Validation

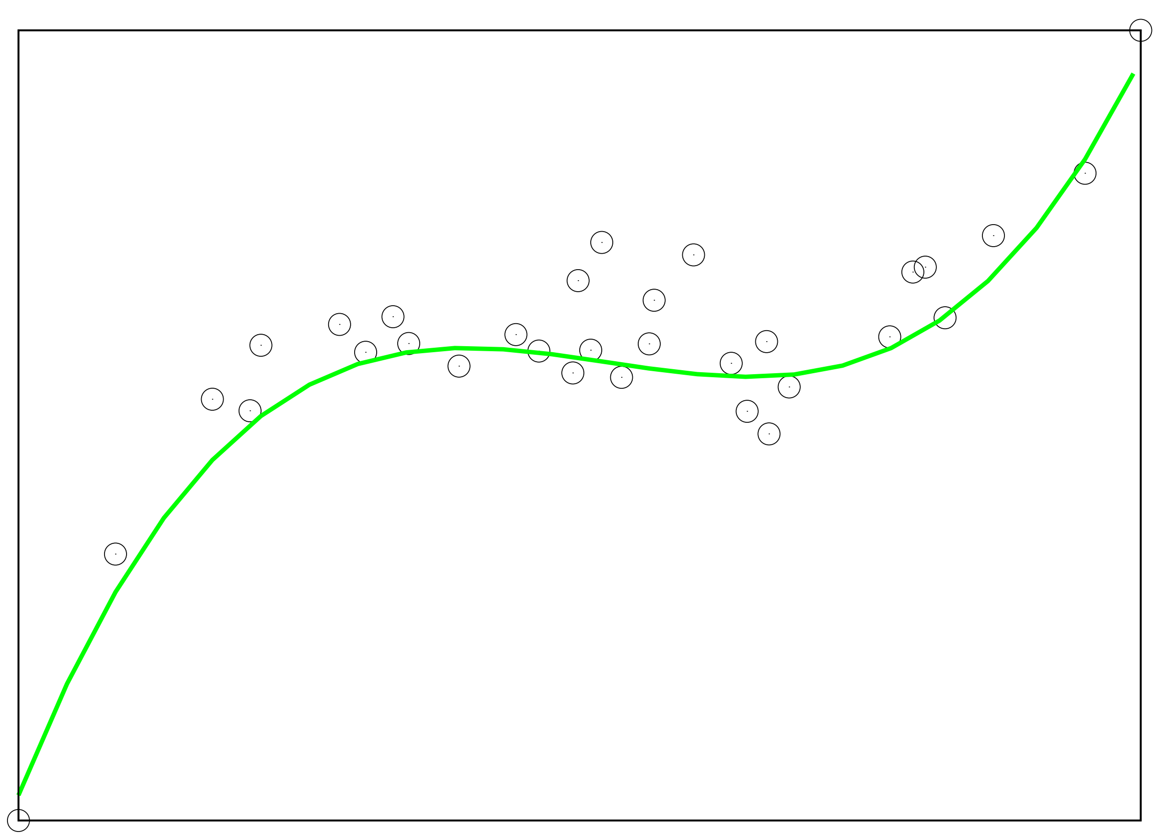
Sewoong Oh

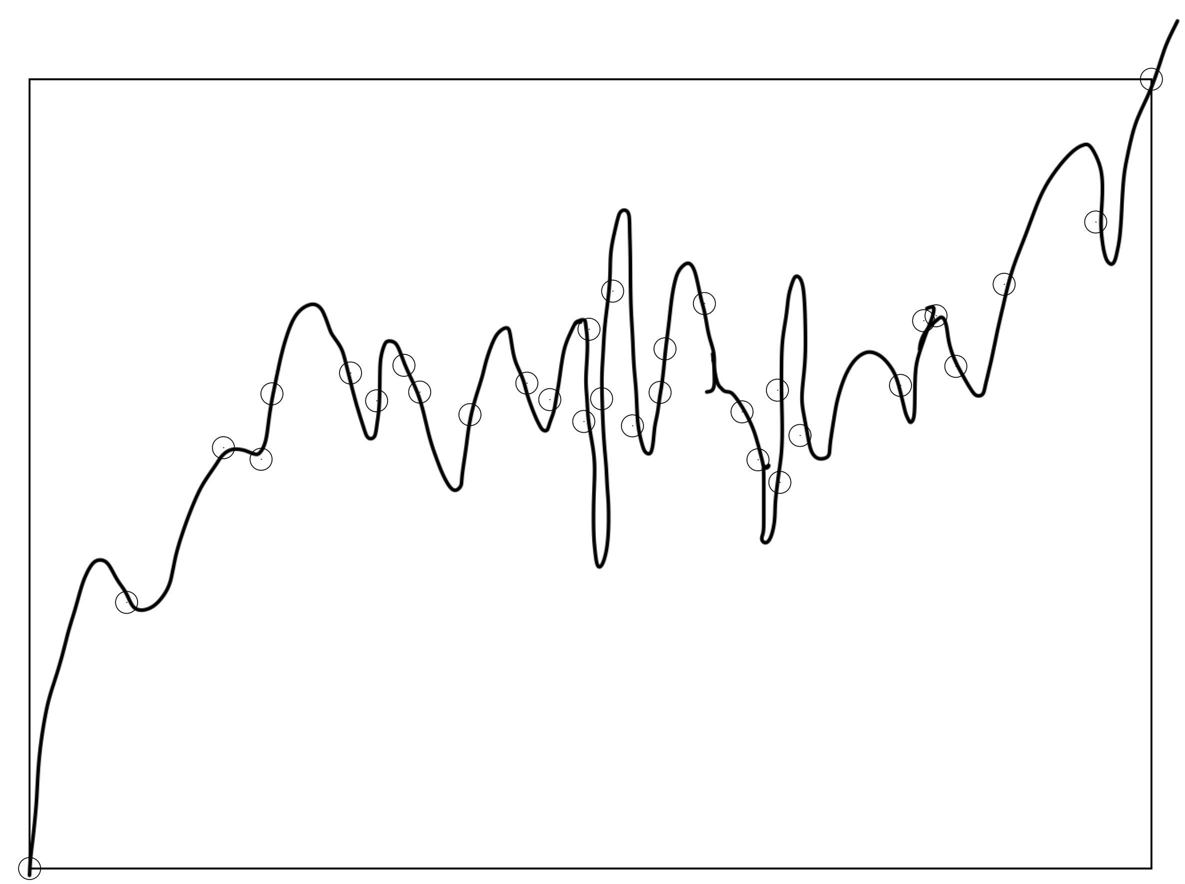
CSE/STAT 416
University of Washington

Generalization:
how do we validate which model is better?

Generalization

- we say a predictor **generalizes** if it performs well on unseen data
- formal mathematical definition involves probabilistic assumptions
- first, we study practical methods for assessing generalization

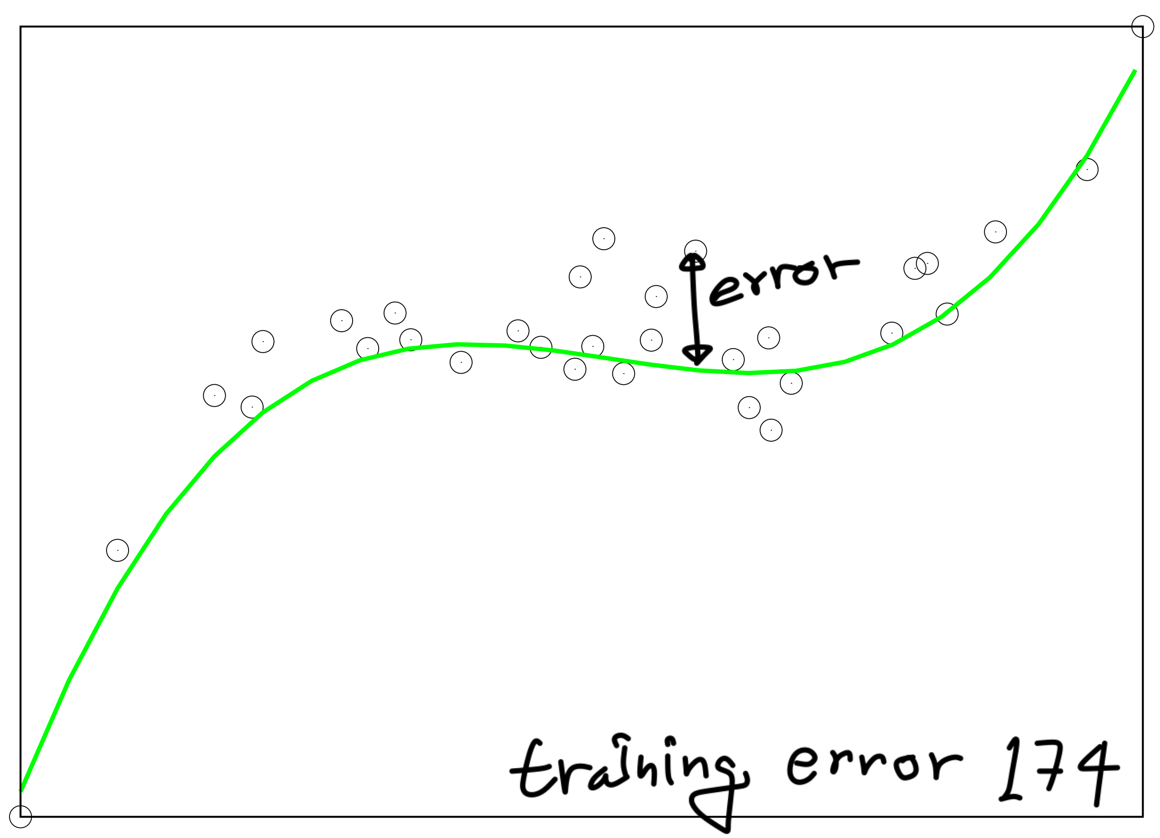


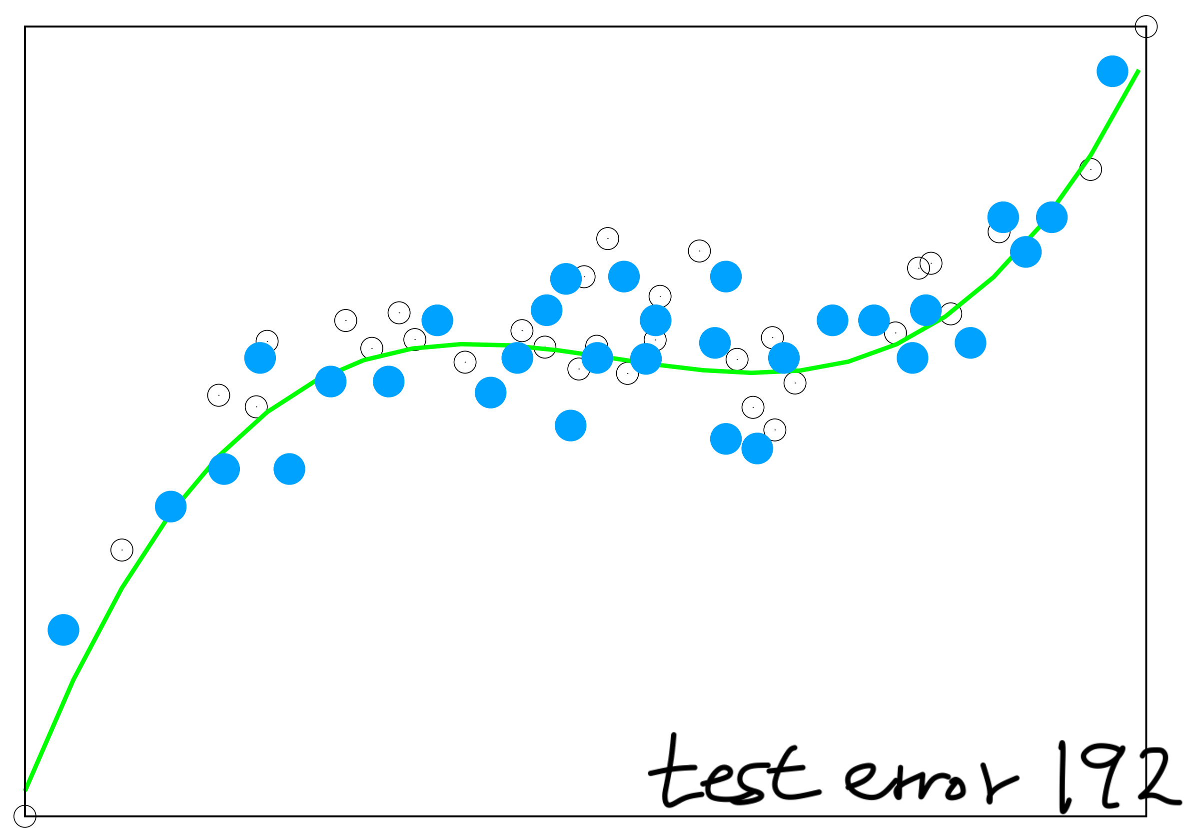


$w_0 + w_1 x + \cdots + w_p x^p$

In-sample and out-of-sample data

- the data used to construct a predictor is **training data** or **in-sample data**
- we want the predictor to work on **out-of-sample data**
- we say a predictor **fails to generalize** if it does not perform well on out-of-sample data





- **train** a cubic predictor on 32 (in-sample) black circles: MSE 174
- **predict** y for 30 (out-of-sample) blue points: MSE 192
- conclude this predictor generalizes: in-sample MSE $\approx$ out-of-sample MSE

Validation

- a way to mimic how the predictor performs on unseen data
- key idea: divide the data into two set for **training** and **testing**

- **training set** used to construct (“train”) the predictor
- **test set** or **validation set** used to evaluate the predictor

- based on the assumption that test set is similar to unseen data

$$\left.\begin{array}{c} 80/20 \\ 90/10 \\ \text{Train / Test} \end{array}\right\} \text{random split}$$

$$\left.\begin{array}{c} (x_1, y_1) \\ \vdots \\ (x_{80}, y_{80}) \end{array}\right\} S_{train}$$

$$\left.\begin{array}{c} (x_{81}, y_{81}) \\ \vdots \\ (x_{100}, y_{100}) \end{array}\right\} S_{test}$$

Validation

- we use **training error** for optimization (or finding the model)

$$\mathrm{MSE}_{\mathrm{train}} = \frac{1}{|S_{\mathrm{train}}|} \sum_{i \in S_{\mathrm{train}}} (f(x_i) - y_i)^2$$

- we use **test error** for validation

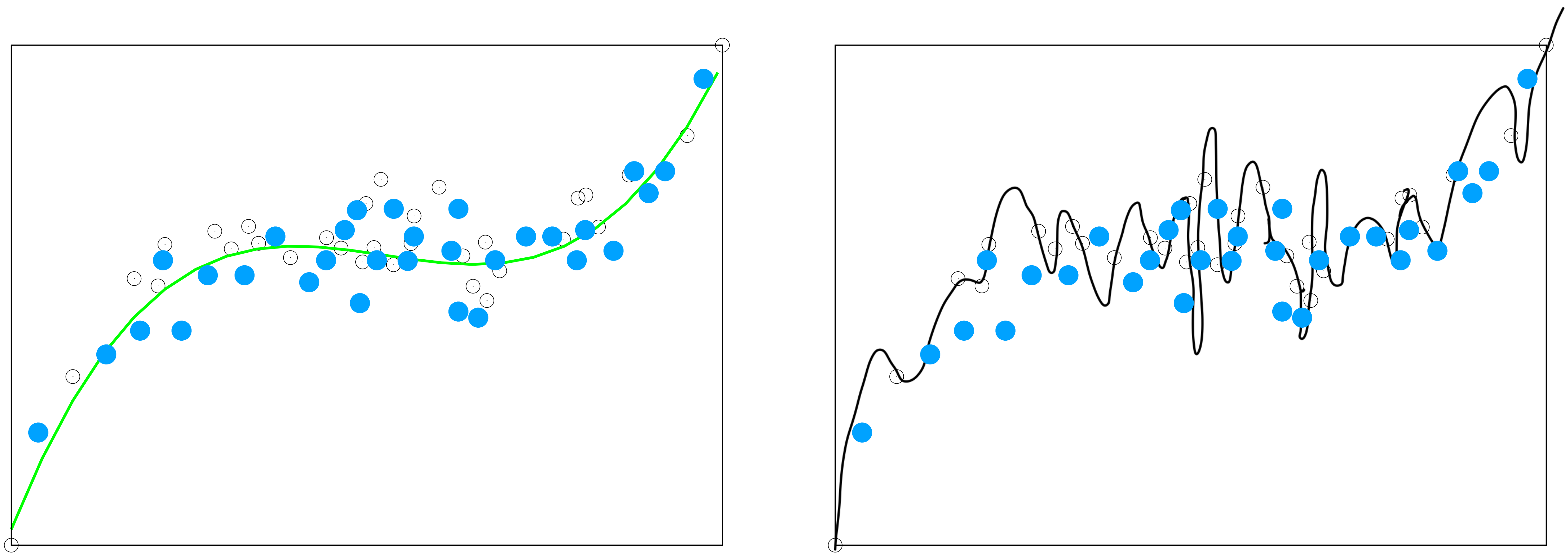
$$\mathrm{MSE}_{\mathrm{test}} = \frac{1}{|S_{\mathrm{test}}|} \sum_{i \in S_{\mathrm{test}}} (f(x_i) - y_i)^2$$

- selecting train/test sets should be **random**
(80/20 or 90/10 are common)

- we say a model or predictor is **overfit** if

$$\mathrm{MSE}_{\mathrm{test}} \gg \mathrm{MSE}_{\mathrm{train}}$$

	small training error	**large training error**
small test error	generalizes perform well	possible, but lucky
large test error	fails to generalize	generalizes perform bad



$w_0 + w_1 x + \cdots + w_p x^p$

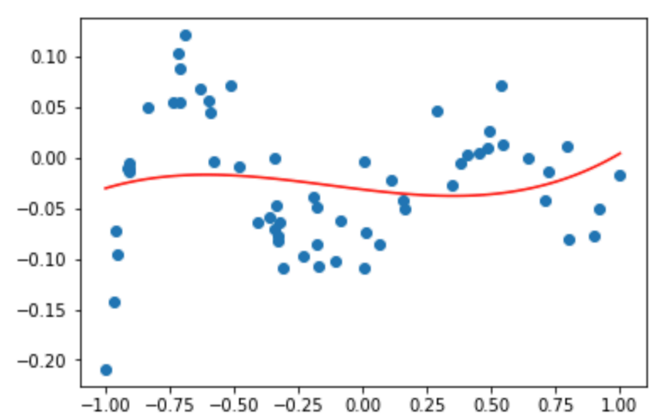
- between two models, the one with smaller test error should be chosen

Overfitting

- a model that fits the training data well but performs poorly on test data suffers from **overfitting**
- overfitting happens if we use a model with high **model complexity**
- for example, for linear regression with polynomial features

$$\hat{y} = f(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_p x^p$$

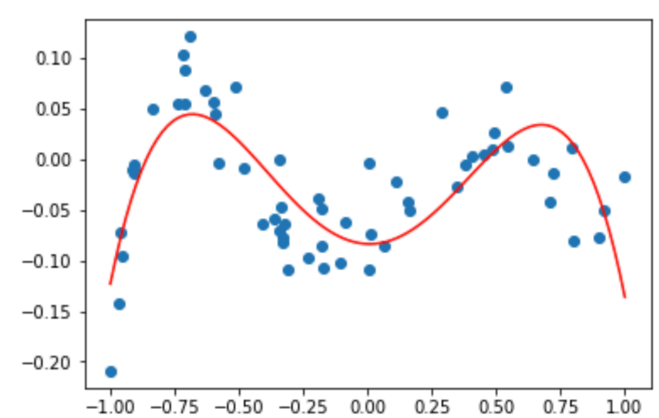
- N = 60 data points, and $p \in \{3, 4, 5, 20\}$



0.004011490884146126

0.003831010290504173

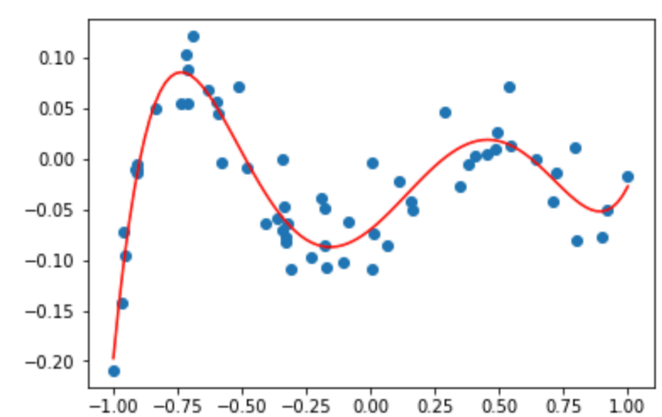
degree 3



0.0019177229761741974

0.002200492720447942

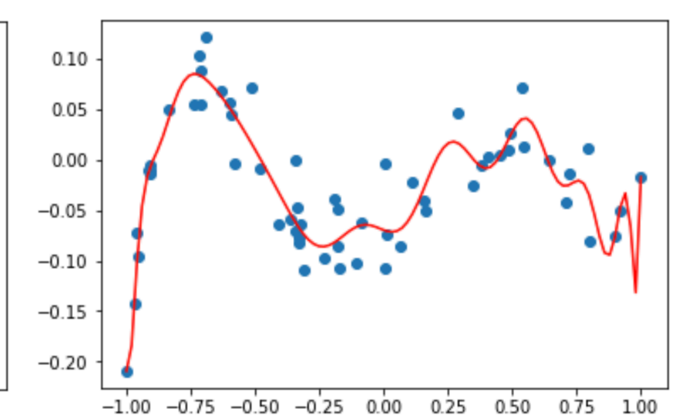
degree 4



0.0007426853089970962

0.0010239915404334238

degree 5

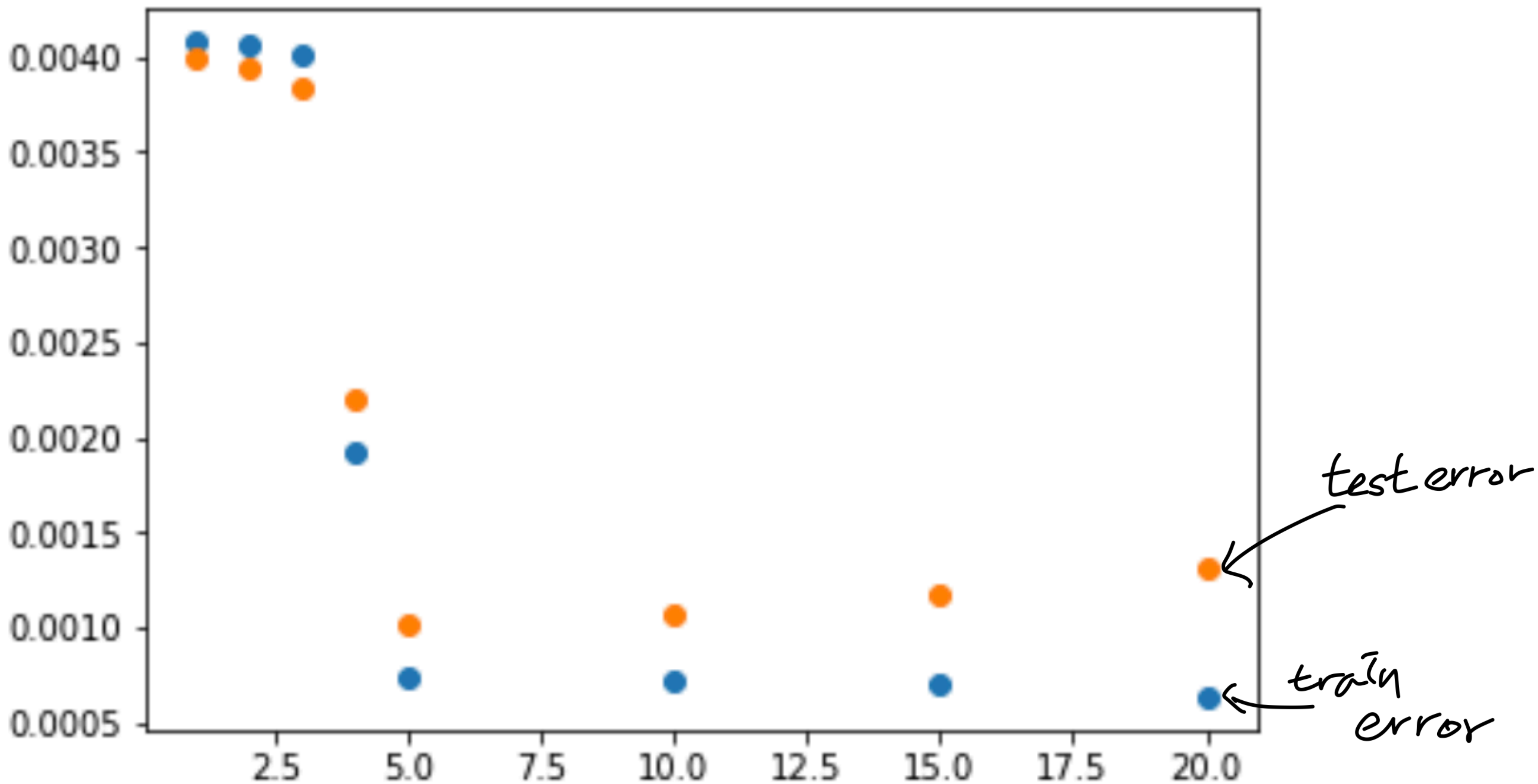


0.0006350545819906561 $\mathrm{MSE}_{\mathrm{train}}$

0.0013118370515986446 $\mathrm{MSE}_{\mathrm{test}}$

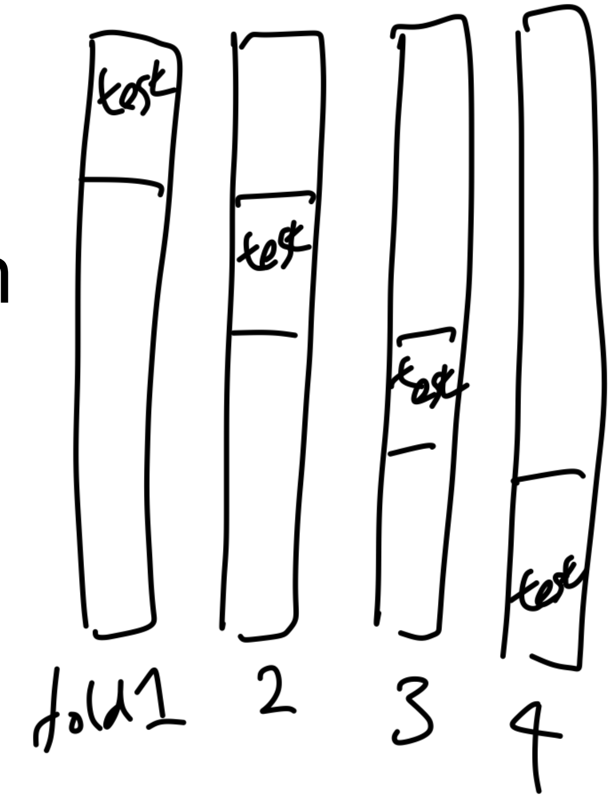
degree 20

How does one choose which model to use?



- first use 60 data points to train and 60 data points to test
- then choose degree 5 as per the above test error
- now re-train on all 120 data points with degree 5 polynomial model

Cross validation

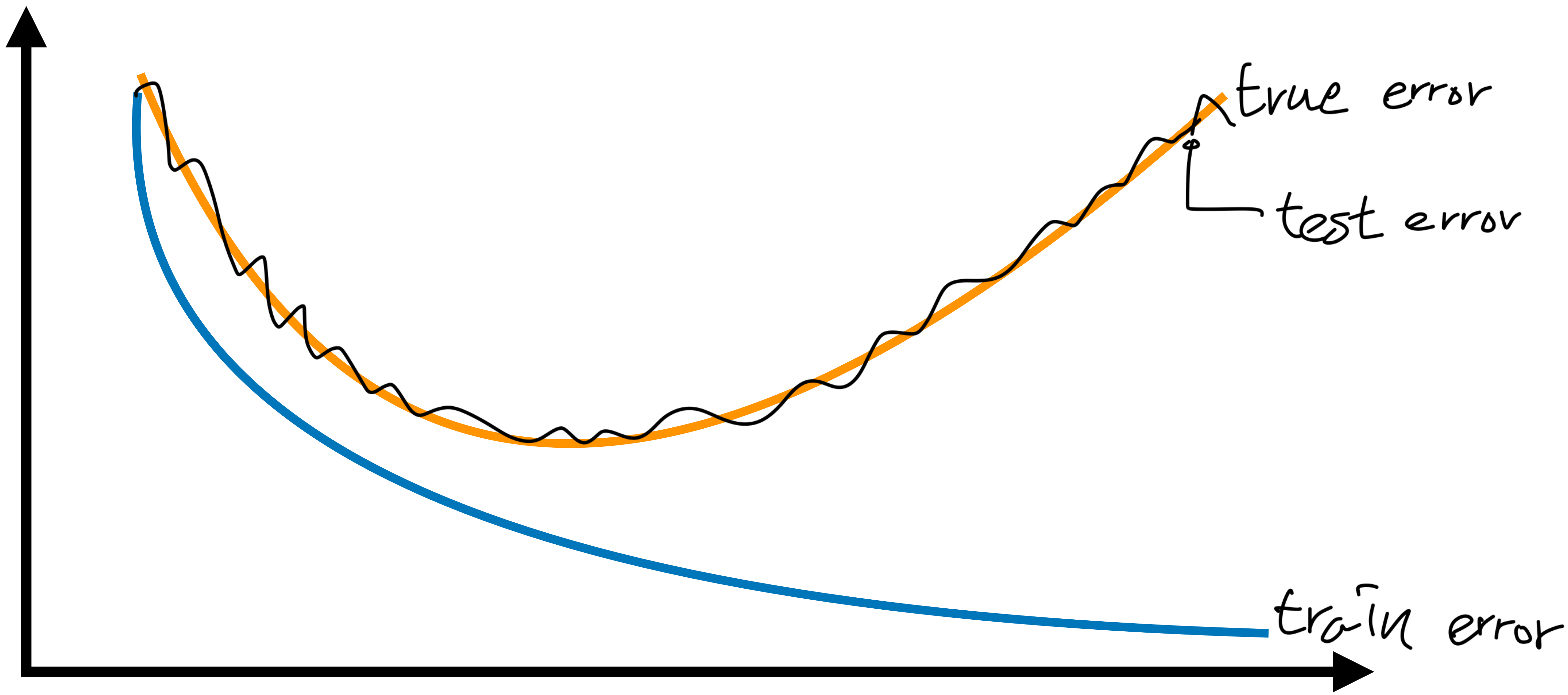


- systematic method for out-of-sample validation
  - divide the data into *k* **folds**
  - for each i, fit predictor on all data but fold i
  - compute test error on fold i
  - average the test error across the folds

- gives some idea of the variability (or variance) of the test error

- we can estimate **stability** of the ML pipeline, by comparing the model parameters found in each fold

- test error gives an approximation of the (unknown) true error
- train error goes down monotonically w.r.t. model complexity

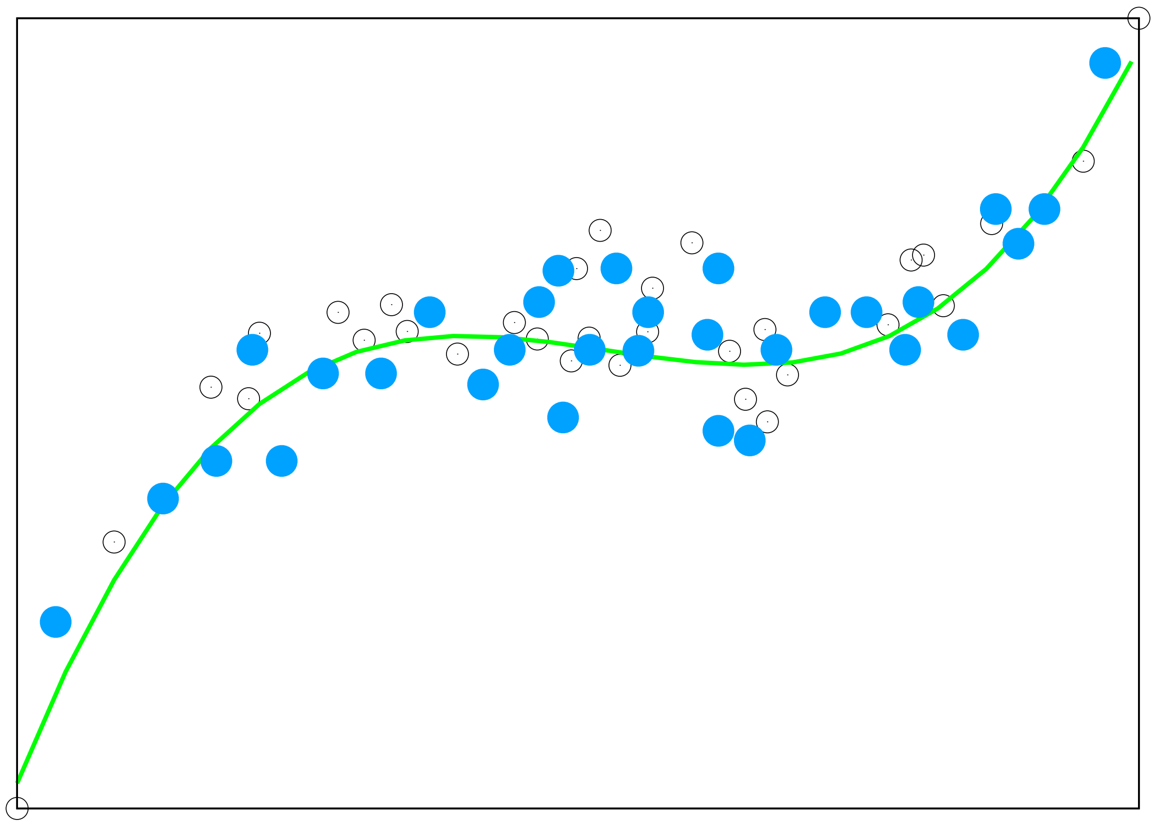


Why does true error go down and up?

- three sources of error: **noise**, **bias**, and **variance**

$$\mathrm{MSE}_{\mathrm{true}} \;=\; \mathrm{MSE}_{\mathrm{noise}} + \mathrm{MSE}_{\mathrm{bias}} + \mathrm{MSE}_{\mathrm{variance}}$$

- error from **noise** in the data cannot be reduced

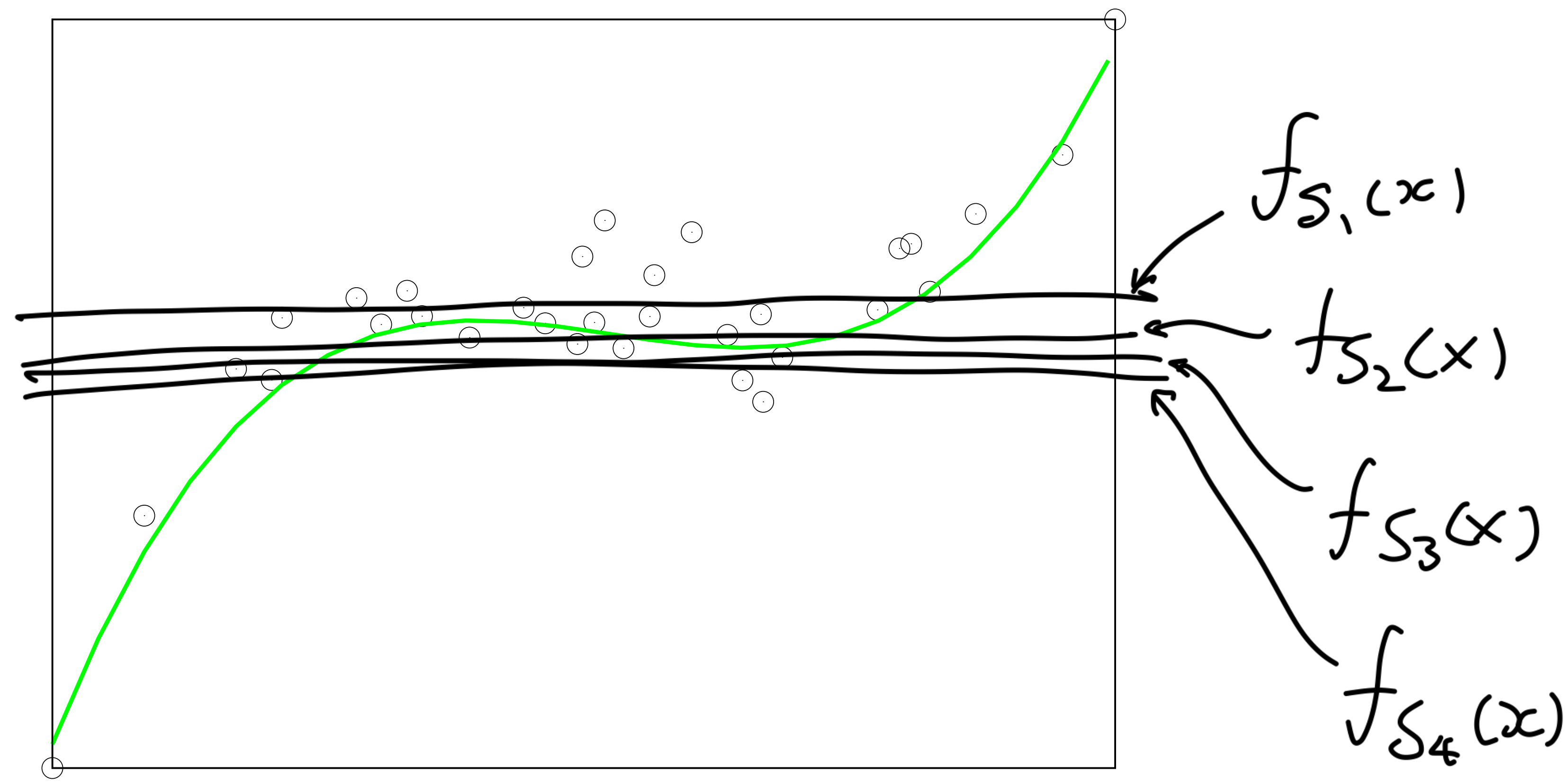


$$y \;=\; f_0(x) + \varepsilon \quad , \hat{y} = f_0(x)$$

$$\mathrm{MSE}_{\mathrm{noise}} = \mathbb{E}[\varepsilon^2] \leftarrow y - \hat{y} = \varepsilon$$

Low complexity models

- suppose we train a constant function, many times each with *N* samples from $y = f_0(x) + \varepsilon$



$$\text{Bias} = \mathbb{E}[|f_0(x) - \overline{f_S(x)}|]$$

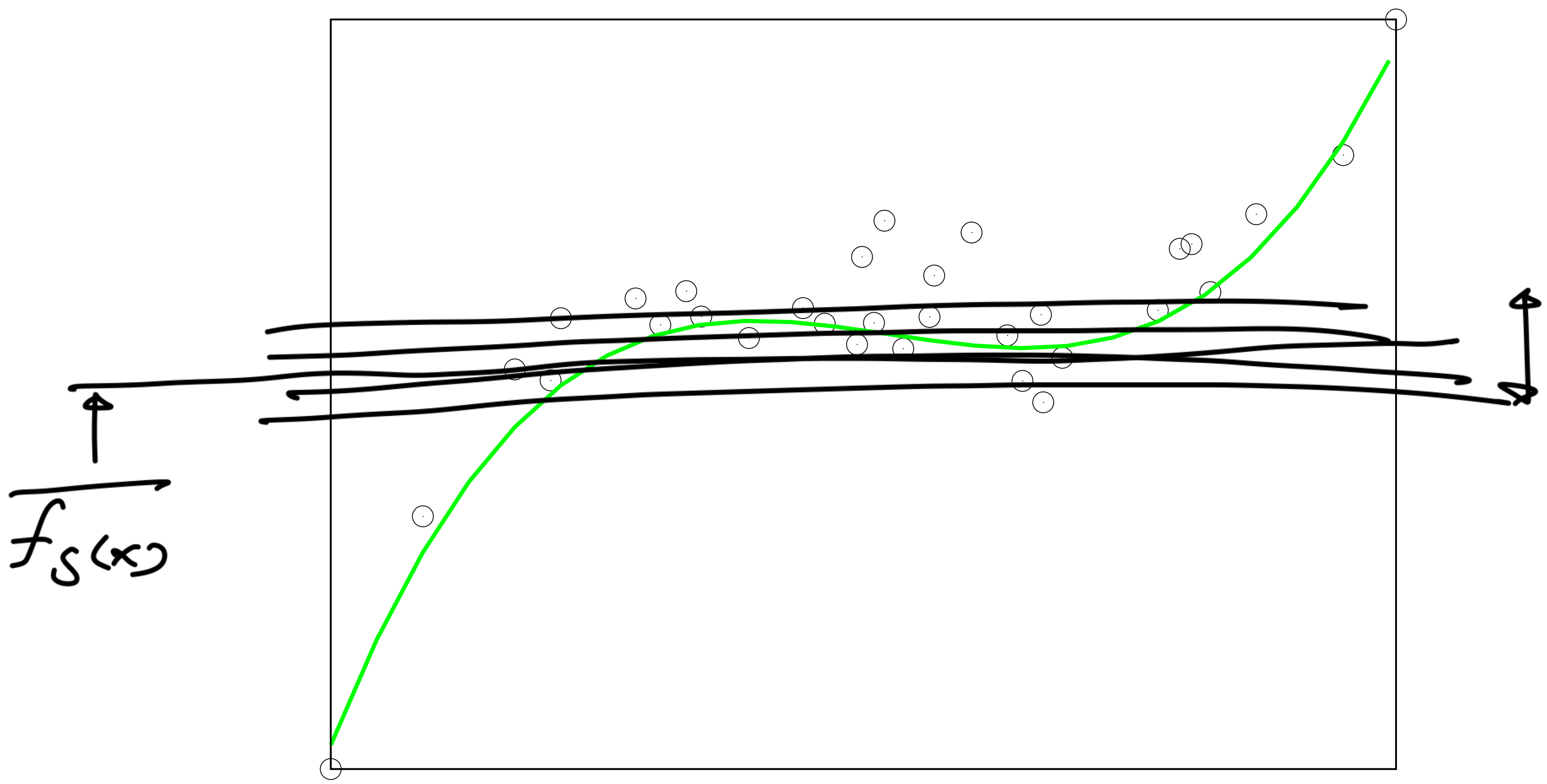
$$\overline{f_S(x)} = \frac{1}{4}(f_{S_1} + f_{S_2} + f_{S_3} + f_{S_4})$$

- low complexity model has a large **bias**

if we change N or model class, bias changes

Low complexity models

- suppose we train a constant function, many times each with *N* samples from $y = f_0(x) + \varepsilon$

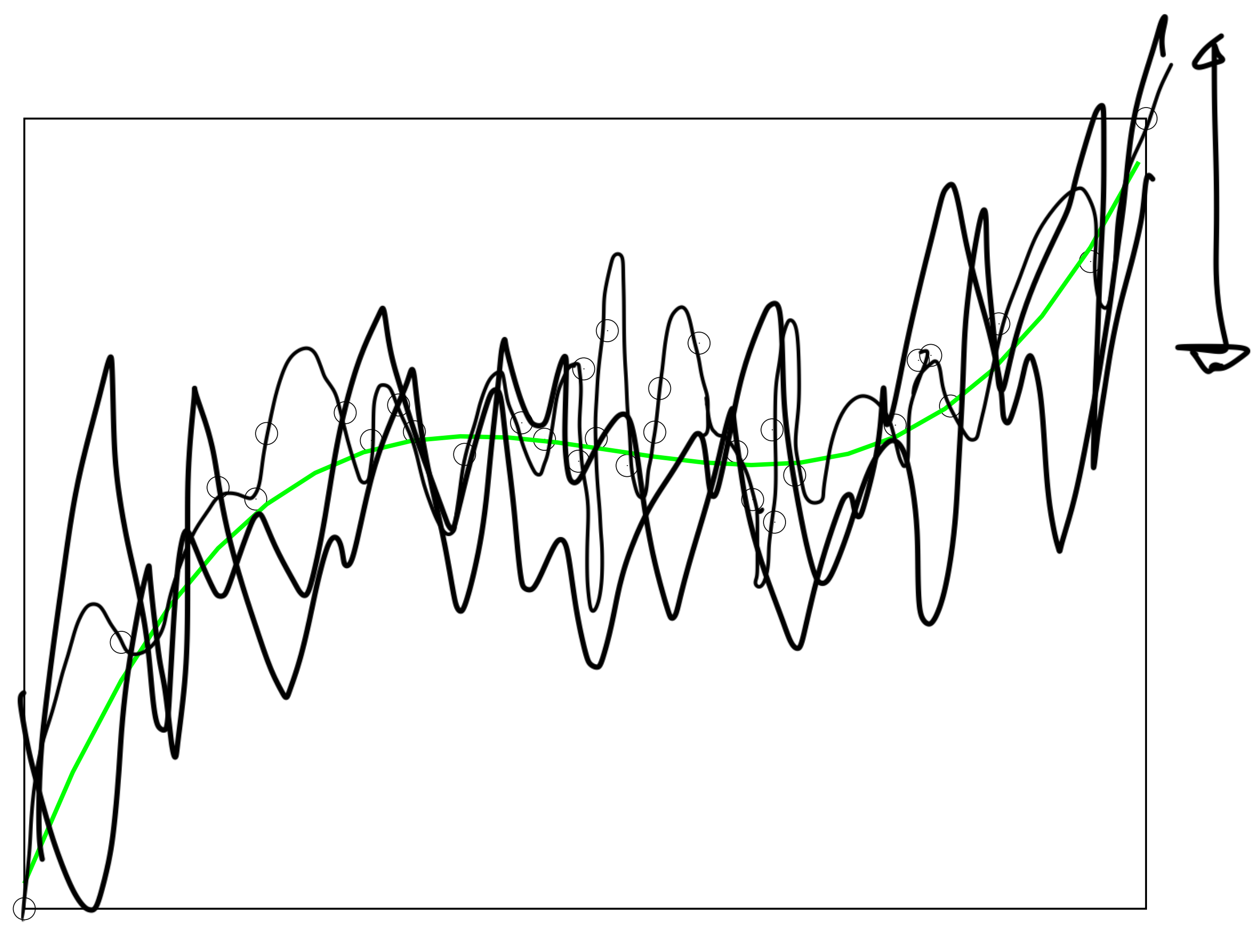


$$\text{Variance} = \mathbb{E}\left[\ (f_{S_i}(x) - \overline{f_S(x)})^2\ \right]$$

- low complexity model has a small **variance**

High complexity models

- suppose we train a high degree polynomial function, many times each with *N* samples from $y = f_0(x) + \varepsilon$



$w_0 + w_1 x + \cdots + w_p x^p$

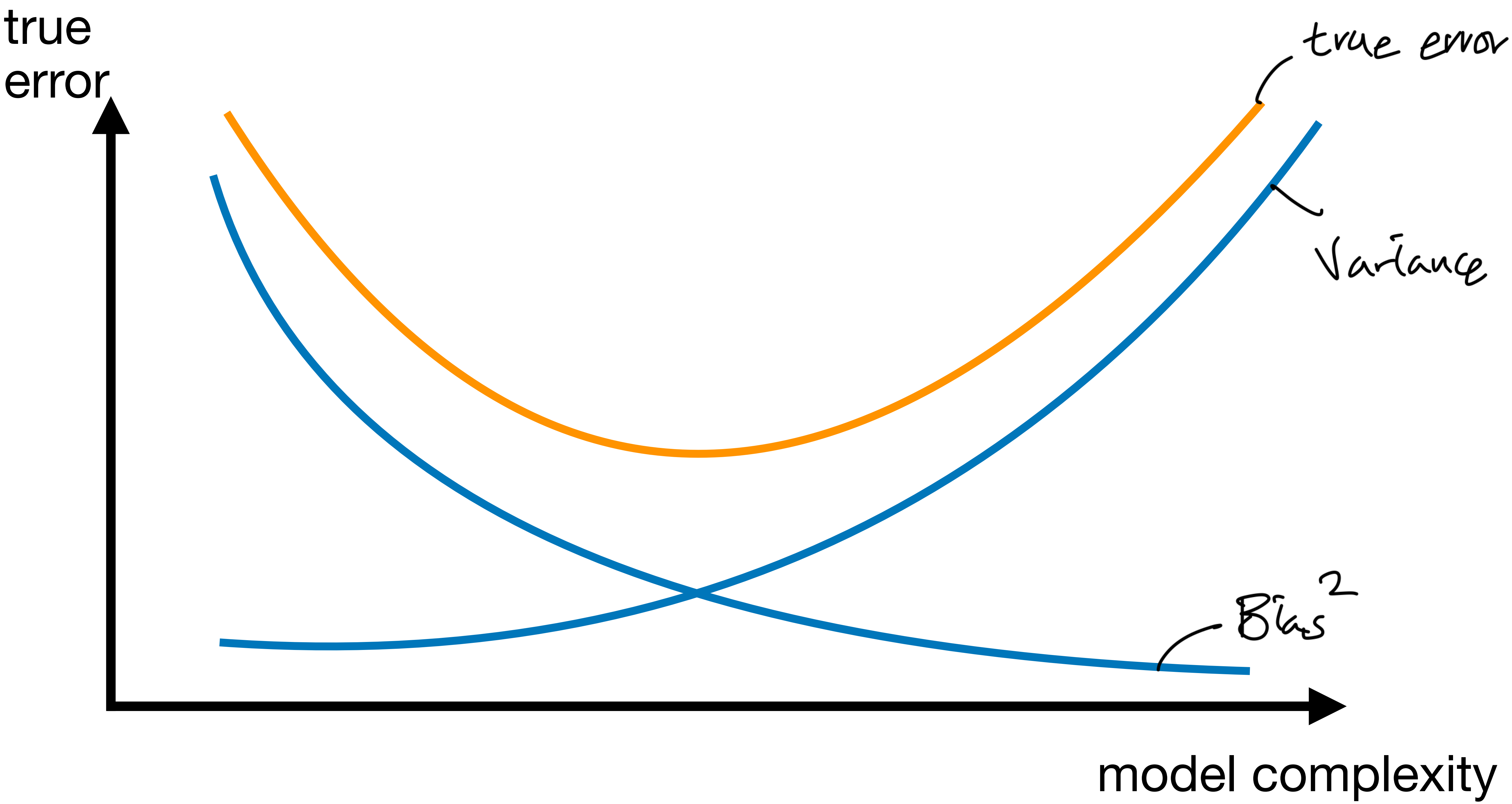
$$\text{Variance} = \mathbb{E}\big[\,(f_S(x) - \overline{f_S(x)})^2\,\big]$$

- high complexity model has a large **variance** but small **bias**

Bias-Variance tradeoff

- for fixed sample size N,

$$\mathrm{MSE}_{\mathrm{true}} \;=\; \mathrm{MSE}_{\mathrm{noise}} + \mathrm{Bias}^2 + \mathrm{Variance}$$



For fixed model complexity

- suppose we fix model complexity such that

$$f_0(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4 + w_5 x^5 + \varepsilon$$ ← True model

$$f(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$ ← Predictor