If the glove fits

A gentle introduction to ML

CHIDATA

CHIDATA

Recap: Simulation

- Use code to generate data

- Evaluate mean, variance, etc of complex quantities over simulation scenarios.

- Basic statistics is a starting point to help you write the code

Recap: Linear Models

$$y_i = A \cdot x_i + \epsilon$$

```
def simulate(x):
    A = #constant
    return A*x + np.random.randn(1)
```

Accurately models “small variation” systems

General Models

$$y_i = f_\theta(x_i, \epsilon_i)$$

Output

Input

Noise

"System or Model"

Each yi (simulation output) is generated by a function of xi plus noise

Theta: Parameters that govern how f behaves.

CHIDATA

More Interesting Example

$$y_i = ec\_count(\begin{bmatrix} 0.54 \\ ... \\ 0.48 \end{bmatrix} + sampling\_error)$$

Output

“System or Model”

Input

Noise

CHIDATA

Recap: Simulation Models

Linear: continuous input, continuous output

Classification: continuous input, discrete output

Mixture: discrete input, continuous output

Code to generate outputs!

Simulation v.s. Model Fitting

$$y_i = f_\theta(x_i, \epsilon_i)$$

Simulation: Observe xi, ei, and theta, **generate yi**

Fitting: Observe xi, yi, **infer theta**

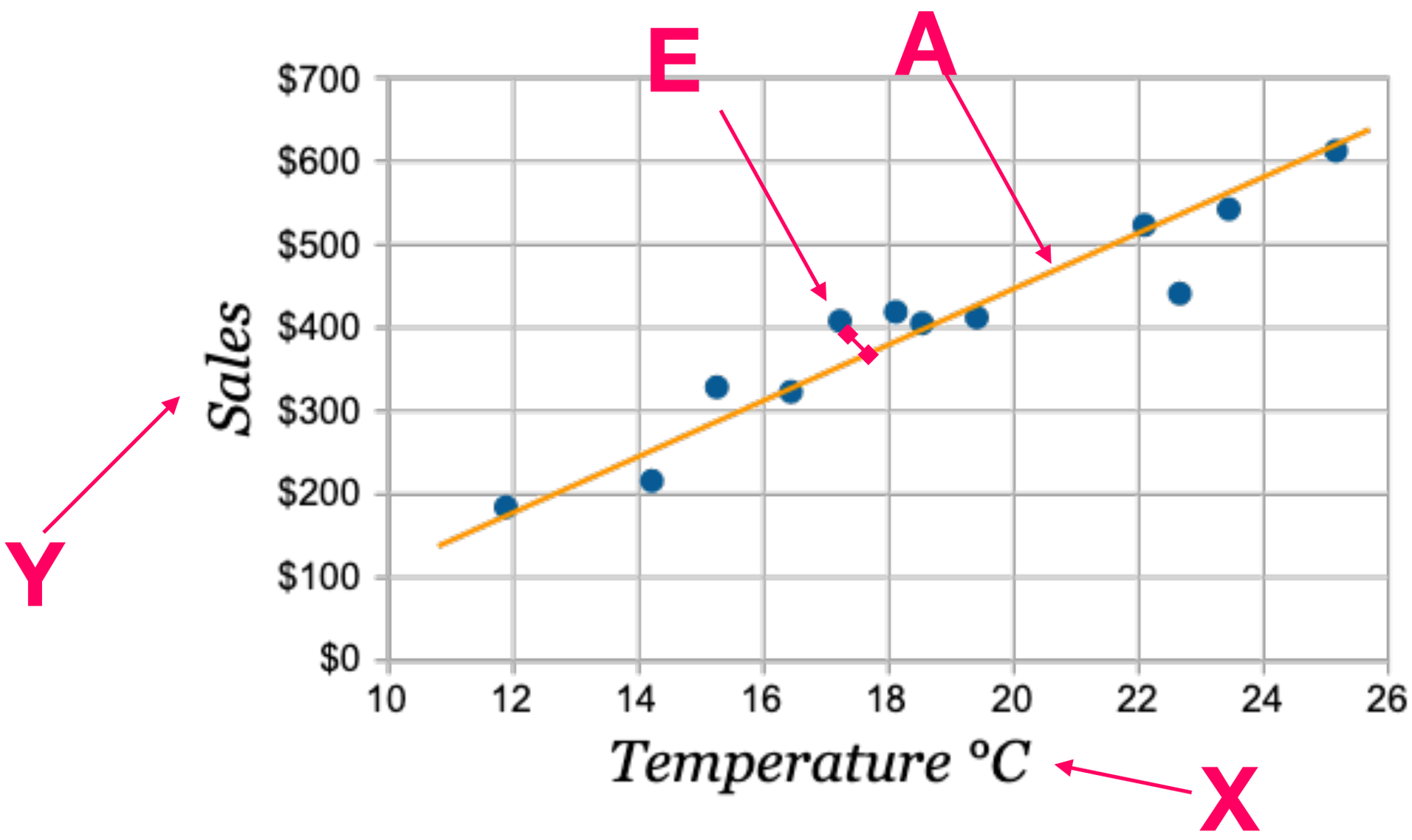
Why not observe epsilon?

"Generic" term that captures process/observation/modeling error

CHIDATA

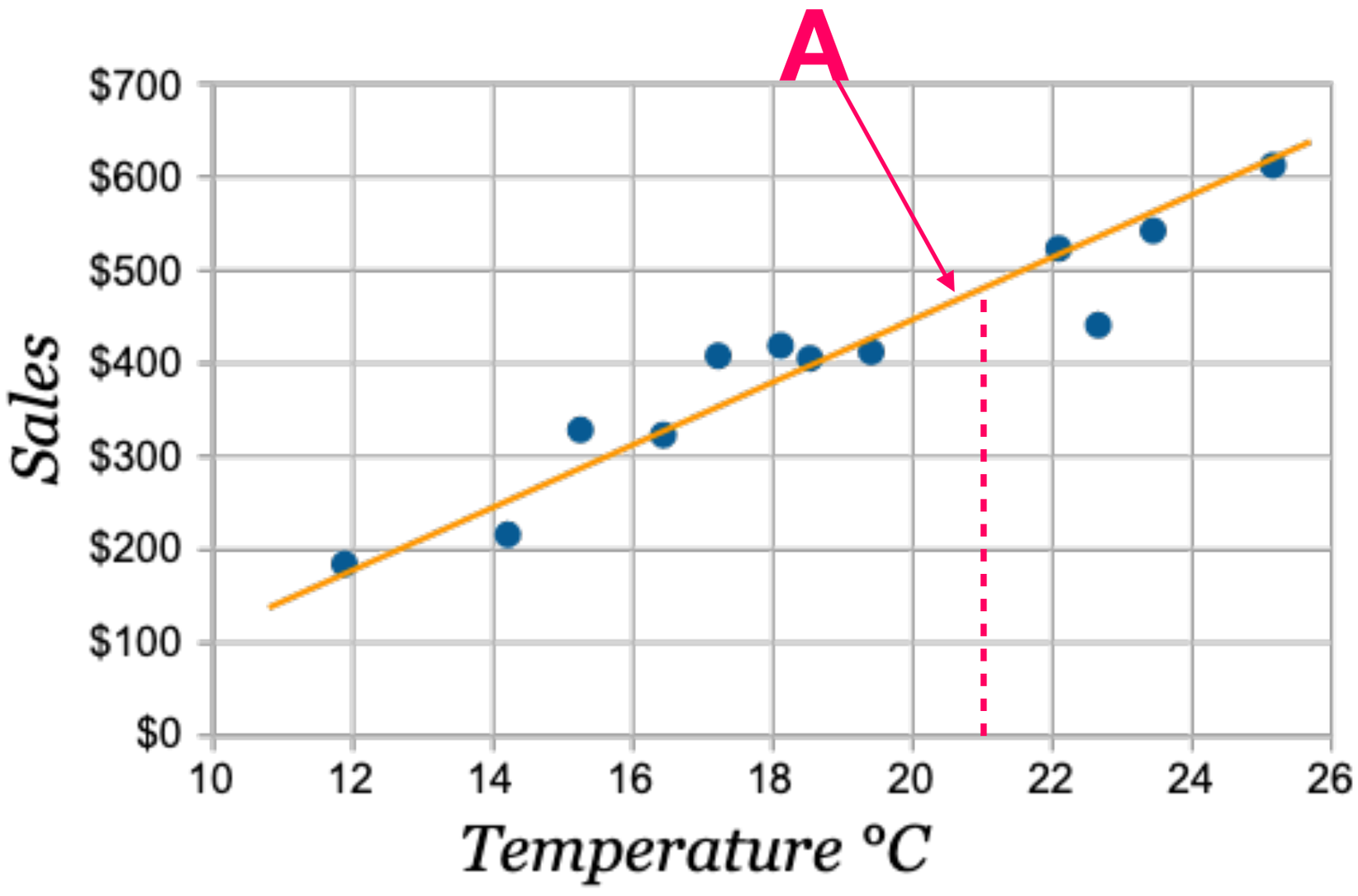
Example With Linear Model

$$y_i = A \cdot x_i + \epsilon$$



Example With Linear Model

I get an idealized predictive model: $y_i = A \cdot x_i$



CHIDATA

Example With Linear Model

$$y_i = A \cdot x_i + \epsilon$$

```
>>> import numpy as np
>>> from sklearn.linear_model import LinearRegression

>>> X = # temperatures
>>> y = #sales
>>> reg = LinearRegression().fit(X, y)

>>> reg.coef_, reg.intercept_
(array([26.00]), -90.0000…)
>>> reg.predict(np.array([21])) #sales = 26*temp - 90
array([456])
```

CHIDATA

How Does It Work?

Least squares principle

$$\min_{a,c} \sum_{i=1}^{N} (y_i - (ax_i + c))^2$$

Actual **Prediction**

Find parameters that minimize the squared error between prediction and actual.

CHIDATA

How Does It Work?

Where does epsilon fit in?

$$\min_{a,c} \sum_{i=1}^{N} (y_i - (ax_i + c))^2 \qquad \epsilon_i = (y_i - (ax_i + c)$$

$$\sum_{i=1}^{N} \epsilon_i^2 \approx \mathbf{Var}[\epsilon] + \mathbf{E}[\epsilon]^2$$

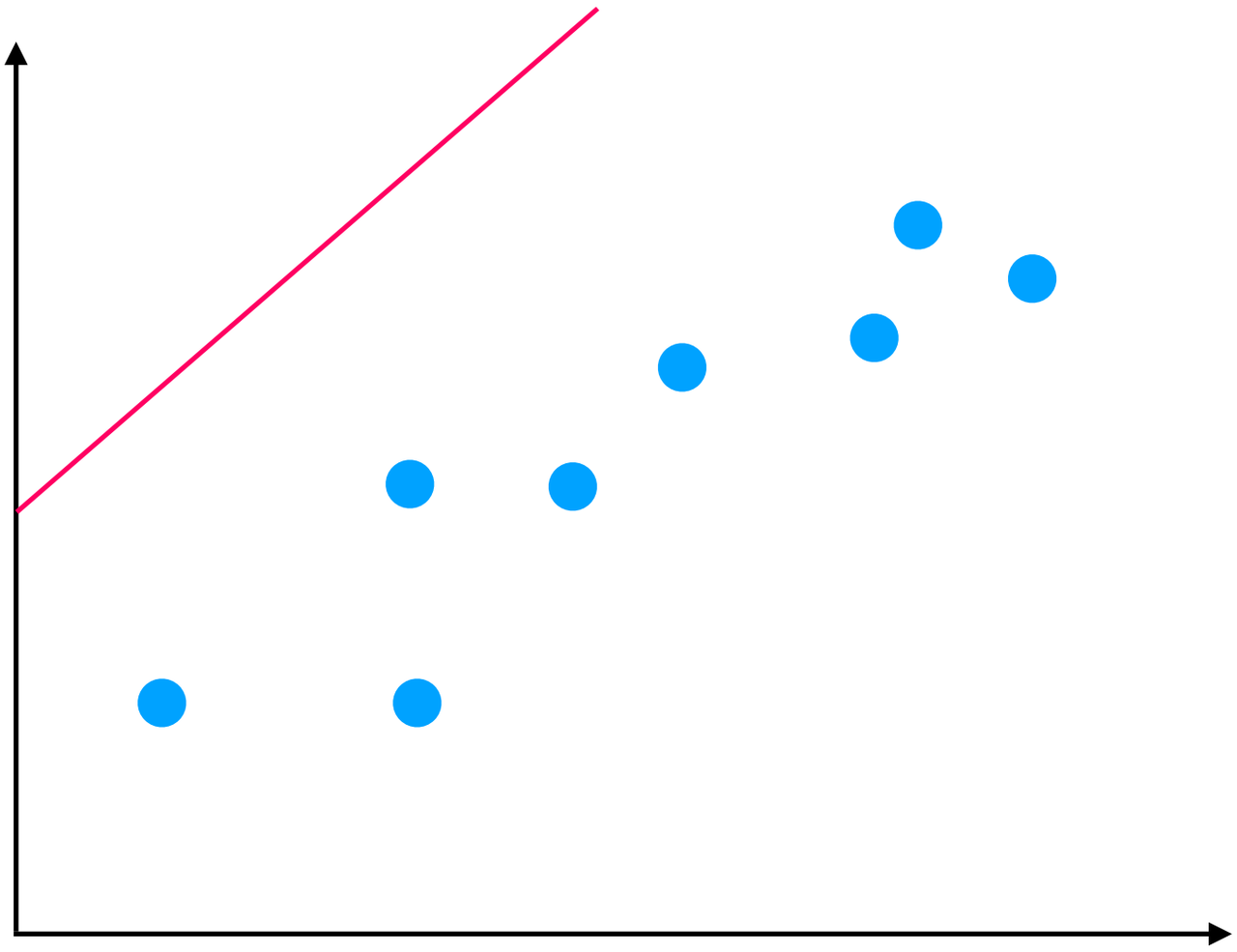
Variation In Error

“Bias” of the model

CHIDATA

High Bias

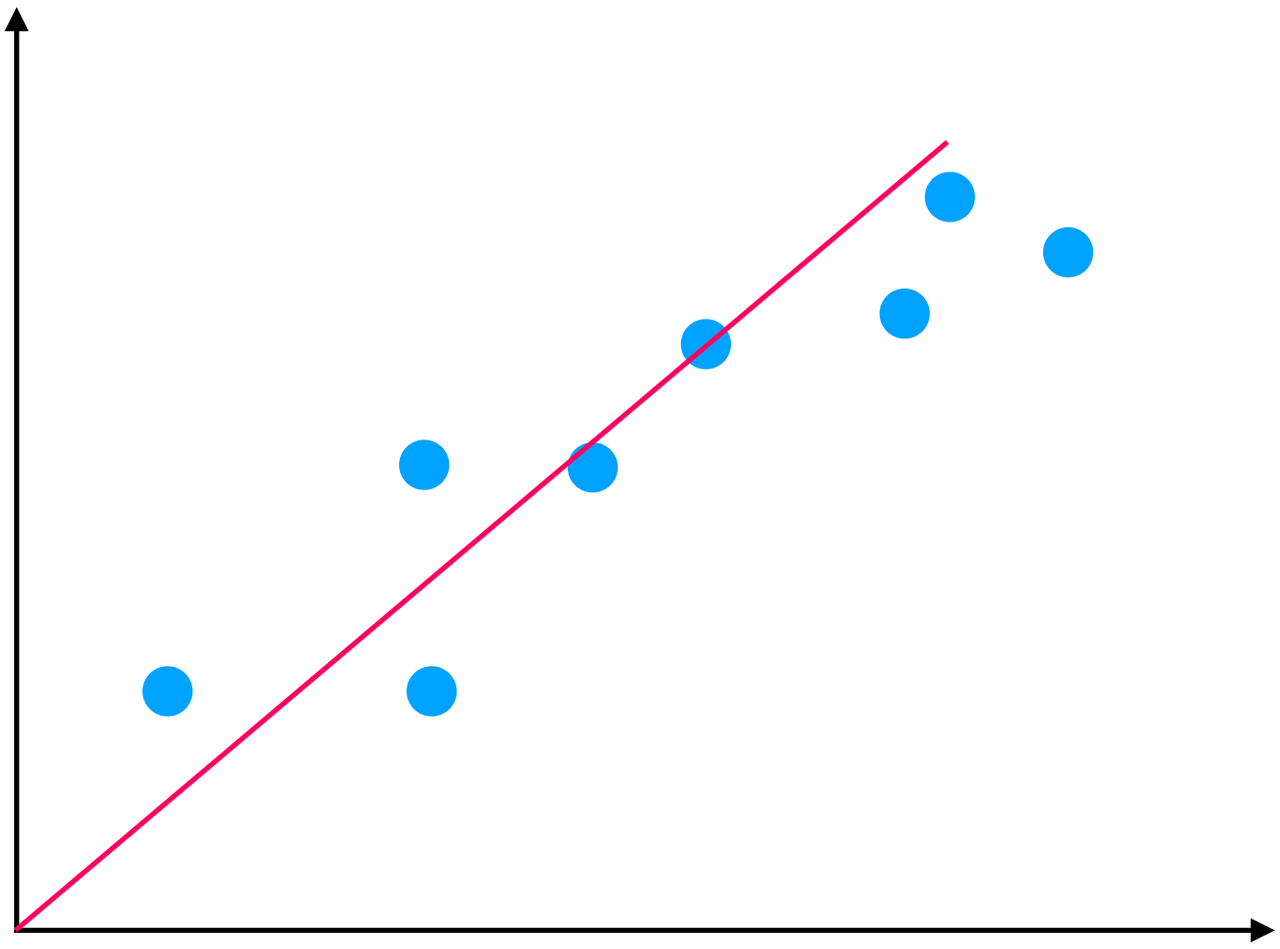
$$\min_{a,c} \mathbf{Var}[\epsilon] + \mathbf{Bias}[\epsilon]^2$$



CHIDATA

Low Bias

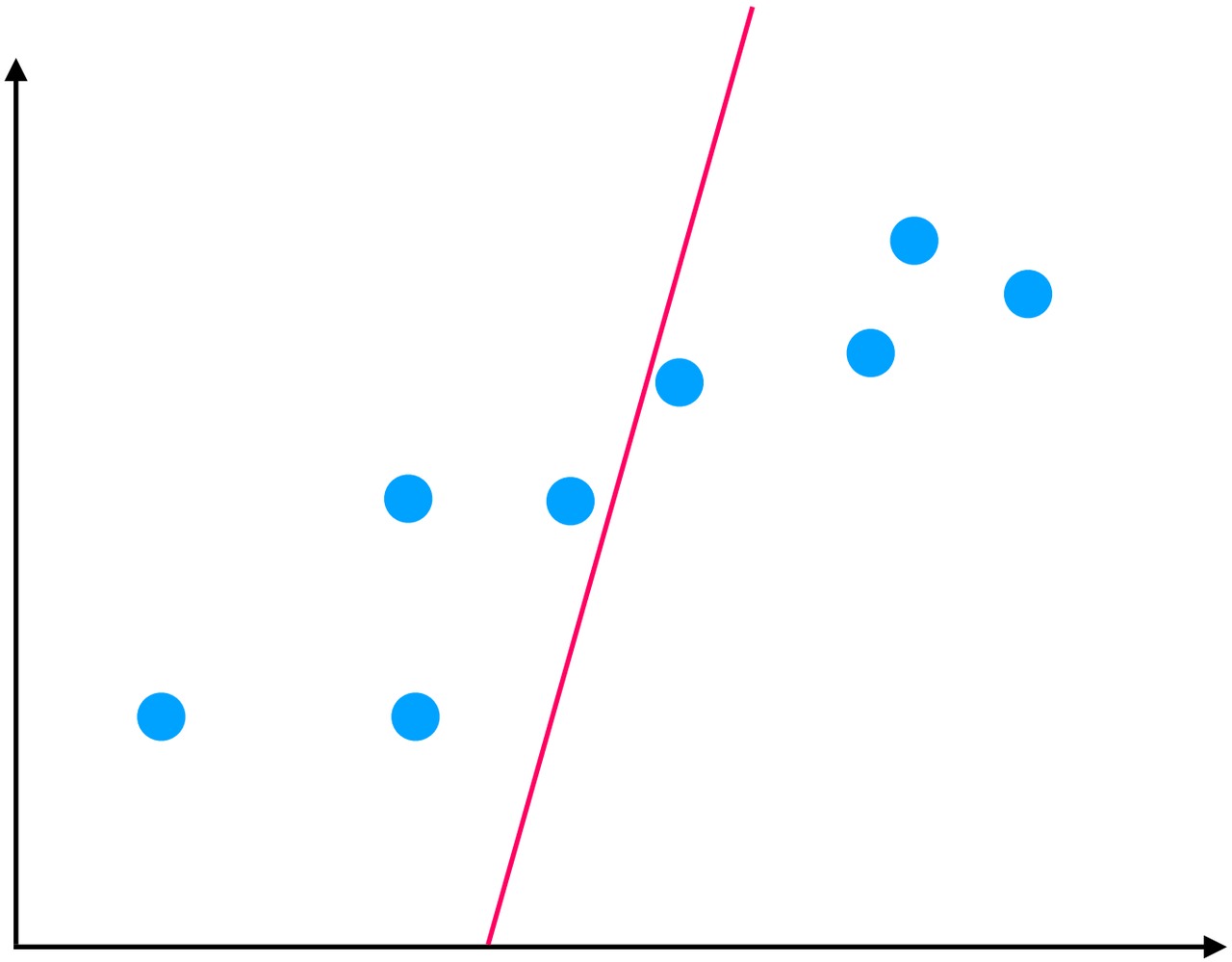
$$\min_{a,c} \mathbf{Var}[\epsilon] + \mathbf{Bias}[\epsilon]^2$$



CHIDATA

High Variance

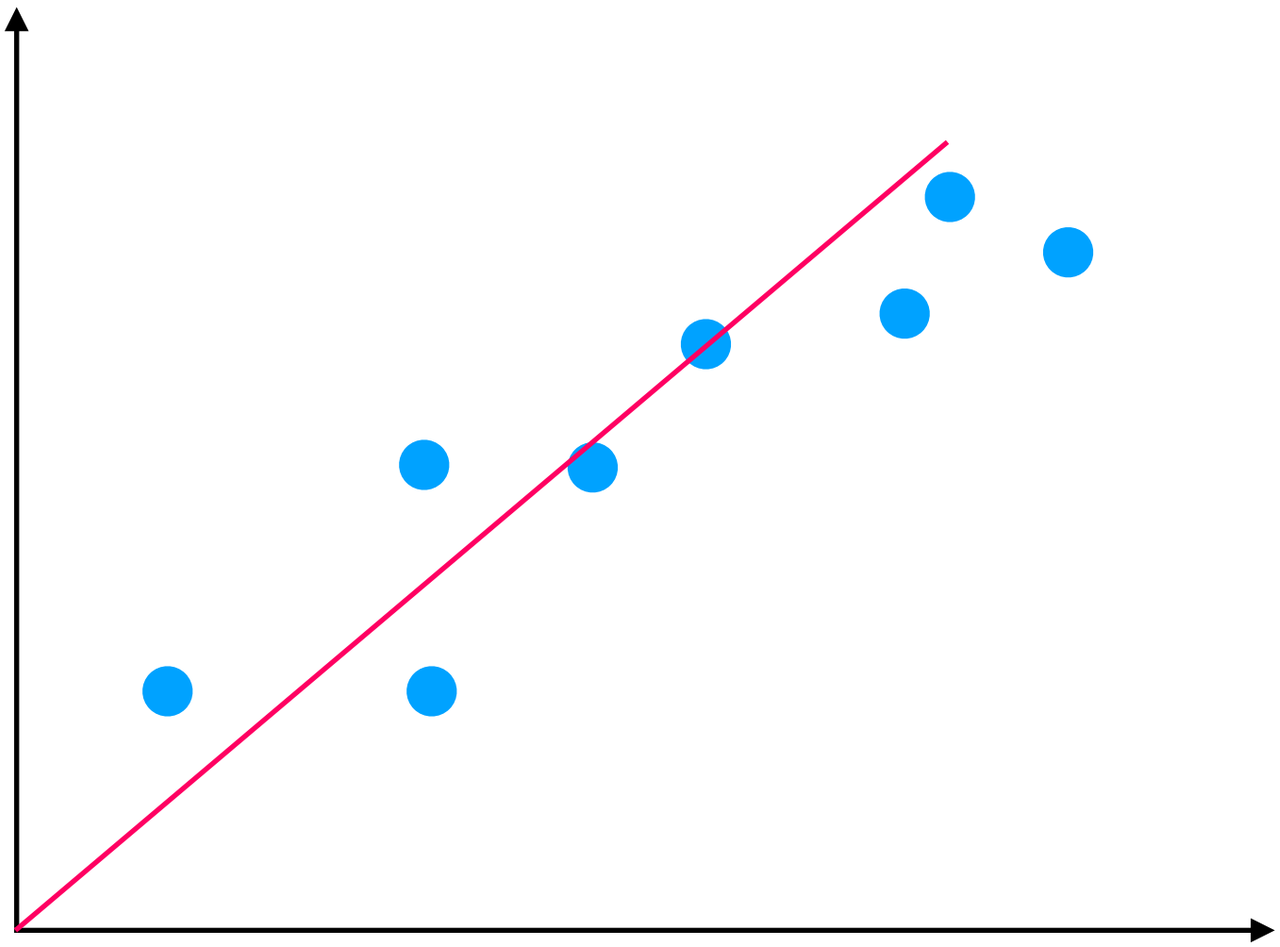
$$\min_{a,c} \mathbf{Var}[\epsilon] + \mathbf{Bias}[\epsilon]^2$$



CHIDATA

Low Variance

$$\min_{a,c} \mathbf{Var}[\epsilon] + \mathbf{Bias}[\epsilon]^2$$



CHIDATA

Bias-Variance Tradeoff

Bias: How closely does your selected model align with the data

Variance: How different predictions are (in terms of error).

CHIDATA

Bias-Variance Tradeoff

Bias: How closely does your selected model align with the data

Measures "systematic" misprediction

CHIDATA

Bias-Variance Tradeoff

Variance: How different predictions are (in terms of error).

Measures predictability of error

CHIDATA

How Does It Work?

Least squares principle

$$\min_{a,c} \sum_{i=1}^{N} (y_i - (ax_i + c))^2$$

Minimize the balance of bias and variance

$$\min_{a,c} \mathbf{Var}[\epsilon] + \mathbf{Bias}[\epsilon]^2$$

CHIDATA

Problem?

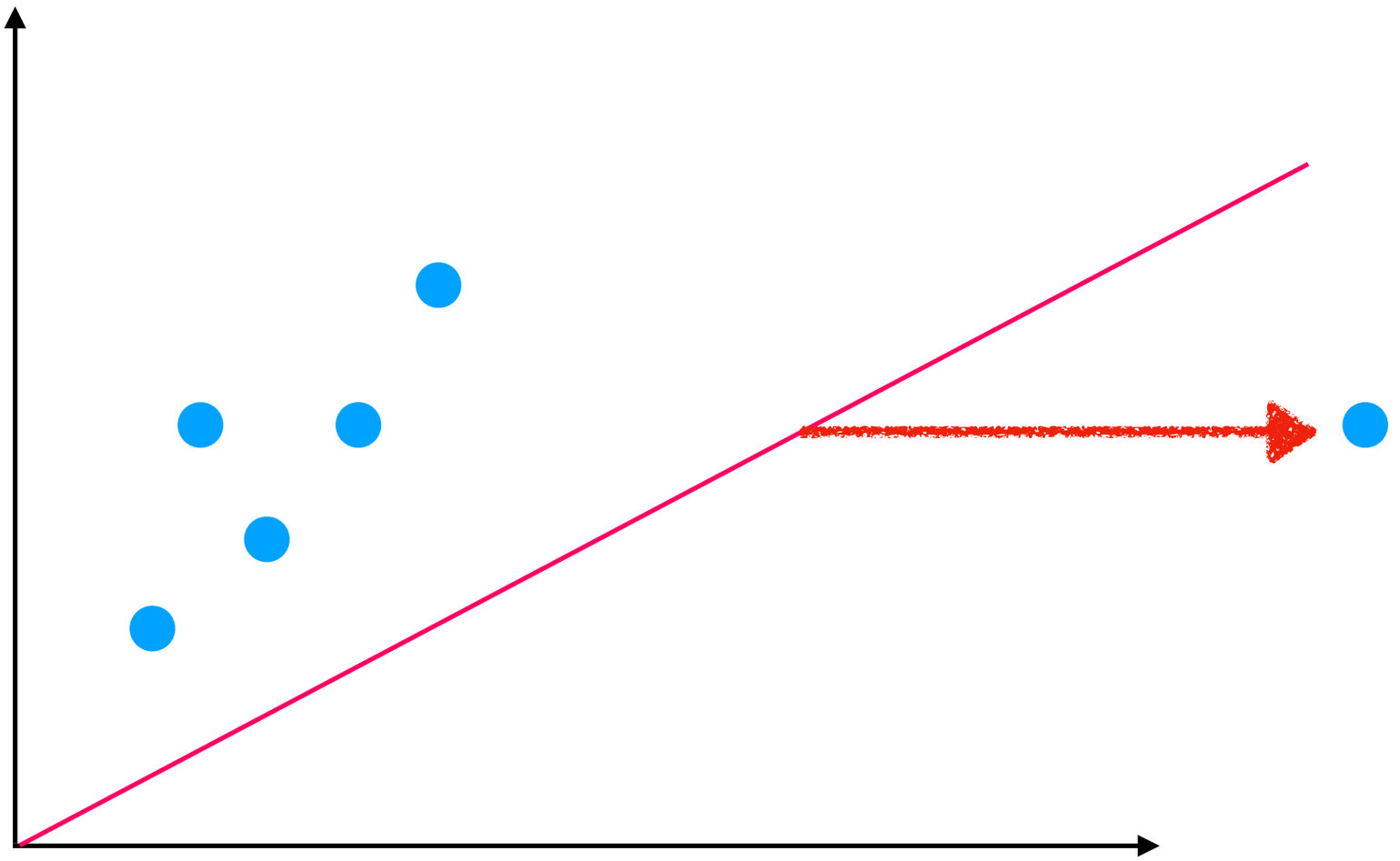
Minimize the balance of bias and variance

$$\min_{a,c} \mathbf{Var}[\epsilon] + \mathbf{Bias}[\epsilon]^2$$

Not robust!!

Problem: Work with finite data

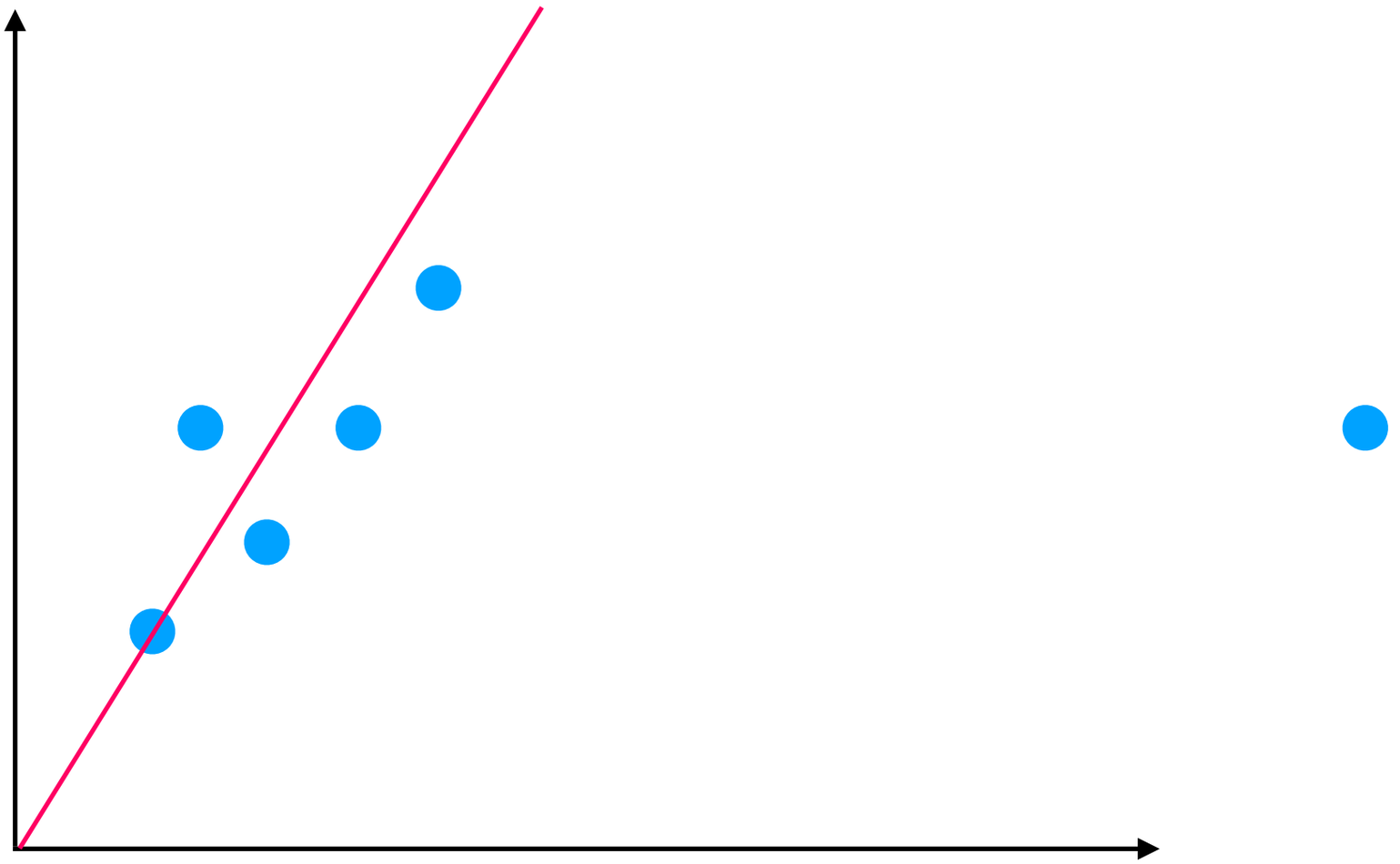
$$\min_{a,c} \mathbf{Var}[\epsilon] + \mathbf{Bias}[\epsilon]^2$$



CHIDATA

Problem: Work with finite data

$$\min_{a,c} \mathbf{Var}[\epsilon] + \mathbf{Bias}[\epsilon]^2$$



Accept some bias if that means much lower variance

CHIDATA

Regularization

Bias that is baked into the optimization problem to avoid being fooled by outliers

$$\min_{\theta} \sum_{i=1}^{N} (y_i - (ax_i + c))^2 + \lambda \cdot \mathbf{Reg}(a, c)$$

Penalize certain solutions

Lambda is the amount of bias

CHIDATA

Types of Models

`sklearn.linear_model.LinearRegression`

$$\min_{\theta} \sum_{i=1}^{N} (y_i - Ax_i + c)^2$$

`sklearn.linear_model.Ridge`

$$\min_{\theta} \sum_{i=1}^{N} (y_i - A \cdot x_i + c)^2 + \lambda (\sum_{j=1}^{d} A_d^2 + c^2)$$

`sklearn.linear_model.Lasso`

$$\min_{\theta} \sum_{i=1}^{N} (y_i - A \cdot x_i + c)^2 + \lambda (\sum_{j=1}^{d} |A_d| + |c|)$$

A General Formula For Fitting

$$\min_{\theta} \sum_{i=1}^{N} L(y_i, f_\theta(x_i)) + \lambda g(\theta)$$

Measure of Success

Actual

Prediction

Regularization

$$\min_{\theta} \sum_{i=1}^{N} (y_i - (ax_i + c))^2 + \lambda \cdot \mathbf{Reg}(a, c)$$

Measure of Success

Actual

Prediction

Regularization

CHIDATA

"Loss Functions"

$$\min_{\theta} \sum_{i=1}^{N} L(y_i, f_\theta(x_i)) + \lambda g(\theta)$$

Measure of Success

$$L(y_i, f_\theta(x_i)) = (y_i - f_\theta(x_i))^2 \quad \textbf{Sq. error}$$

$$L(y_i, f_\theta(x_i)) = |y_i - f_\theta(x_i)| \quad \textbf{Abs error}$$

$$L(y_i, f_\theta(x_i)) = \mathbf{1}(y_i! = f_\theta(x_i)) \quad \textbf{Exact match only}$$

CHIDATA

"Loss Functions"

$$\min_{\theta} \sum_{i=1}^{N} L(y_i, f_\theta(x_i)) + \lambda g(\theta)$$

Measure of Success

Minimizing "average" error (for some definition of error)

CHIDATA

"Loss Functions"

$$\min_{\theta} \sum_{i=1}^{N} L(y_i, f_\theta(x_i)) + \lambda g(\theta)$$

Measure of Success

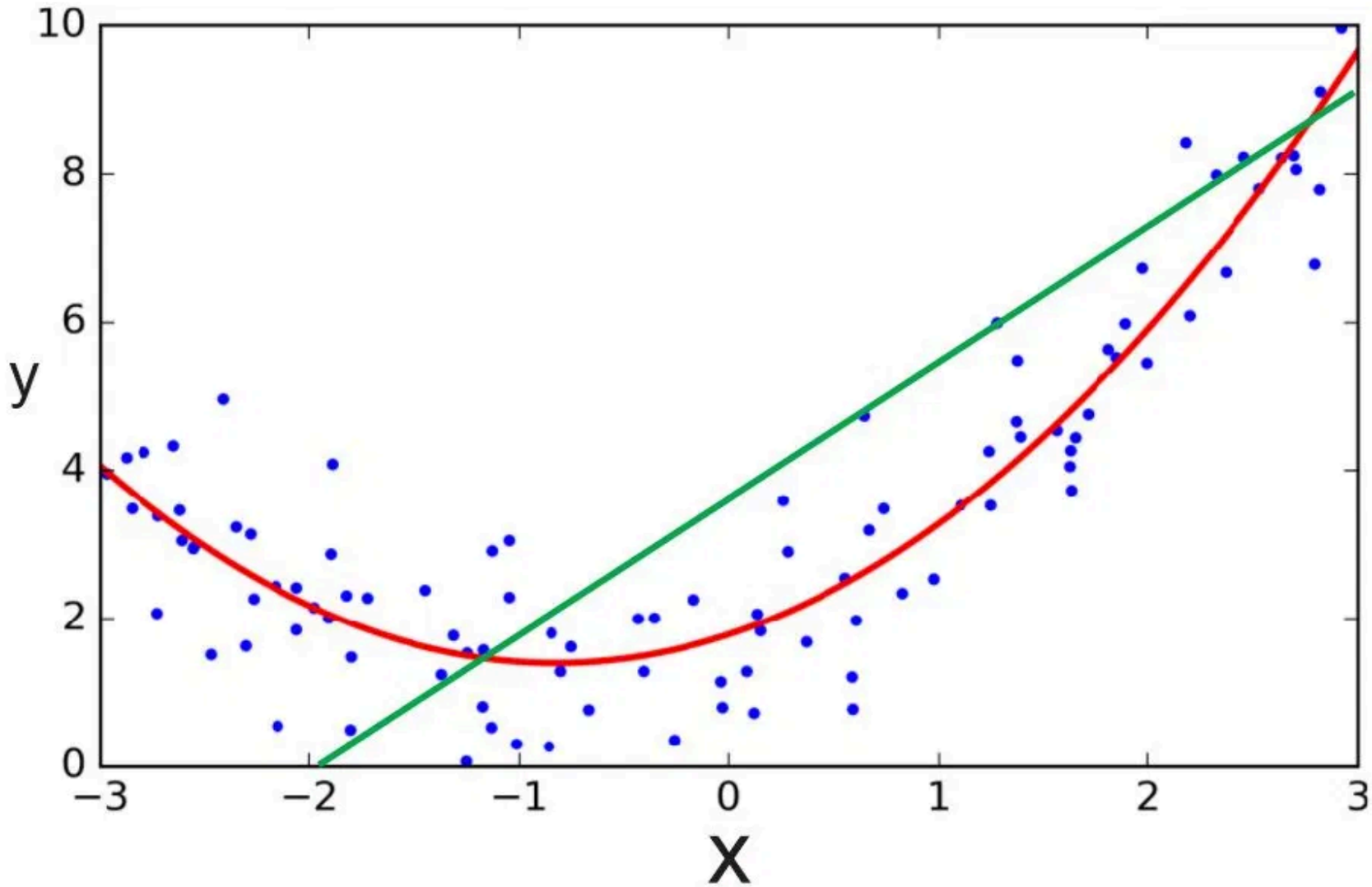
Also relevant to classification problems!

More on this next time…

CHIDATA

Quadratic Fit

$$\min_{a,b,c} \sum_{i=1}^{N} (y_i - (ax_i^2 + bx_i + c))^2$$



CHIDATA

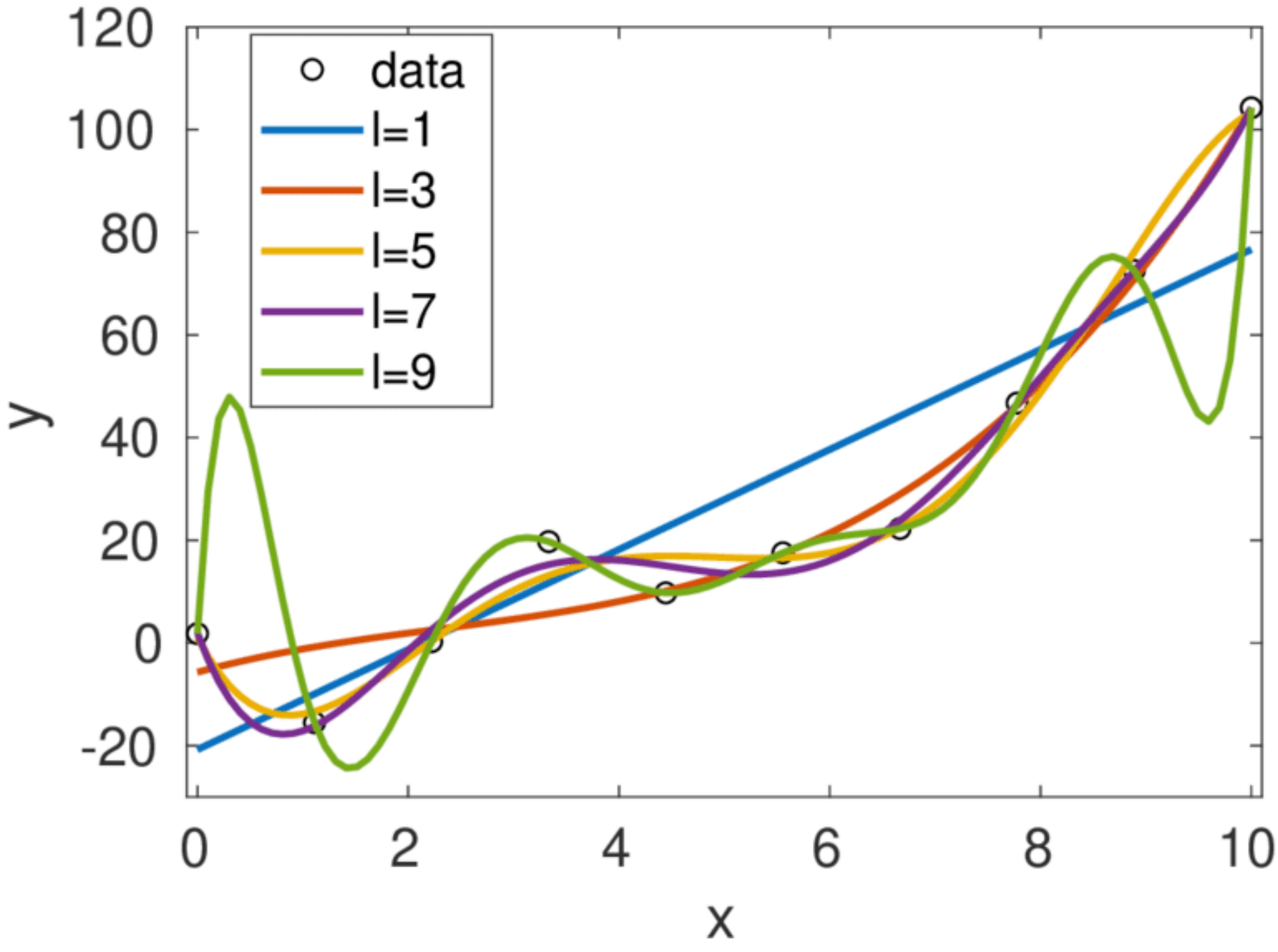
Quadratic Fit

$$\min_{a,b,c} \sum_{i=1}^{N} (y_i - (ax_i^2 + bx_i + c))^2$$

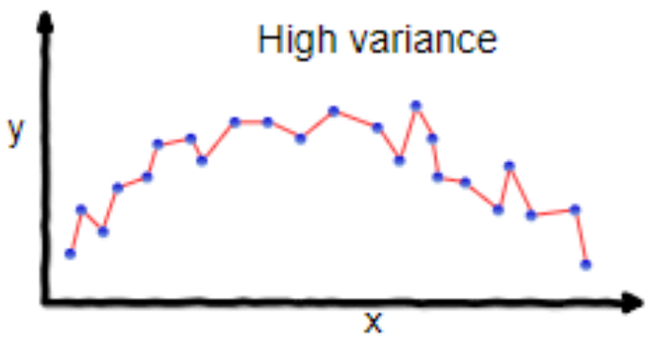
The “class” of models that are considered.

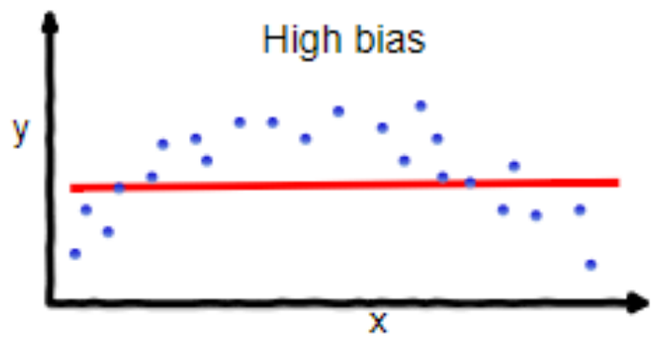
CHIDATA

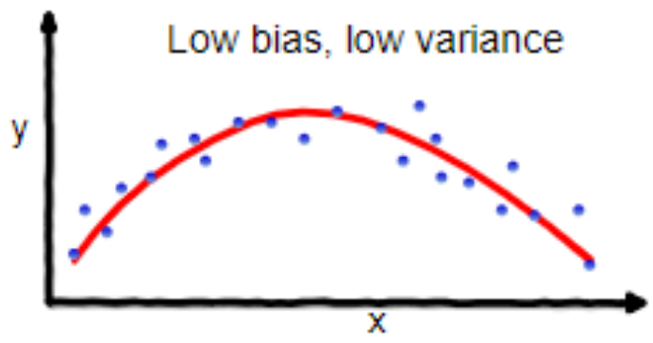
Why Not More?



Bias-Variance Tradeoff



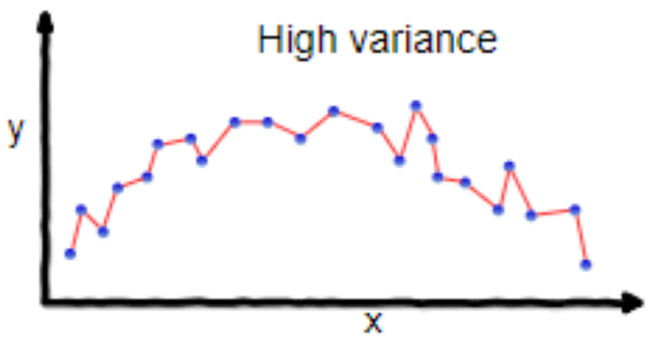


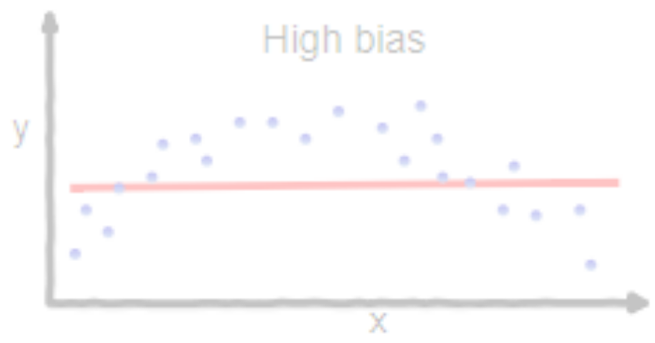


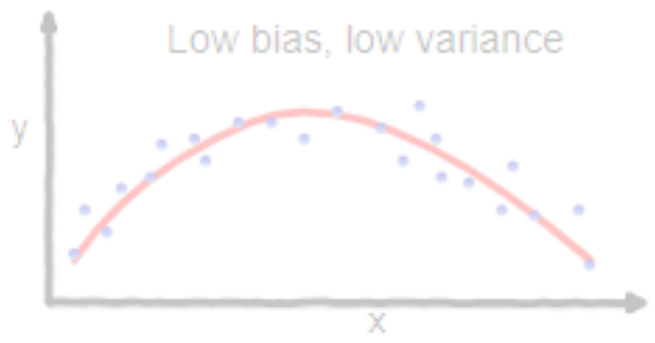
overfitting **underfitting** Good balance

Large classes (more parameters) have **lower bias** but **higher variance**.

Bias-Variance Tradeoff



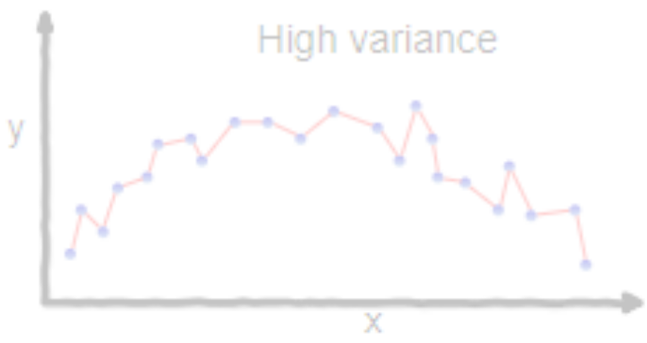


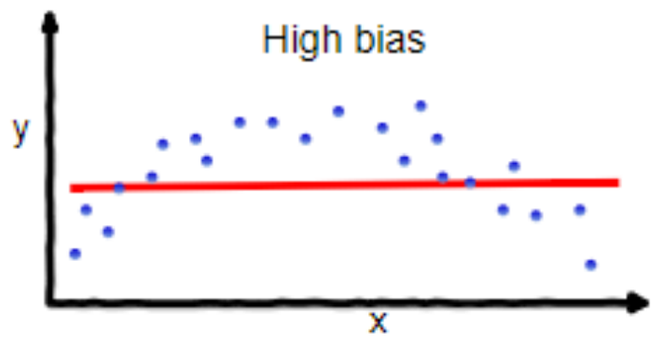


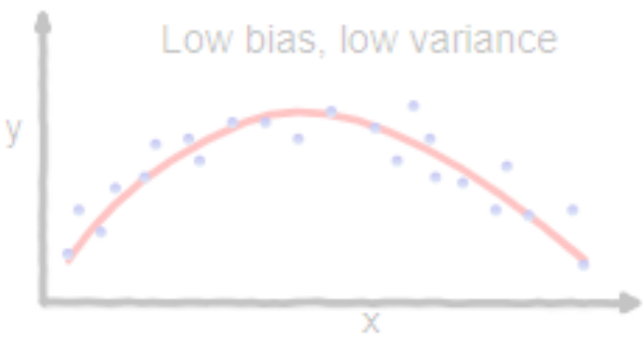
overfitting underfitting Good balance

- Very sensitive to small amount of noise
- Misprediction due to data variation.

Bias-Variance Tradeoff





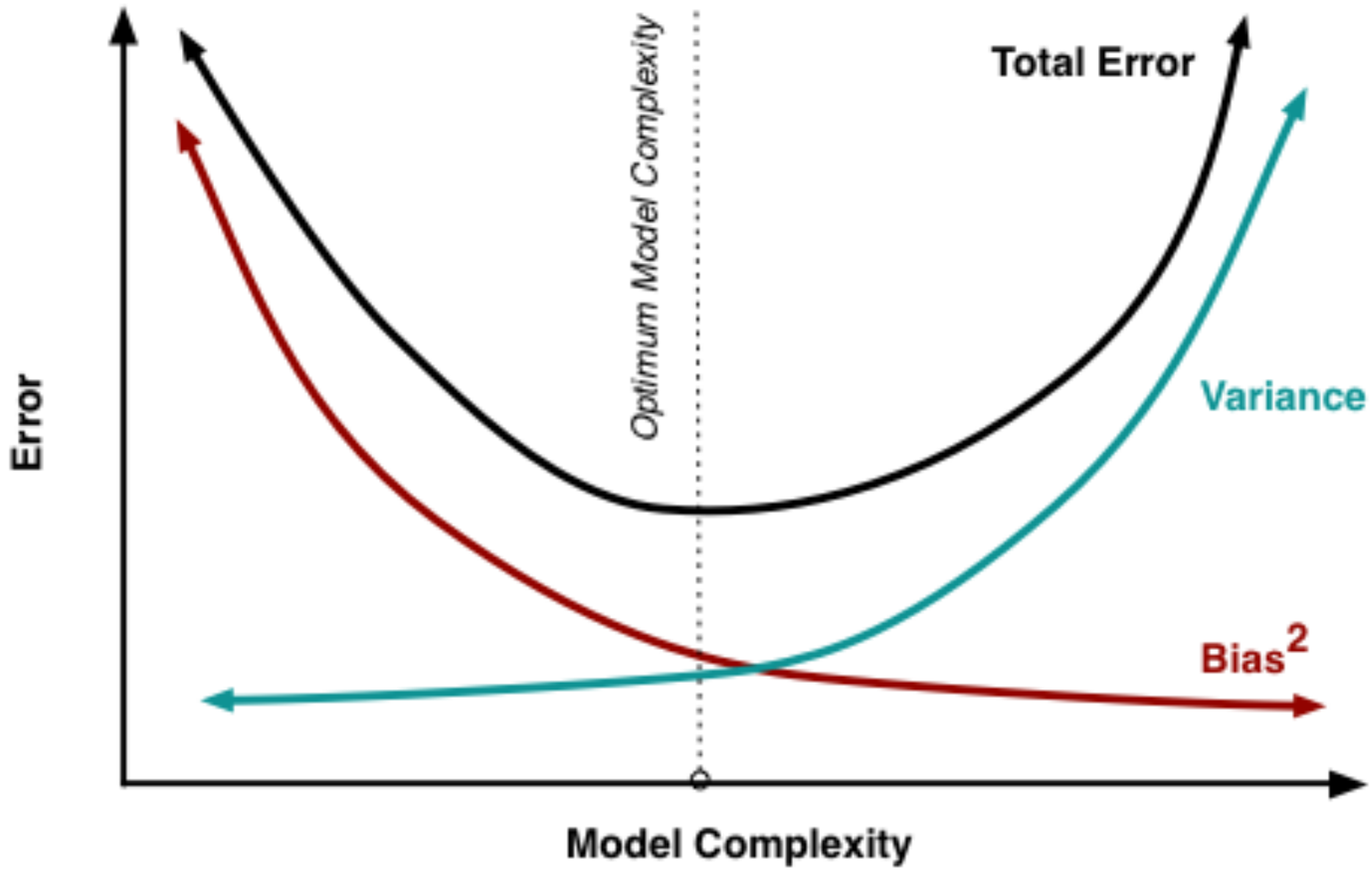


overfitting **underfitting** Good balance

- Misprediction due to model mismatch

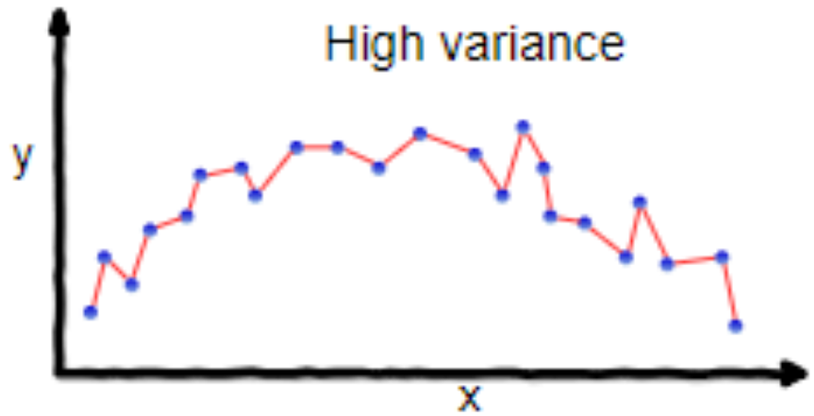
CHIDATA

Bias-Variance Tradeoff



CHIDATA

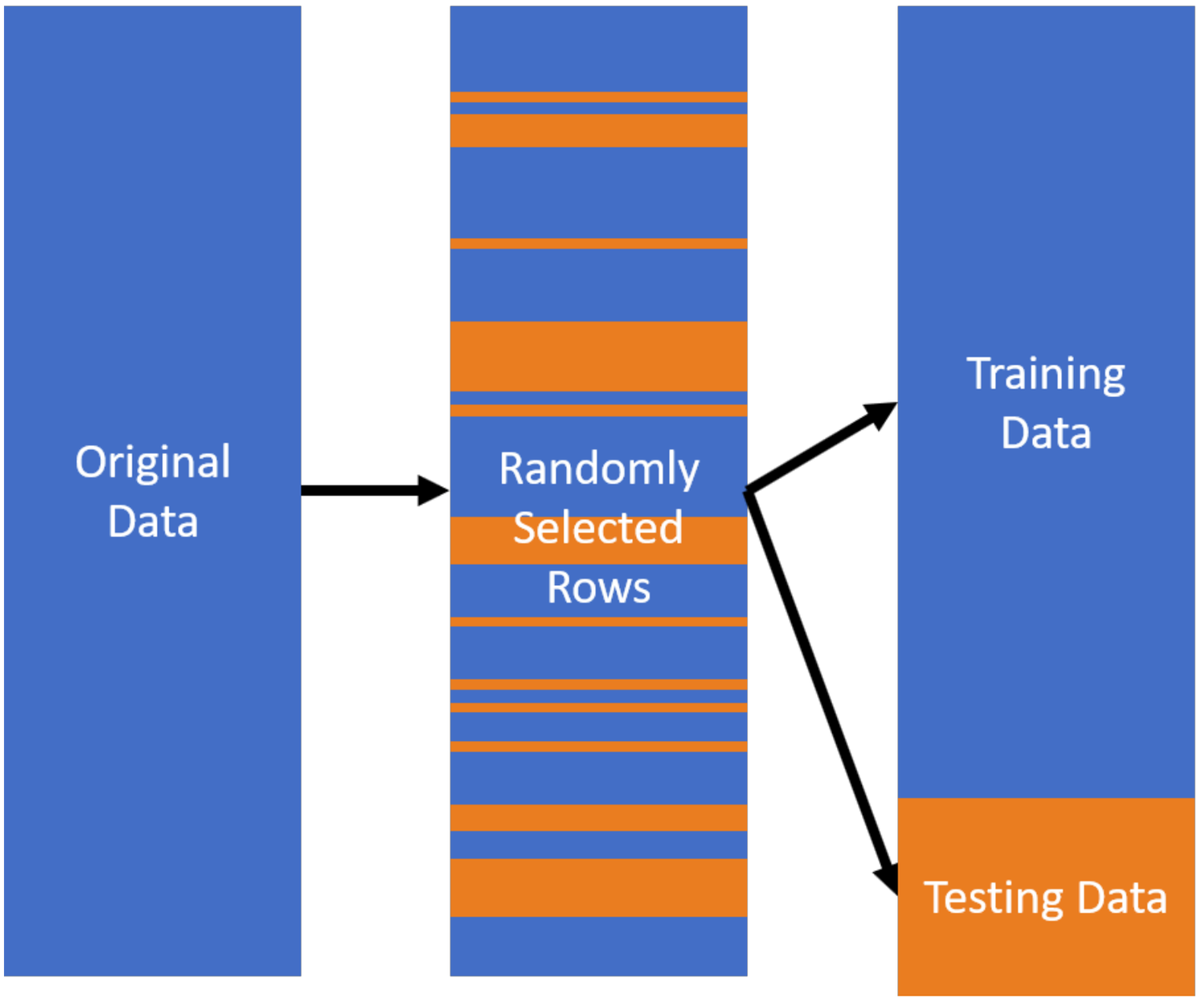
High Variance = Sensitivity to Unseen Data



Works well on data you have, won't work for prediction.

Idea: Evaluate models on unseen data to test for overfitting.

Training and Testing



CHIDATA

Training and Testing

```
>>> import numpy as np
>>> from sklearn.model_selection import train_test_split

>>> X_train, X_test, y_train, y_test = \
                    train_test_split(X, y, test_size=0.2)
```

Evaluate memorizing the data in the training set.

CHIDATA

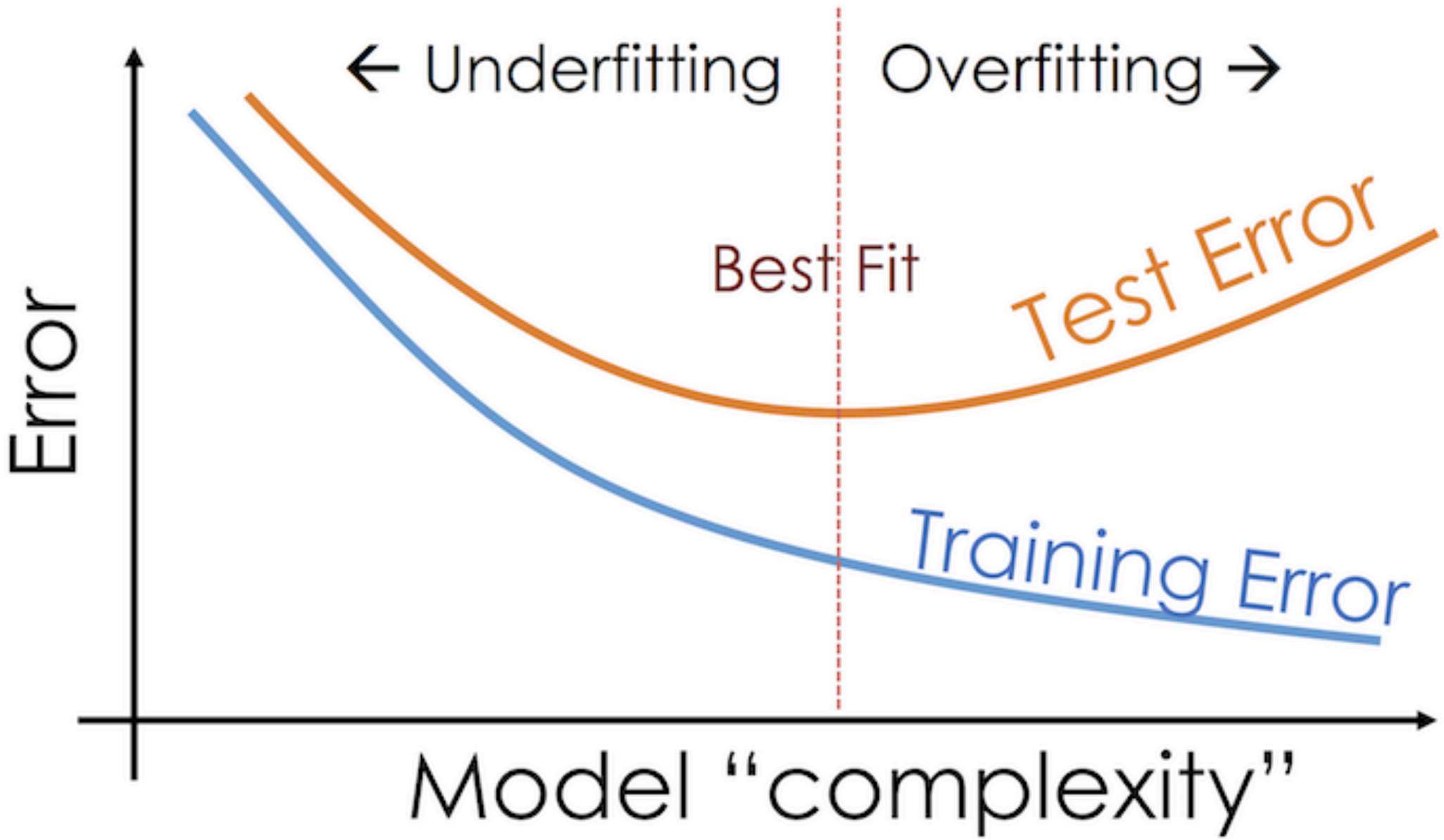
Training and Testing

Training error: average loss over the training set

Testing error: average loss over the testing set

CHIDATA

Training and Testing



CHIDATA

Summary

$$\min_{\theta} \sum_{i=1}^{N} L(y_i, f_\theta(x_i)) + \lambda g(\theta)$$

Always possible to accurately fit data you already have! (if your model is complex enough…)

Use withheld data (called testing data) to evaluate this effect.