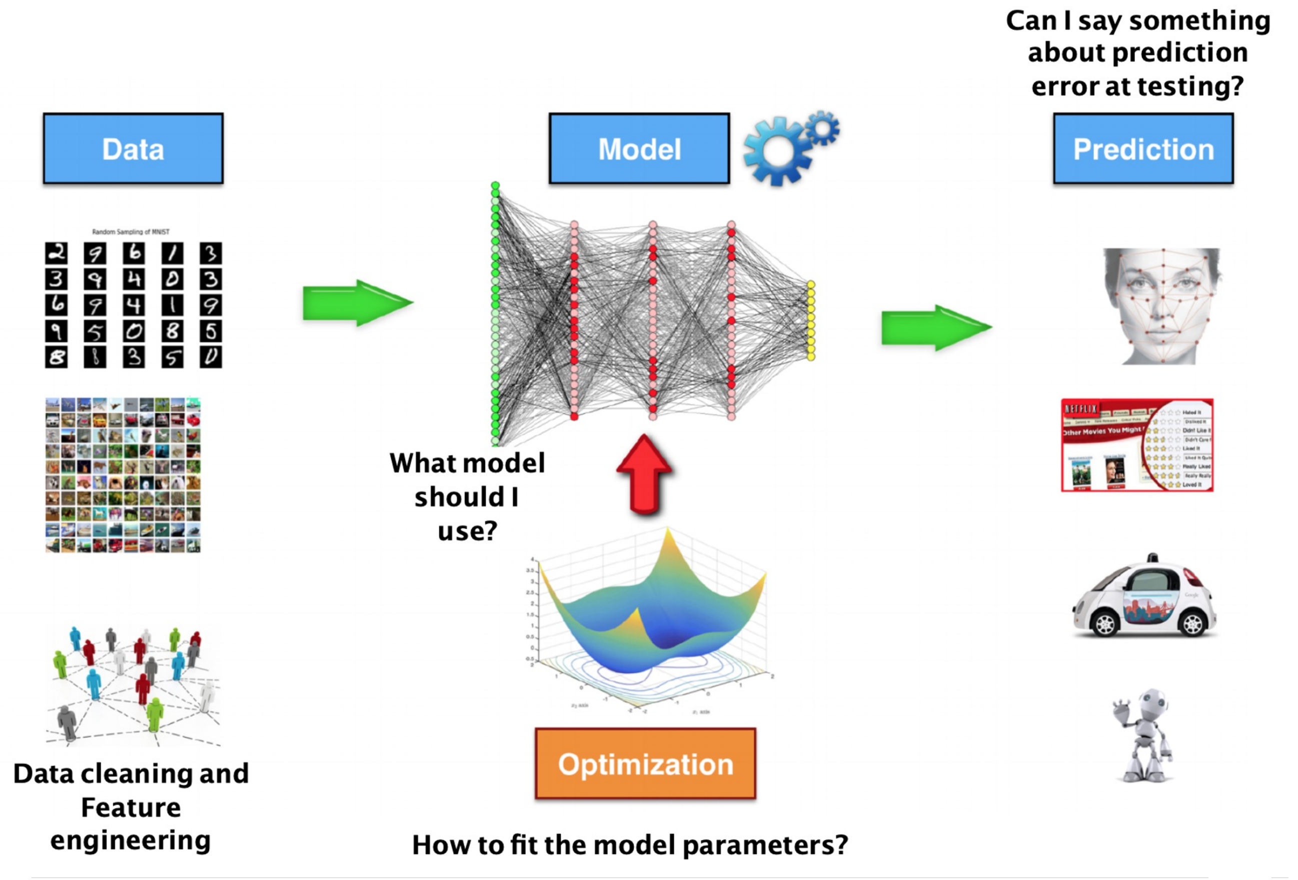
1. Intro: Machine Learning Basics

Slides adapted from Rui Zhang

The big picture



Goal of machine learning

- What we care about is the **TEST** error. Algorithms that can **generalize from training data to unseen test data**!

Midterm analogy:
- The training error is the practice midterm.
- The test error is the actual midterm.

Goal: do well on actual midterm, not the practice one.

- Memorization vs Learning:
  - We can do well on training data by memorizing it.
  - You've only learned if you can do well on predicting new situations (test error).

Machine learning all is about generalizing (performance on unseen test data)

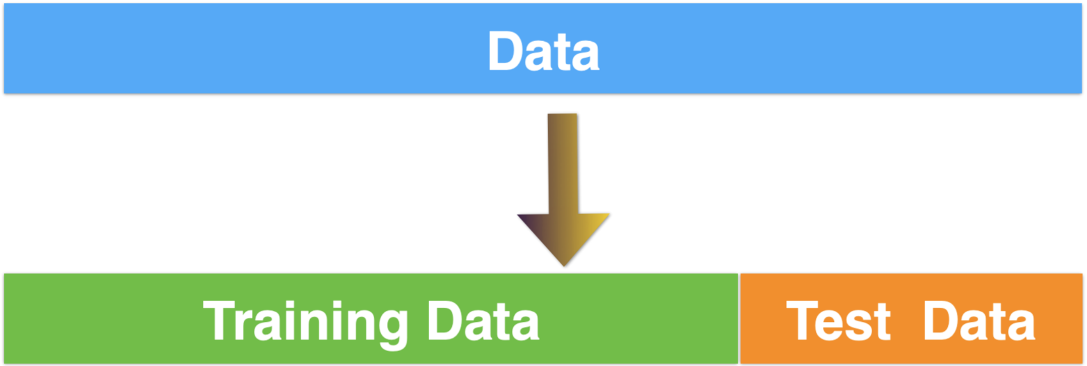
I.I.D assumption

- Training/test data is **independent and identically distributed (i.i.d)** if:
  - All objects come from the same distribution (identically distributed).
  - The object are sampled independently (order doesn't matter).
  - We do NOT need to know the underlying distribution as long as the samples are sampled i.i.d.

- Examples in terms of cards:
  - Pick a card, put it back in the deck, re-shuffle, repeat.
  - Pick a card, put it back in the deck, repeat.
  - Pick a card, don't put it back, re-shuffle, repeat

What if I do not have test data

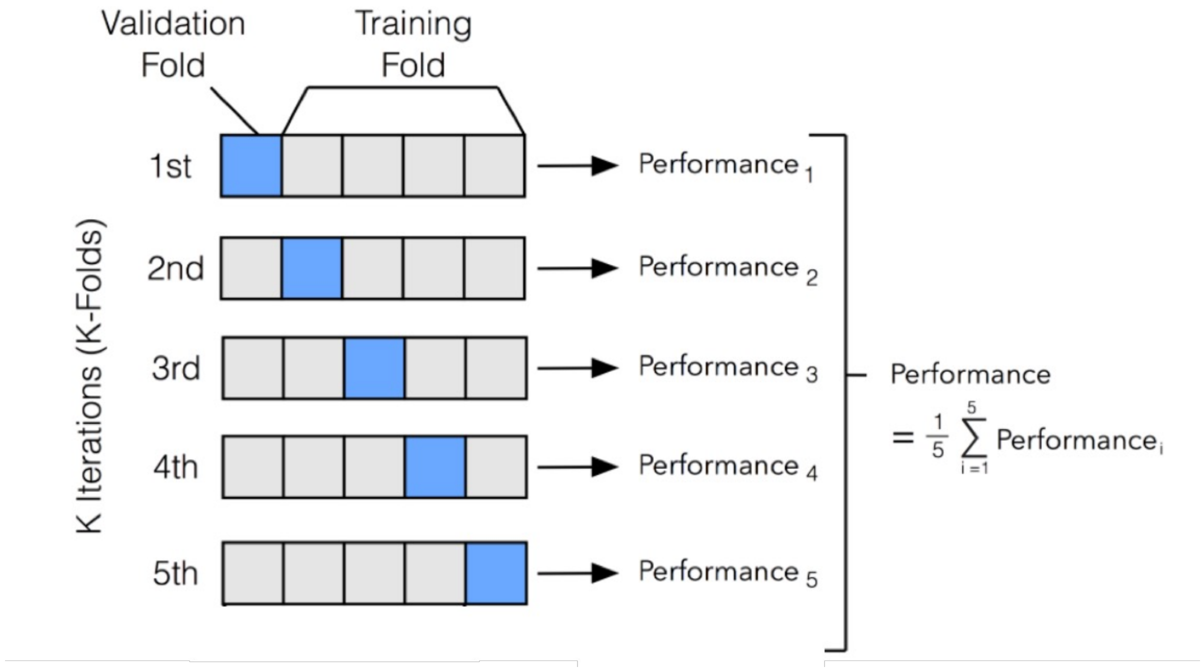
- At training time, I have no idea how test data going to look like?
- Let's split the training data into two parts (80%+20% or 70%+30%):
  - first part used only for training
  - second part used only for evaluation



- Basically trying to simulate the future test examples during training to evaluate model.
- The splitting should be random!

What if I do not have test data

- Cross validation: It is also used to flag problems like overfitting or selection bias



Golden rules

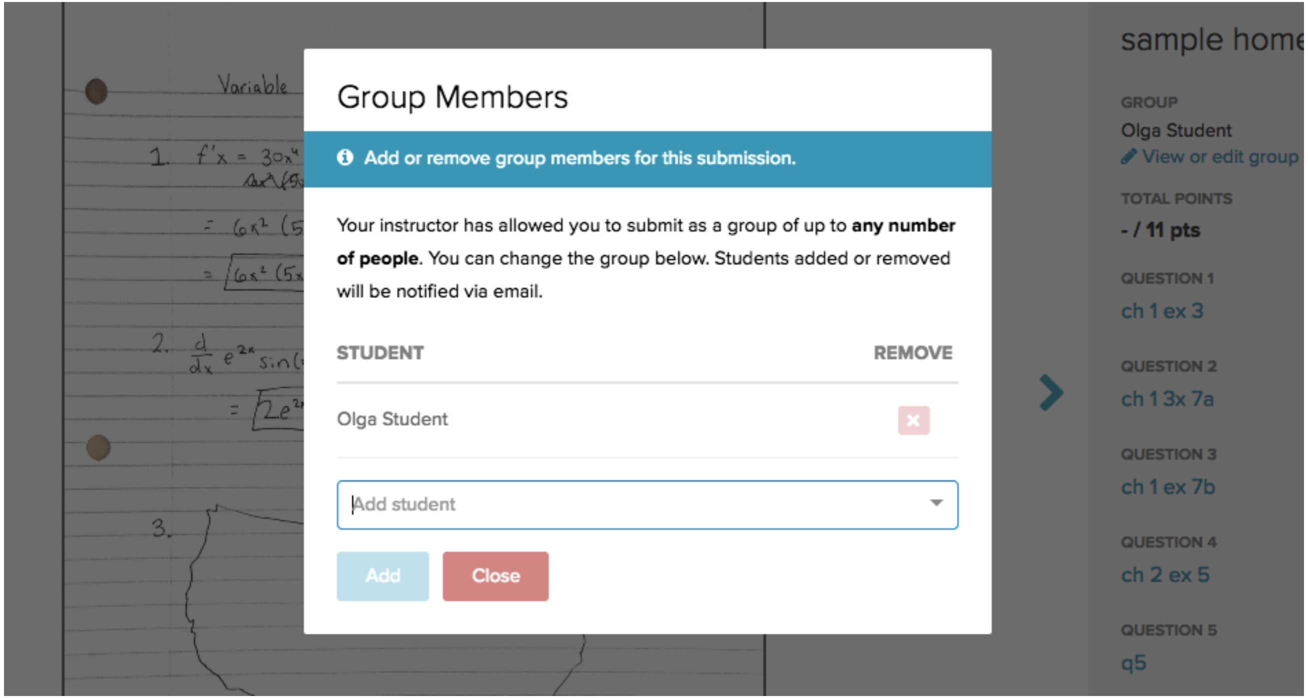
- Even though what we care about is test error:

THE TEST DATA CANNOT INFLUENCE THE **TRAINING** PHASE IN ANY WAY. Otherwise the model is cheating

- We're measuring test error to see how well we do on new data:
    - If used during training, doesn't measure this.
    - You can start to overfit if you use it during training.
    - Midterm analogy: you are cheating on the test.

Logistics

- Gradescope submission
  - Assignments, Presentation slides, final project report



Logistics

- https://help.gradescope.com/article/ccbpppziu9-student-submit-work

Model

Statisticians and data scientists capture the uncertainty and randomness of data-generating processes with mathematical functions that express the shape and structure of the data itself.

"All models are wrong, but some are useful"

Question: How do you have any clue whatsoever what functional form the data should take?

Answer:

"All models are wrong, but some are useful"

Question: How do you have any clue what functional form the data should take?

Answer: We do not know! But we can create model spaces and hopefully they will approximate well the actual functional form of the real data

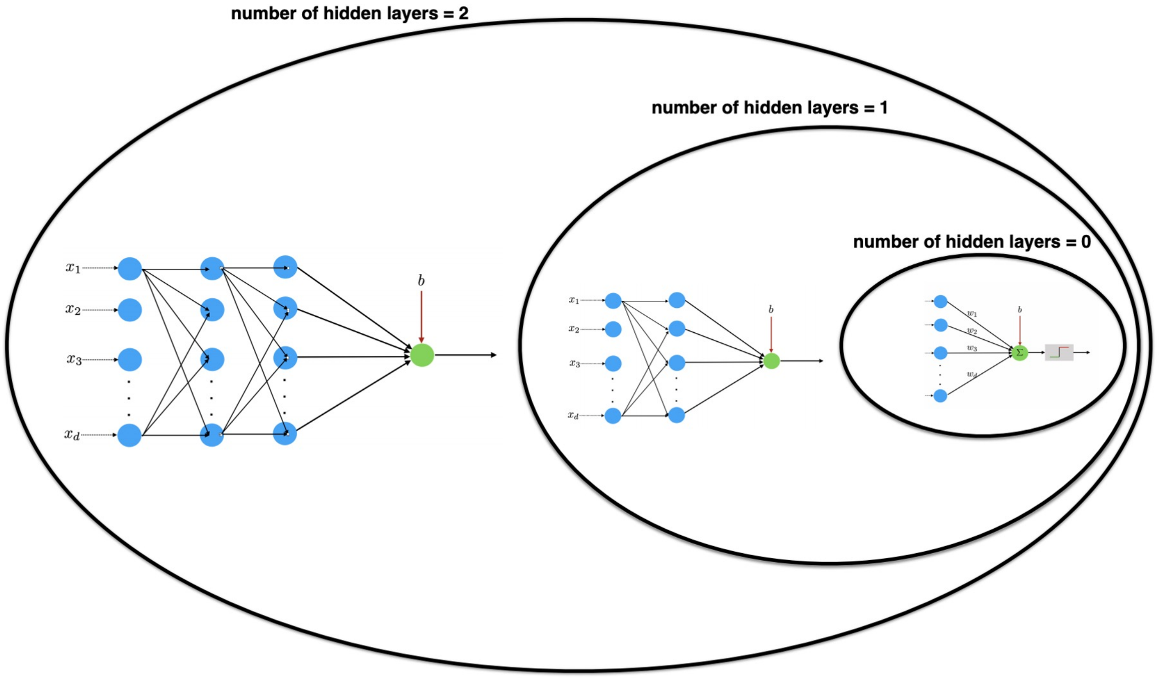
Fitting a model

- I cleaned my data, decided on a model, what is next?
  - We need to find the best model (parameters) among all models!

- Learning is almost like optimization (Optimization algorithms are our best friends here!)
  - Choose a model
  - Choose an objective function (loss function, error function, etc.)
  - Find parameters that maximize/minimize objective function over training data

Model Complexity: Neural Networks

The richness (complexity) of a model is the function space it can represent!

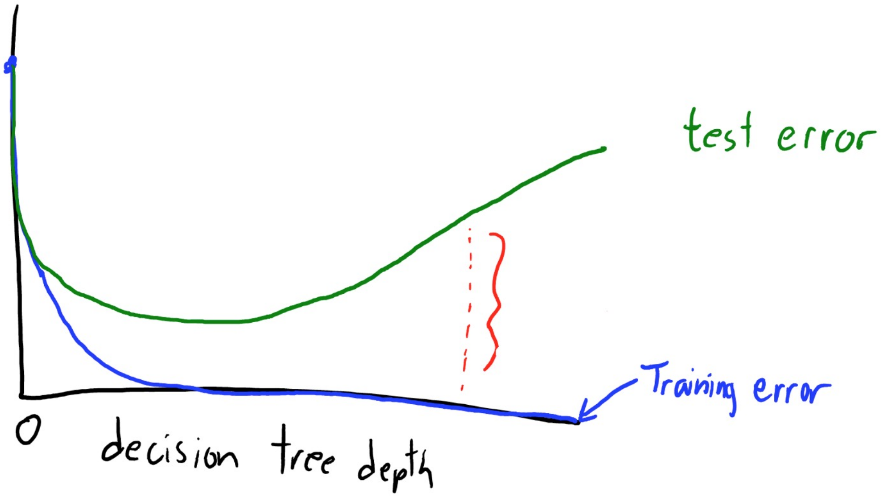


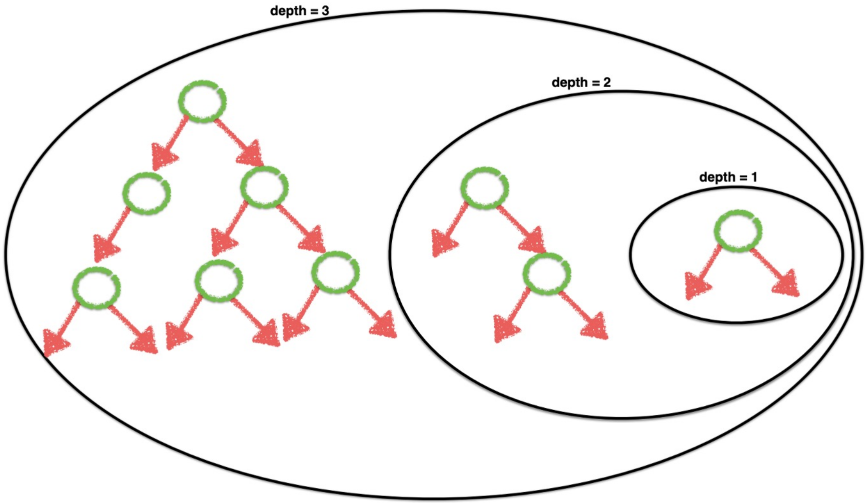
By increasing the number of hidden layers, we can learn more complex decision rules!

The Fundamental Trade-Off

Training error vs. test error for choosing depth in decision trees:

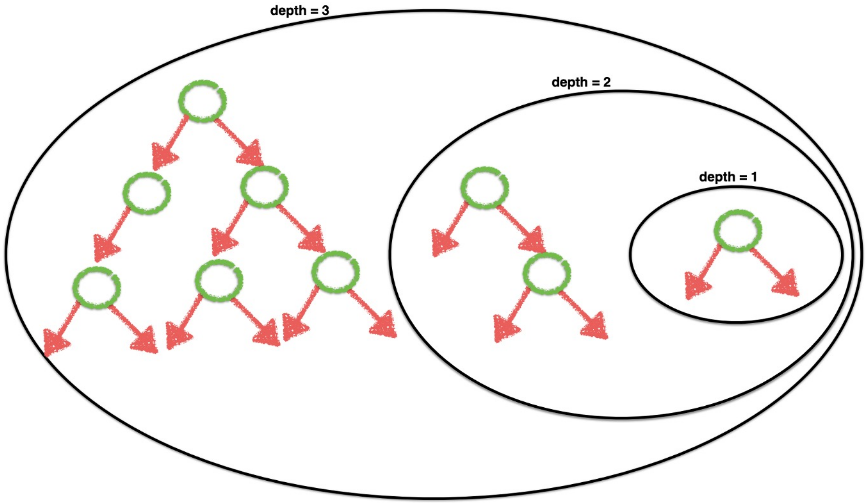
- Training error gets better (decreases) with depth. Why? Model gets richer and richer and can approximate more complex functions!
- Test error initially goes down, but eventually increases.





Overfitting vs Underfitting

- Zero training error is not necessarily a good thing.
- There is the danger of **Overfitting**
  - When the parameters of a model are exactly tuned to a particular set of training data, it fails to predict future unseen observations reliably.
  - Happens for example when we learn with a very very complex model so the model fits (learns) the noise.
- **Underfitting** is the opposite: learning with a very very naive and simple model that does not fit data at all.



Bias-Variance Tradeoff

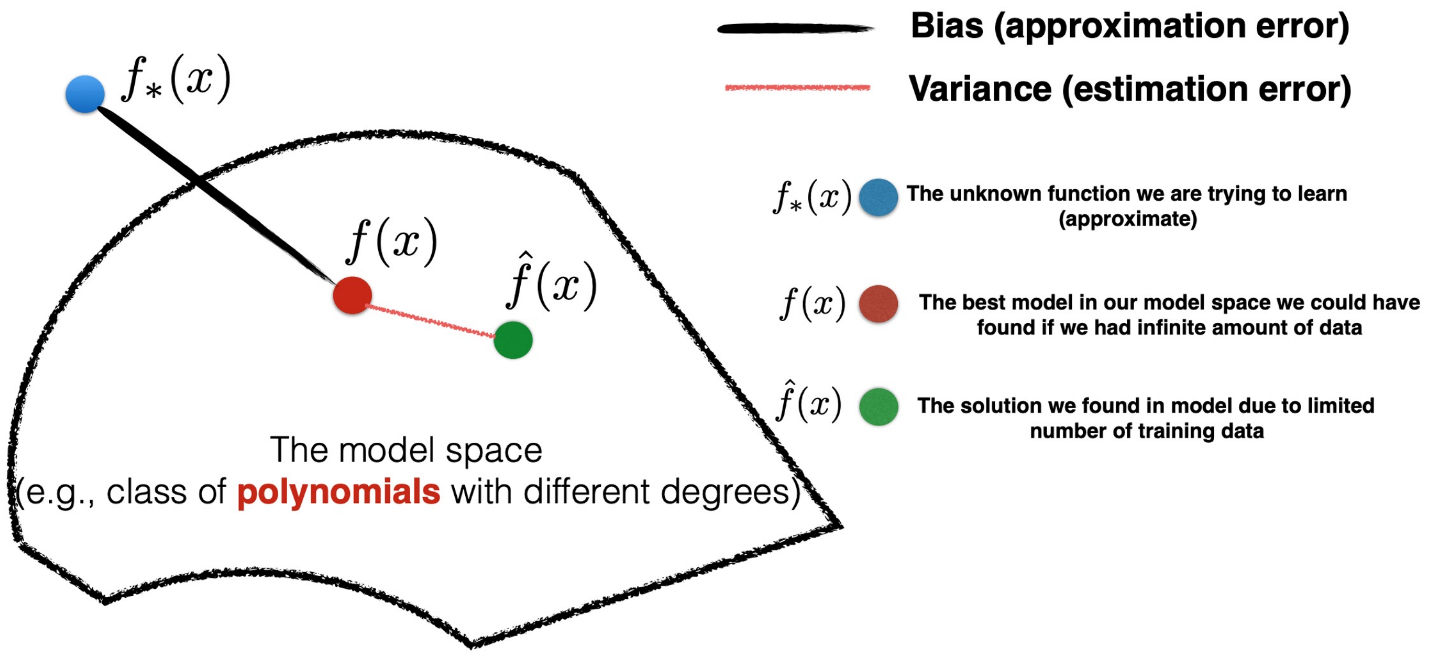
$$\underbrace{E_{\mathbf{x},y,D}\left[\left(h_D(\mathbf{x}) - y\right)^2\right]}_{\text{Expected Test Error}} = \underbrace{E_{\mathbf{x},D}\left[\left(h_D(\mathbf{x}) - \bar{h}(\mathbf{x})\right)^2\right]}_{\text{Variance}} + \underbrace{E_{\mathbf{x},y}\left[\left(\bar{y}(\mathbf{x}) - y\right)^2\right]}_{\text{Noise}} + \underbrace{E_{\mathbf{x}}\left[\left(\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x})\right)^2\right]}_{\text{Bias}^2}$$

Variance: Captures how much your classifier changes if you train on a different training set. How "over-specialized" is your classifier to a particular training set (overfitting)? If we have the best possible model for our training data, how far off are we from the average classifier?

Bias: What is the inherent error that you obtain from your classifier even with infinite training data? This is due to your classifier being "biased" to a particular kind of solution (e.g. linear classifier). In other words, bias is inherent to your model.

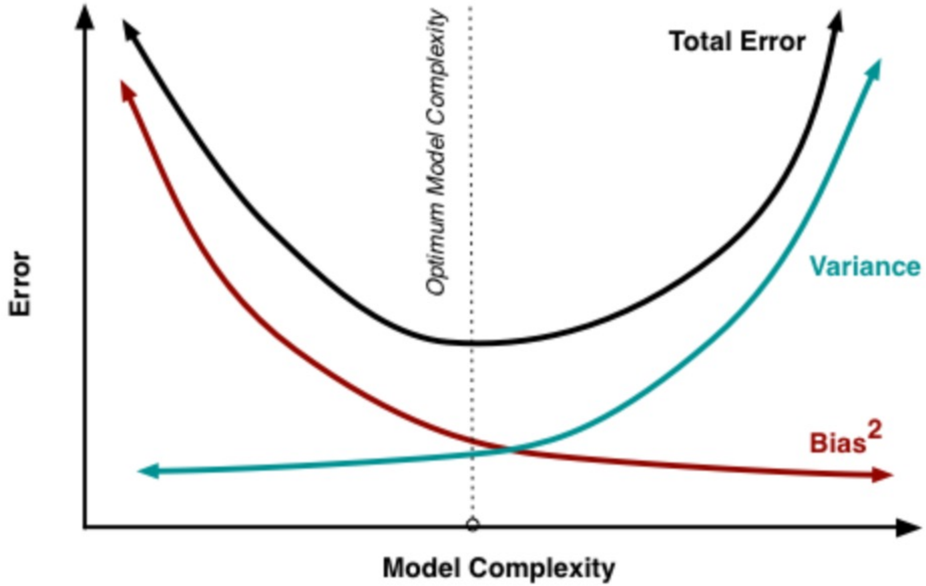
Noise: How big is the data-intrinsic noise? This error measures ambiguity due to your data distribution and feature representation. You can never beat this, it is an aspect of the data.

Bias-Variance Tradeoff



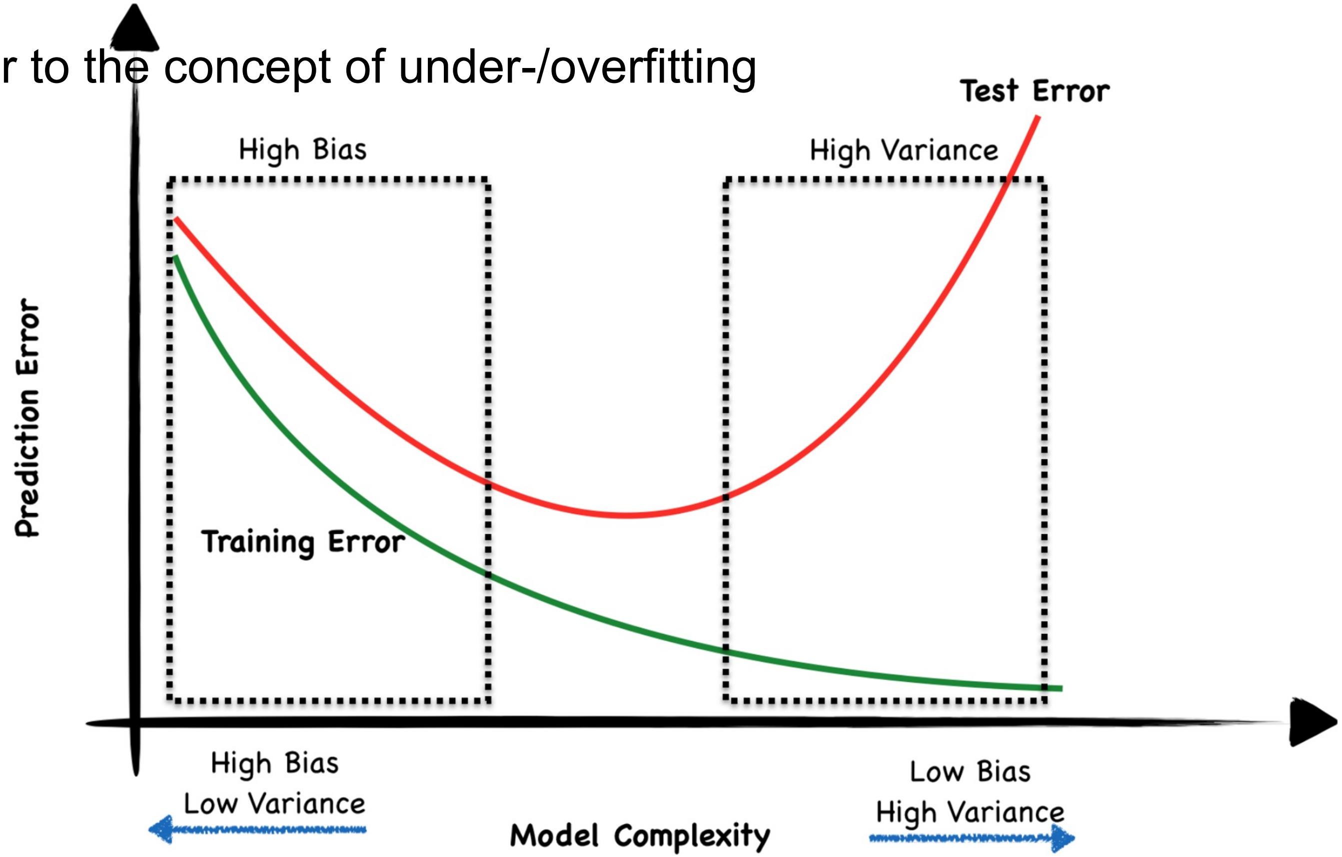
- **Bias** is due to our assumption about model space.
- **Variance** is due to finite number of training data!
- Complex models have low bias and *high variance*
- Simple models have *high bias* and low variance

Model complexity and generalization



Model complexity and generalization

- Similar to the concept of under-/overfitting



Underfitting / Overfitting

- Model Regularization

Model Regularization

- Originally, we find parameters that minimizes training error (the discrepancy between the predictions of our model and actual labels)

$$\underset{w_0, w_1, \ldots, w_9}{\textbf{minimum}} \quad \textbf{training error}$$

- Now, we find parameters that minimizes training error and penalizes parameters:

$$\underset{w_0, w_1, \ldots, w_9}{\textbf{minimum}} \quad \textbf{training error} + \lambda \left(w_0^2 + w_1^2 + \ldots + w_9^2\right)$$

- $\lambda \geq 0$ is the regularization parameter, and controls how aggressive we are in penalizing the model parameters (e.g., $\lambda = 0$ means no penalization)!

Takeaways

1. We can improve performance by restricting number of parameters (simpler models).

2. We can improve performance by getting more data.

3. We can improve performance by **regularization**:

- **Aggressive regularization** results in simpler models, thus increasing bias and decreasing variance
- **Passive regularization** results in more complex models thus decreasing bias while increasing variance.

Hyperparameters

Note that regularization parameter λ is not a part of model parameters, i.e,

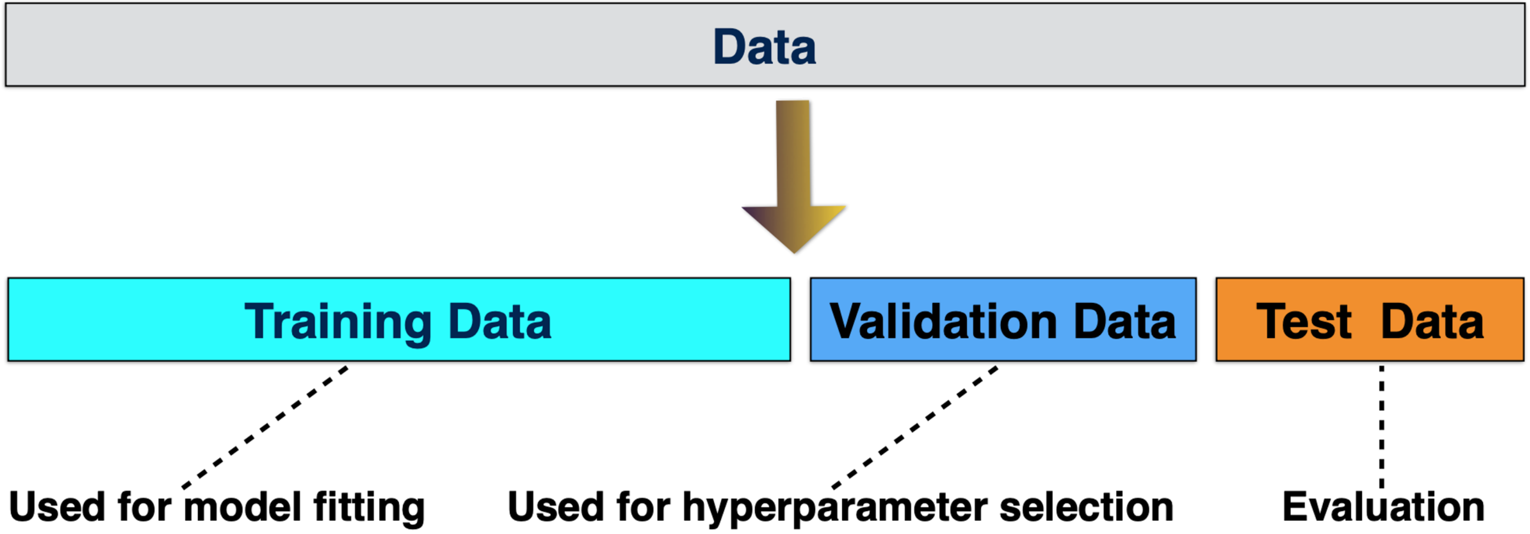
$$w_0, \ldots, w_9$$

To distinguish it from **model parameters** we call it a **hyper-parameter**

How to find the best value for the regularization parameter that results in minimum test error (better generalization)?

The answer to this question is part of our model selection task!

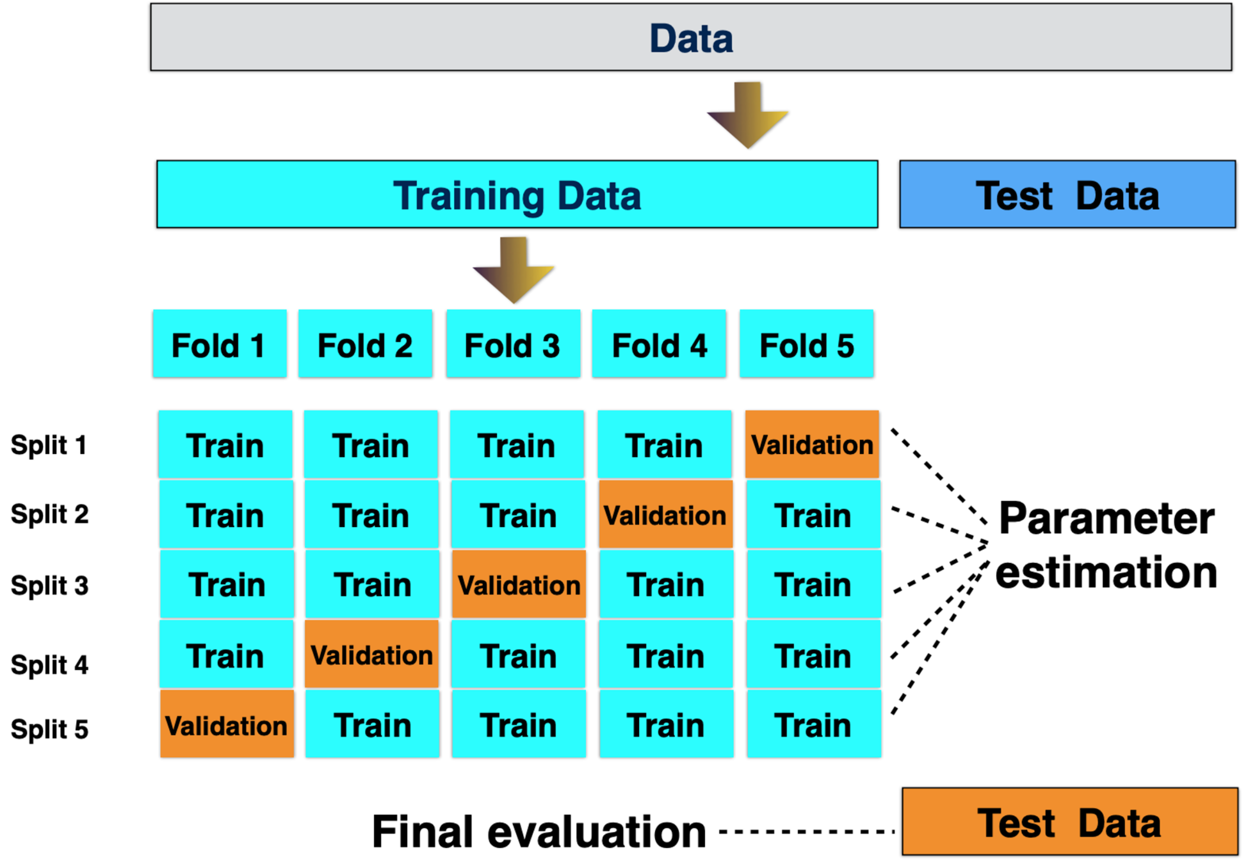
Threefold split



Pro: fast, simple

Con: high variance, bad use of data

K-fold cross validation



All together