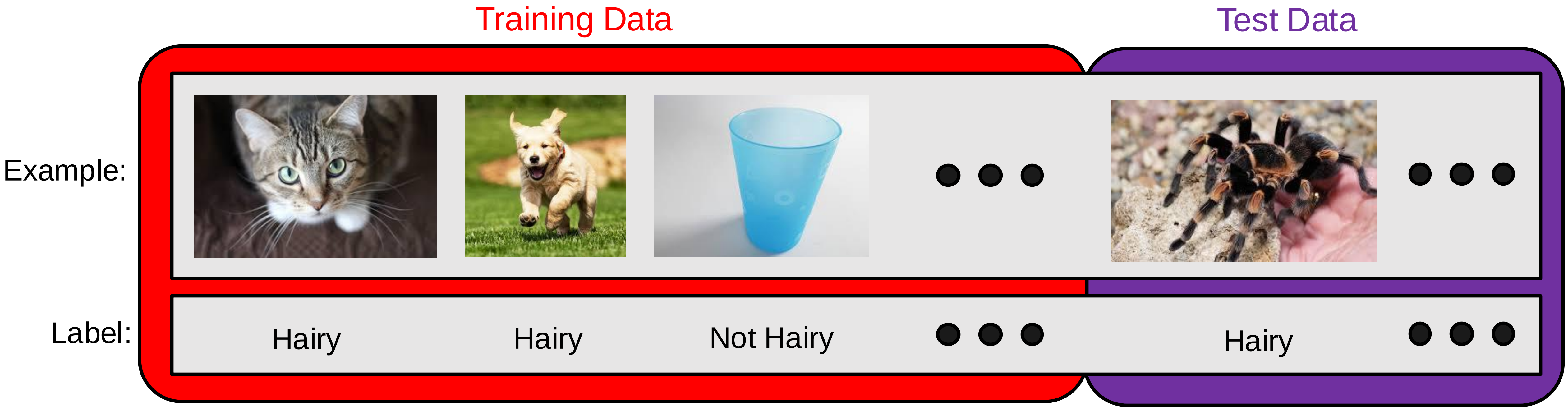
Model Design Decisions

Hyperparameters (selected); e.g.,
- Number of layers
- Number of units in each layer
- Activation function
- Batch size
- Learning rate
- …

Parameters (learned)
- Weights
- Biases
- …

Challenge: how to design a model without repeatedly observing the test data (which leads to overfitting)?

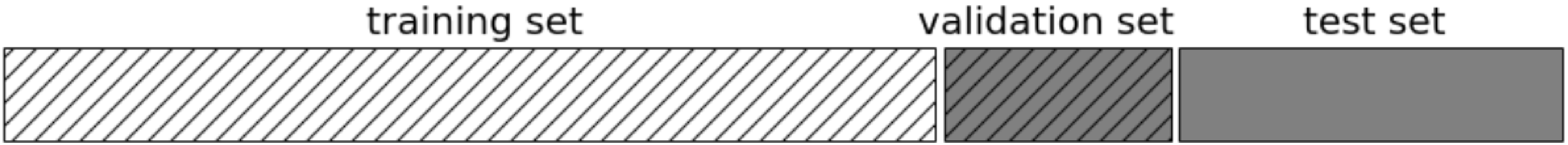
Recall: Our Goal is to Design Models that **Generalize** Well to New, Previously Unseen Examples (Test Data)



Challenge: how to design a model without repeatedly observing the test data (which leads to overfitting)?

Validation Dataset for Hyperparameter Tuning

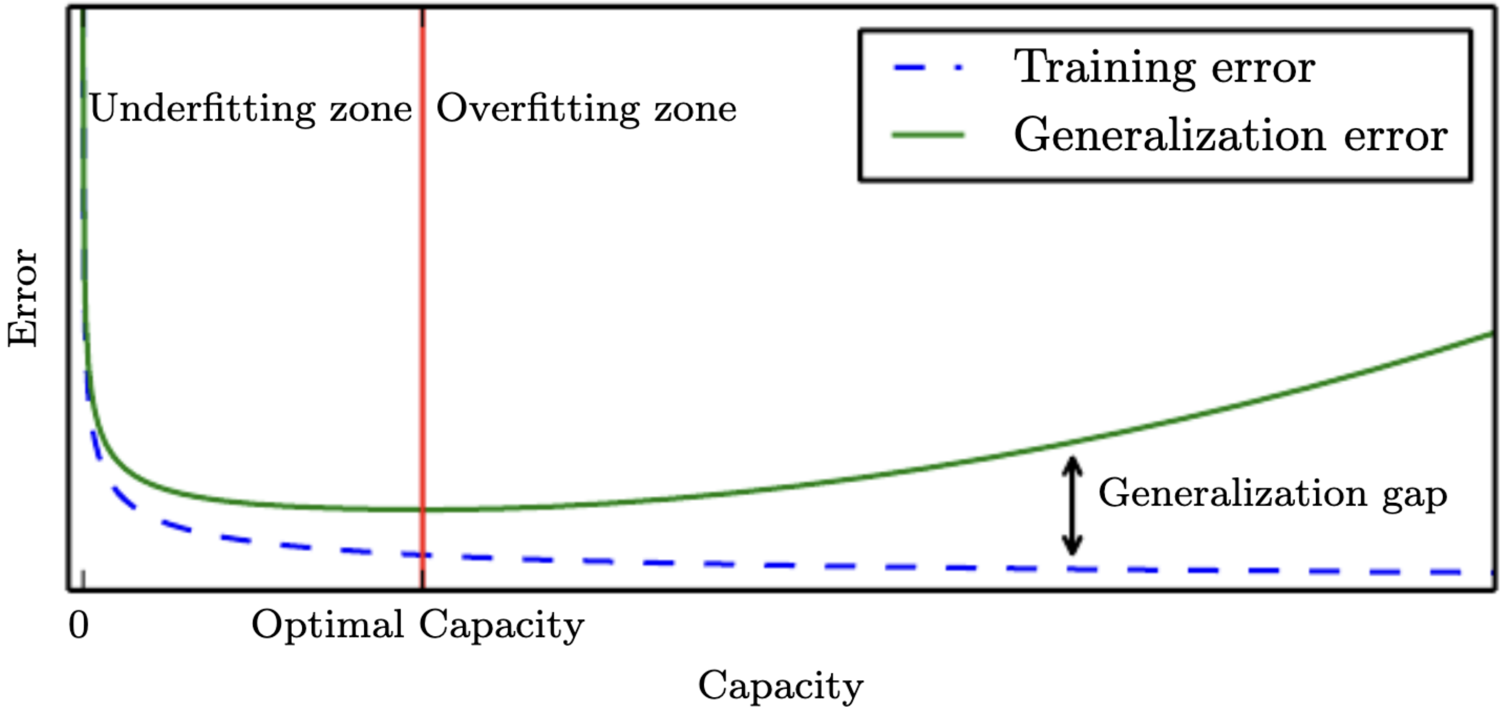
- Dataset divided into three splits; e.g., using a 60%/20%/20% train/val/test split



- **Hyperparameter selection**: test on validation set to identify best hyperparameters
- **Final model**: train on training AND validation splits with best hyperparameters

https://github.com/amueller/introduction_to_ml_with_python/blob/master/05-model-evaluation-and-improvement.ipynb

Hyperparameter Tuning for Optimal Capacity



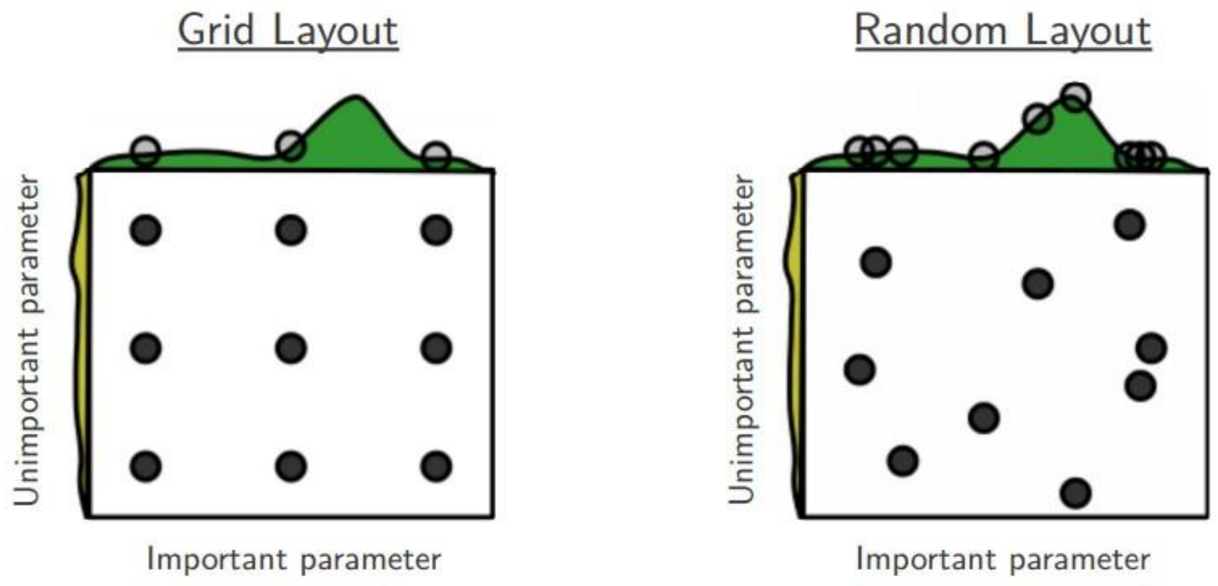
Tweaking a hyperparameter can shift model between underfitting and overfitting; e.g., impact of increasing vs decreasing…

- # of hidden layers and hidden units (model architecture)?
- weight decay coefficient (regularization strength)?
- learning rate (model training)?

Ian Goodfellow, Yoshua Bengio, and Aaron Courville; Deep Learning, 2016

Hyperparameter Tuning Approaches

- Automatic: extensive and so computationally costly; e.g.,



As exemplified, grid search is inferior when selecting multiple hyperparameters since **fewer values are tested** for each (in this example, two hyperparameters); random search better supports identifying impactful hyperparameters

- Manual: leverages “art” of knowing how to effectively train
  - understanding relationship between hyperparameters, training error, generalization error, and computational constraints (e.g., GPUs, memory)

Bergstra et al; Random Search for Hyper-Parameter Optimization, 2012

Discussion: What Do You Think Is Happening and What Might Be a Fix?

Error on training set is larger than your target error rate

Today’s Topics

- Model capacity: how it affects learning
- Regularization: learning methods for improving model generalization
- Hyperparameter selection: tuning to improve model performance
- Programming tutorial

Today’s Topics

- Model capacity: how it affects learning
- Regularization: learning methods for improving model generalization
- Hyperparameter selection: tuning to improve model performance
- Programming tutorial


The End