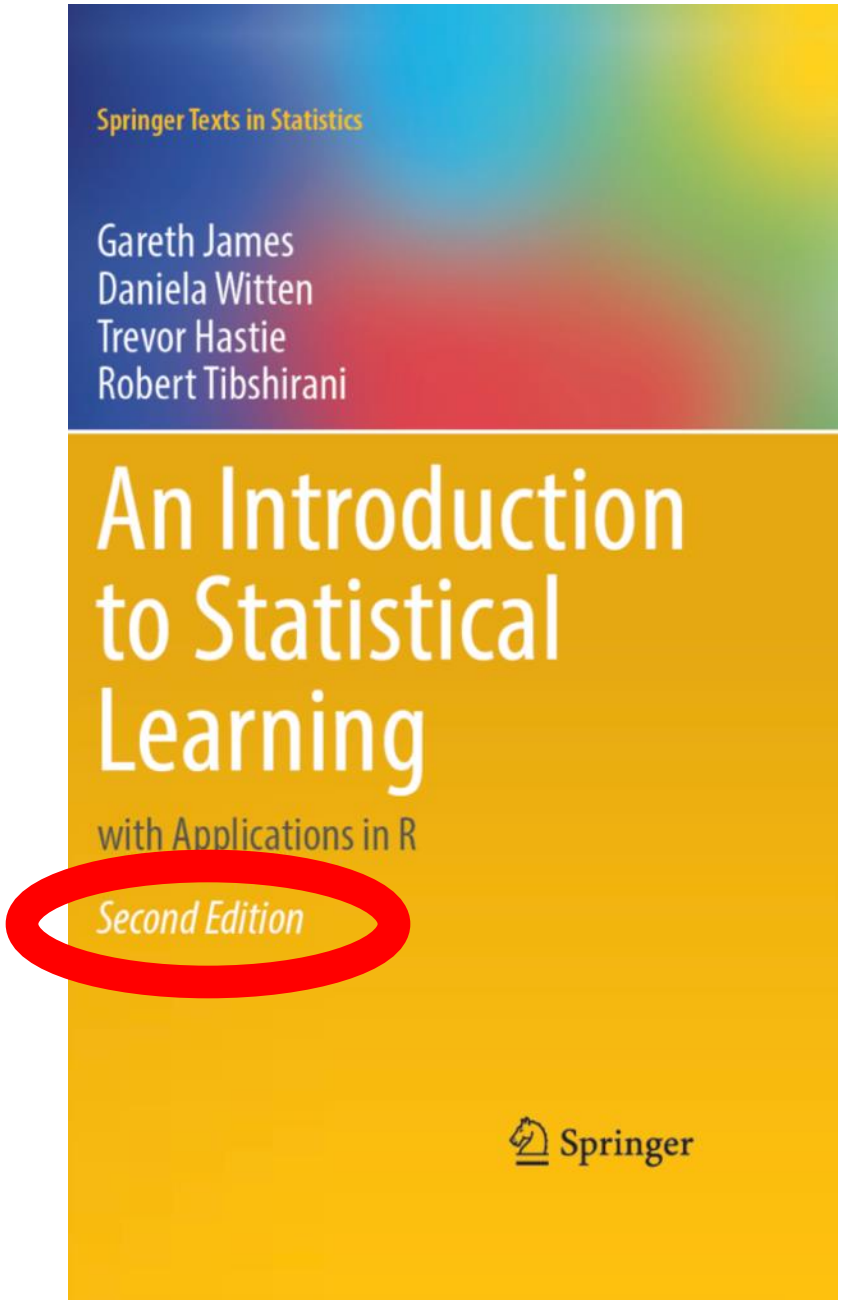
# Data Wrangling and Data Analysis
# **Machine learning!**

**Daniel Oberski**

Department of Methodology & Statistics

Utrecht University

*VERSION: 2025-09-28*




Online free version:
https://www.statlearning.com/

# Bird’s eye (high-level) view of machine learning

# **ML definition** (Samuel/Mitchell, 1959)

”A computer program is said to learn from **experience** *E* with respect to some class of **tasks** *T* and **performance measure** *P* if its performance at tasks in *T*, as measured by *P*, improves with experience *E*.”

# Some *translation*

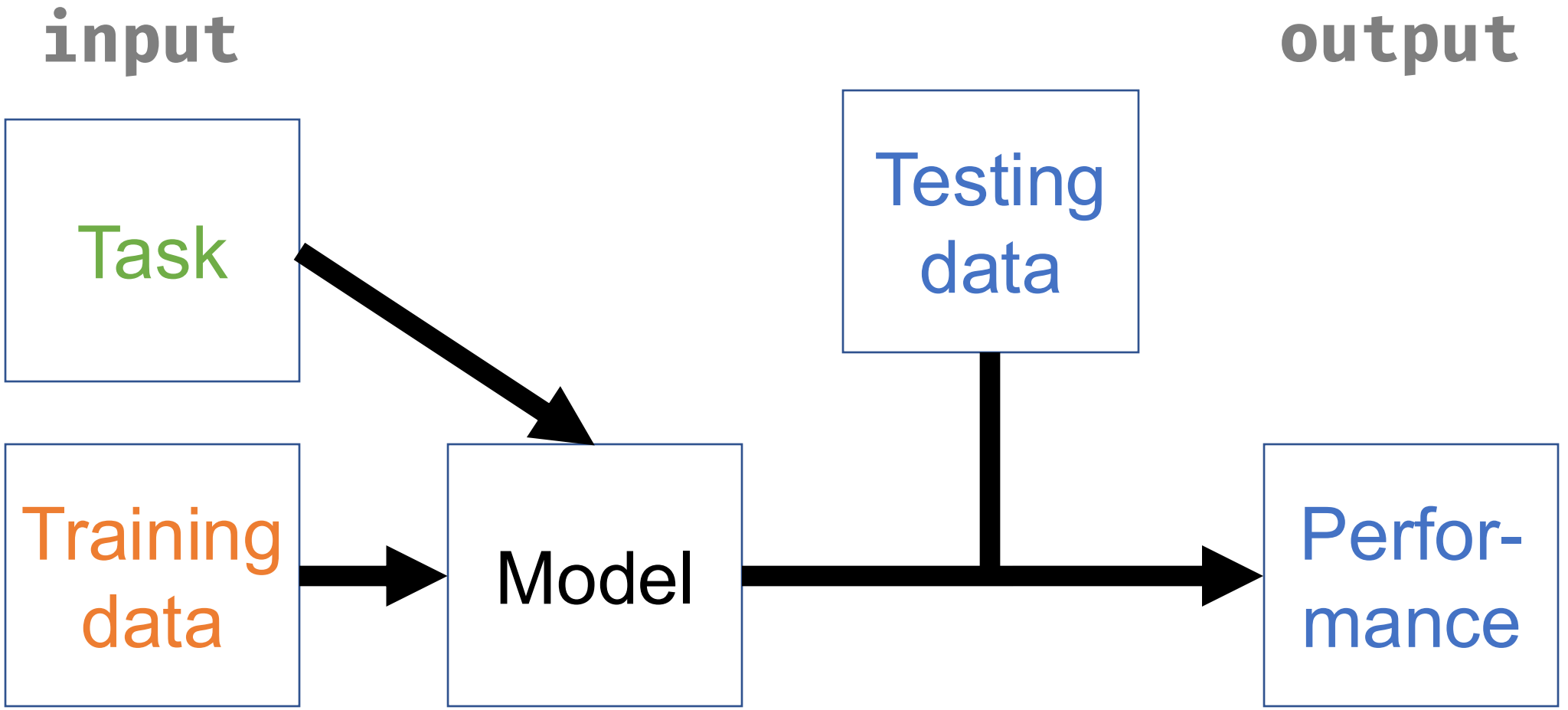
”A computer program is said to learn from **experience** *E* with respect to some class of **tasks** *T* and **performance measure** *P* if its performance at tasks in *T*, as measured by *P*, improves with experience *E*.”

| | *(Loose) translations* |
|---|---|
| **Experience** | Data; Example; Observation |
| **Task** | (Analysis) Purpose; Goal; Aim; Reason; Inference |
| **Performance measure** | Error; Loss; Cost; Risk; |

# Example tasks

1. Identifying customer segments and demographics to help build targeted advertising campaigns
2. Understanding sentiment of Twitter comments as either "positive" or "negative"
3. Identifying financial transactions that are potentially fraudulent
4. Ranking hotels you might like on hotel booking website
5. Recommending a book you might want to buy
6. Detecting brain lesions on an MRI
7. Predicting house prices based on house attributes such as number of bedrooms, location, and size

*What could be the* **experience** *and the* **performance measure** *for each of these?*




**good model.** */gʊd ˈmɒdl/ concept.* **1** (*statistics*) a model with perfect input. **2** (*machine learning*) a model with good output, as measured by the performance metric.

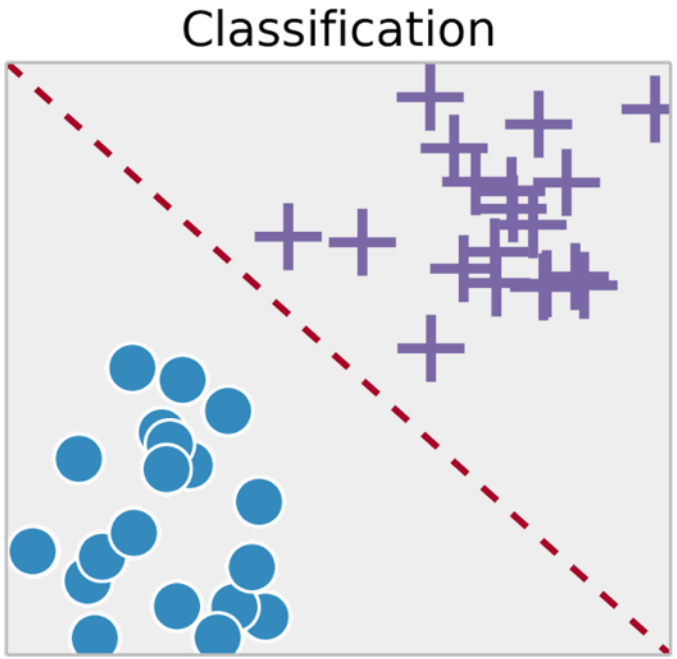
# Types of machine learning

We often distinguish **3 types** of machine learning:

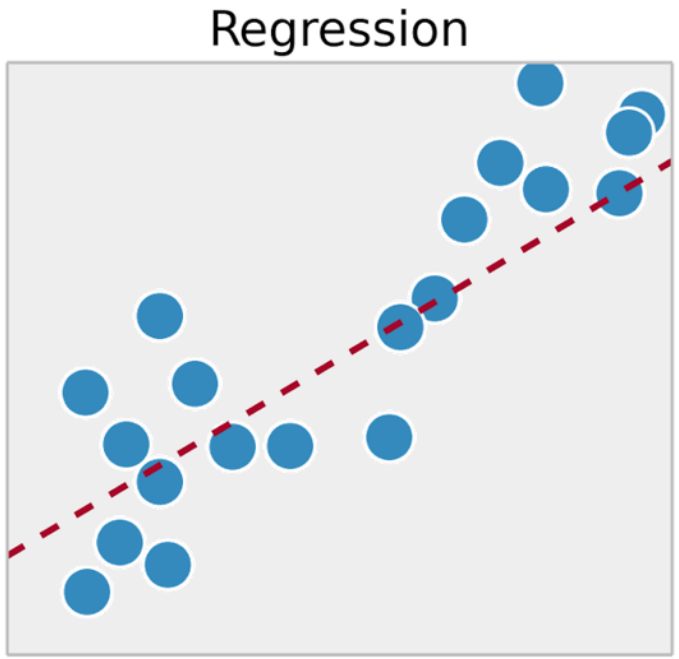
- **Supervised Learning**: learn a model from labeled training data, then make predictions
- **Unsupervised Learning**: explore the structure of the data to extract meaningful information
- **Reinforcement Learning**: develop an agent that improves its performance based on interactions with the environment

# Supervised classification vs. regression

**Classification**: predict a class label (discrete category)

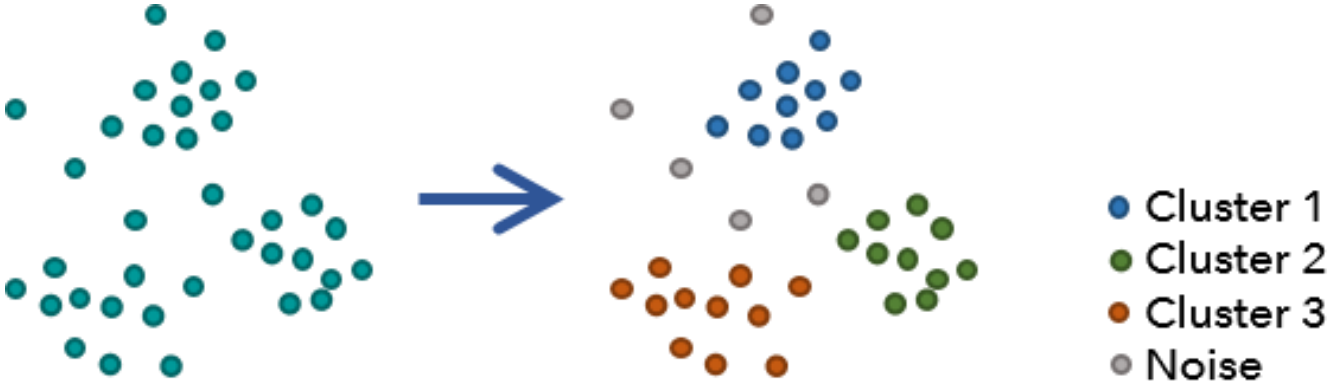



**Regression:**
predict a continuous value



# Unsupervised learning

Learn a model from **unlabeled** training data



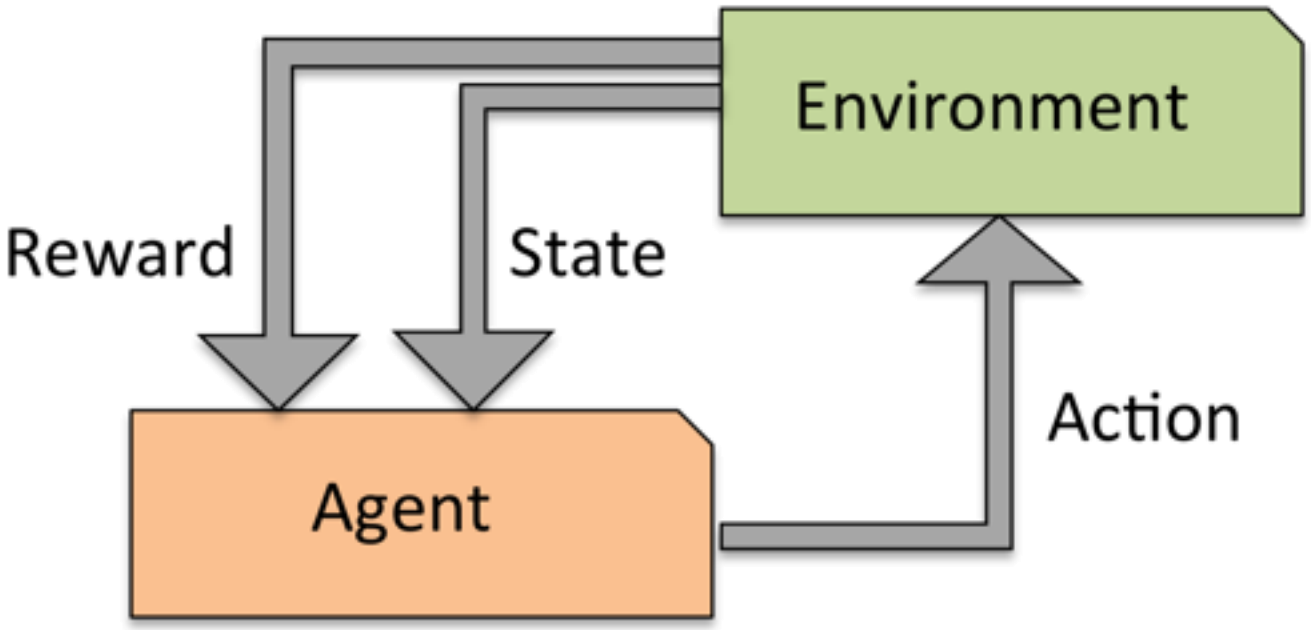

Predict “labels” from unlabeled data, or predict new data

E.g.: PCA, clustering, VAE, GANs, …


*Generate a photo of students in the Master Applied Data Science at Utrecht University.*

# Reinforcement learning
(not in this course)

- Develop agent that improves performance based on interactions with environment
- Example: Chess, Go, ¿healthcare?…
- Reward function defines how well actions work
- Learn actions that maximize reward through exploration



# Regression

# Regression

$$y = f(\boldsymbol{x}) + \epsilon$$

There are usually a bunch of x's. We keep notation legible by saying **x** might be a vector of *p* predictors.

# Regression

$$y = f(\boldsymbol{x}) + \epsilon$$

$y$ : Observed outcome;

$\boldsymbol{x}$ : Observed predictor(s);

$f(\boldsymbol{x})$ : *True** prediction function, to be estimated (so **unknown**);

$\epsilon$ : Unobserved residuals, just *defined* as the "irreducible error",

$$\epsilon = y - f(\boldsymbol{x})$$

The higher the variance of the irreducible error, var($\epsilon$) = $\sigma$, the less we can hope to explain with the data ($\boldsymbol{x}$) at hand.

# Different goals of regression

**Prediction**:

- Given $\boldsymbol{x}$ and $y$, work out $f(\boldsymbol{x})$ as closely as possible.

**Inference**:

- "Is $\boldsymbol{x}$ related to $y$?"
- "How is $\boldsymbol{x}$ related to $y$?"
- "How precise are parameters of $f(\boldsymbol{x})$ estimated from the data?"

Galit Schmueli (2010). *To Explain or to Predict?*. Statistical Science.

# Example regression techniques

*Least-squares based:*

- Linear regression $f(x_i) = b_0 + b_1 x_i$
- Polynomial regression $f(x_i) = b_0 + b_1 x_i + b_2 x_i^2 + b_3 x_i^3$
- Basis functions, Splines $f(x_i) = b_0 + b_1 x_i + b_2 x_i^2 + b_3 x_i^3 + b_4 (x - \xi)_+^3$

*Tree-based:*

- Regression trees
- Random forests
- Gradient boosting

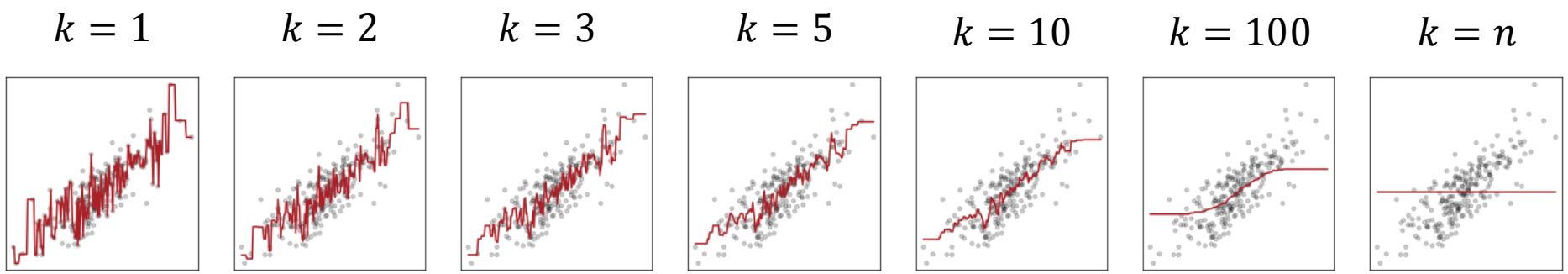
*Other:*

- Neural networks, deep learning, k-NN

# “Nonparametric” regression

- Most modern regression techniques are ”nonparametric”
- **Nonparametric** means the *effective number of parameters increases with the sample size:*
- Model complexity grows with the amount of data
- Simplest example: k-nearest neighbors (ISLR Section 3.5)

- *Effective number of parameters for k-NN is* $\frac{n}{k}$



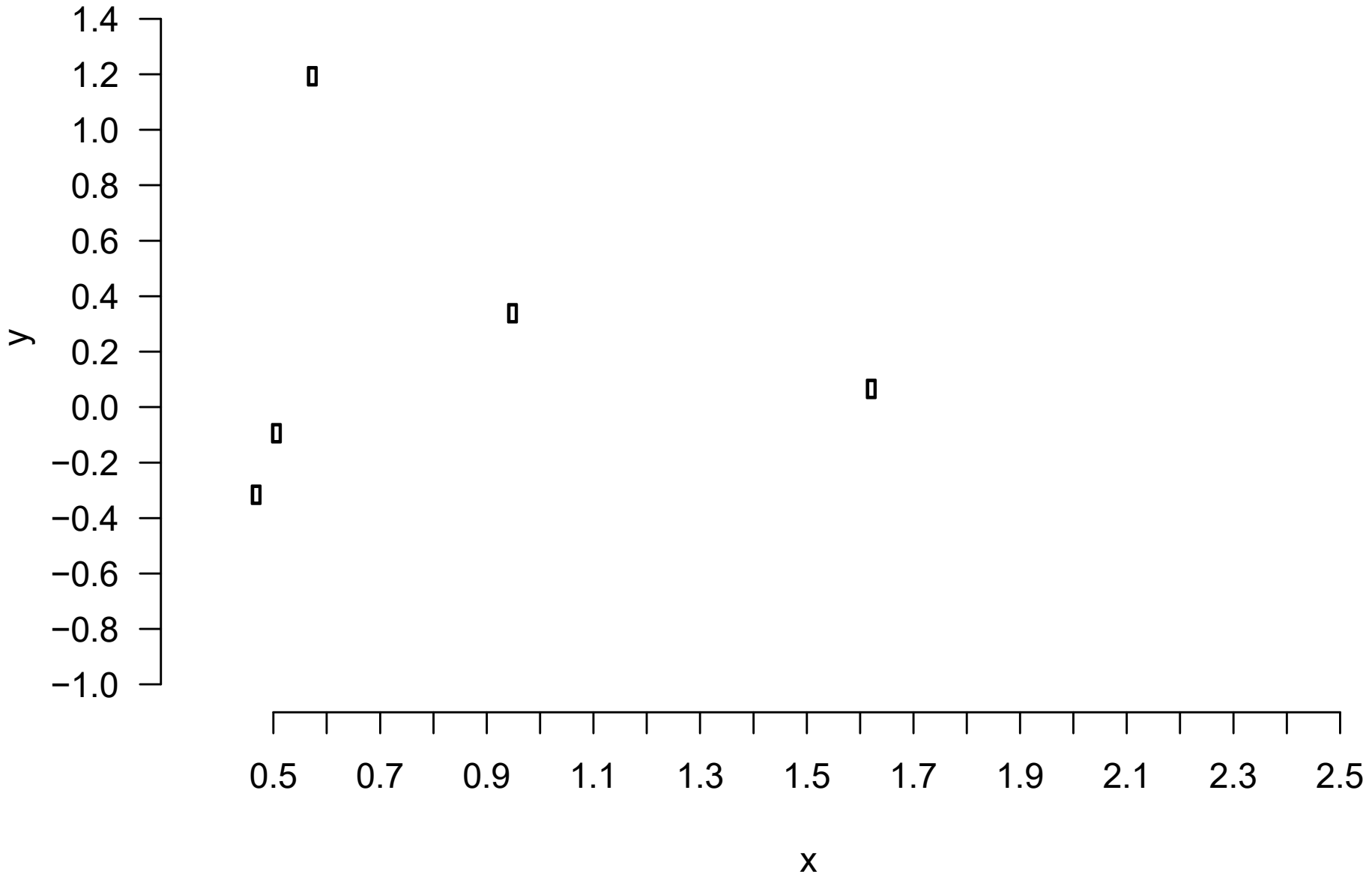
# More nonparametrics.

- k-NN is not so great at controlling complexity when the number of features is more than ~5.
- Hence the rest of machine learning.

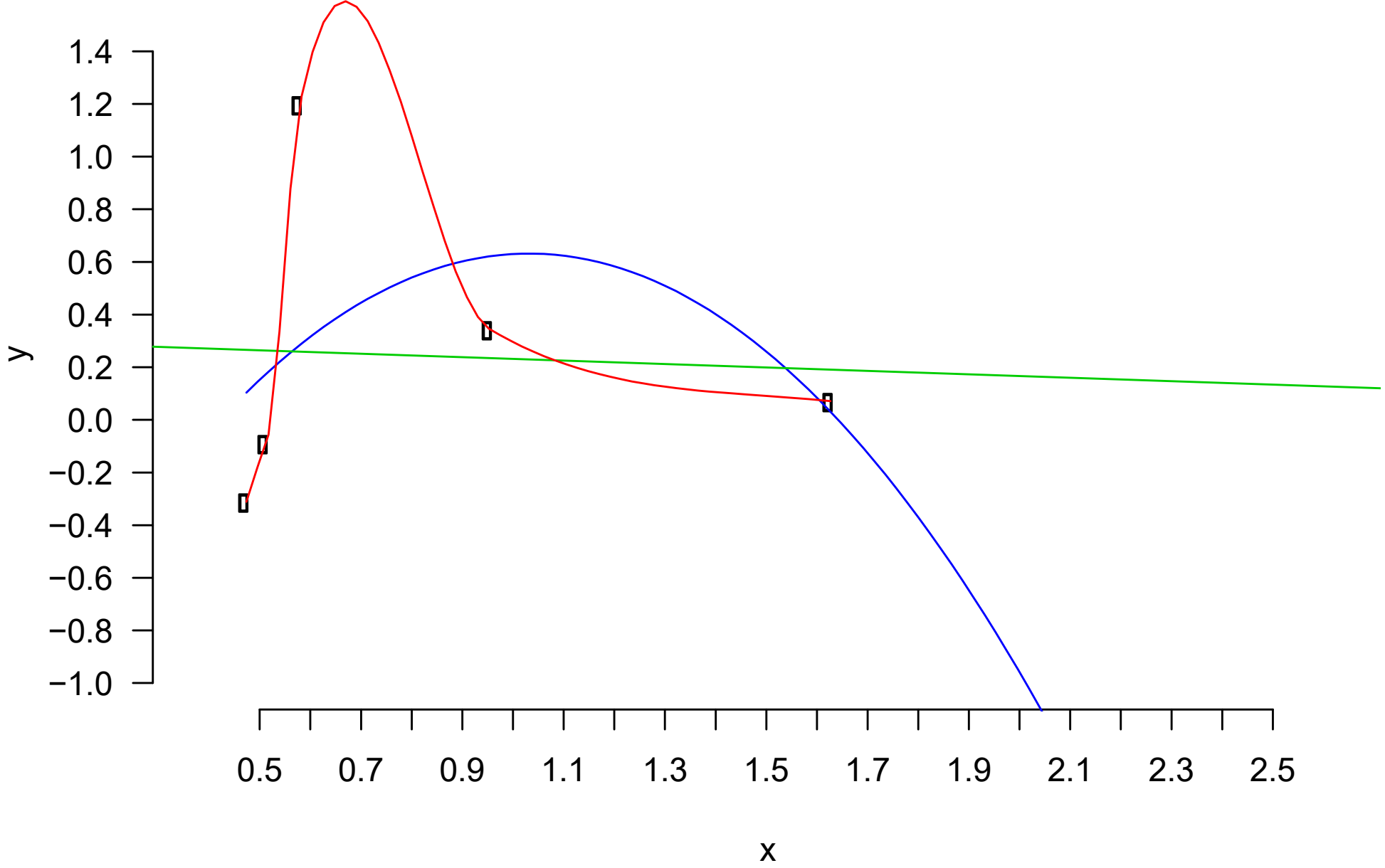
| **“Bottom-up”: complexity increases with data** | **“Top-down”: complexity decreases down to data** |
|---|---|
| K-nearest neighbors | Random forests |
| Kernel regression | Deep learning / Neural nets / Transformers |
| Support vector regression | |
| Boosting | |
| Gaussian process regression | |

# An exercise in prediction

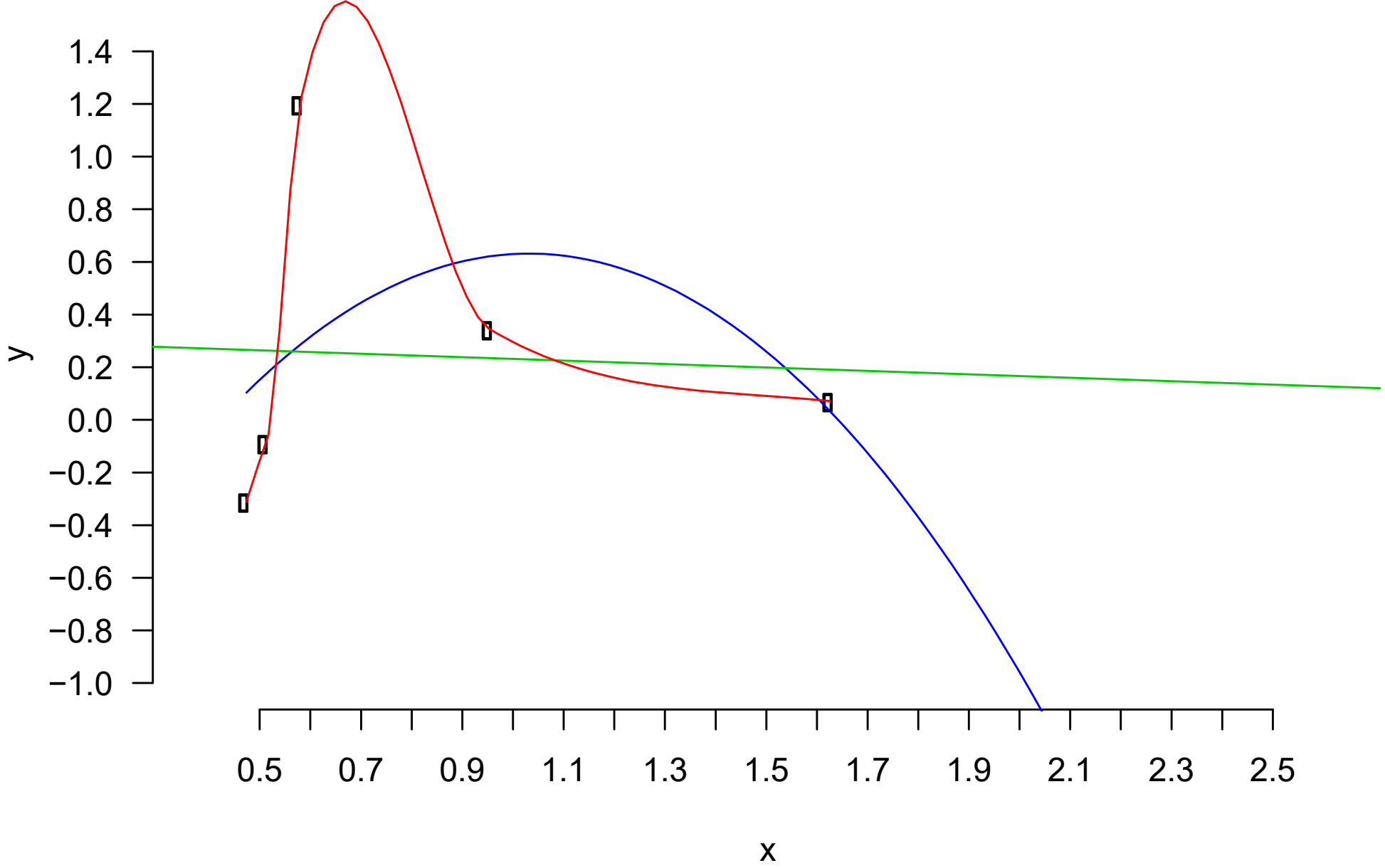
- I am going to show you a data set, and we (i.e. you) are going to try to estimate $\hat{f}(x)$ and predict $y$;
- I generated this data set myself using R, so I know the true $f(x)$ and distribution of $\epsilon$.



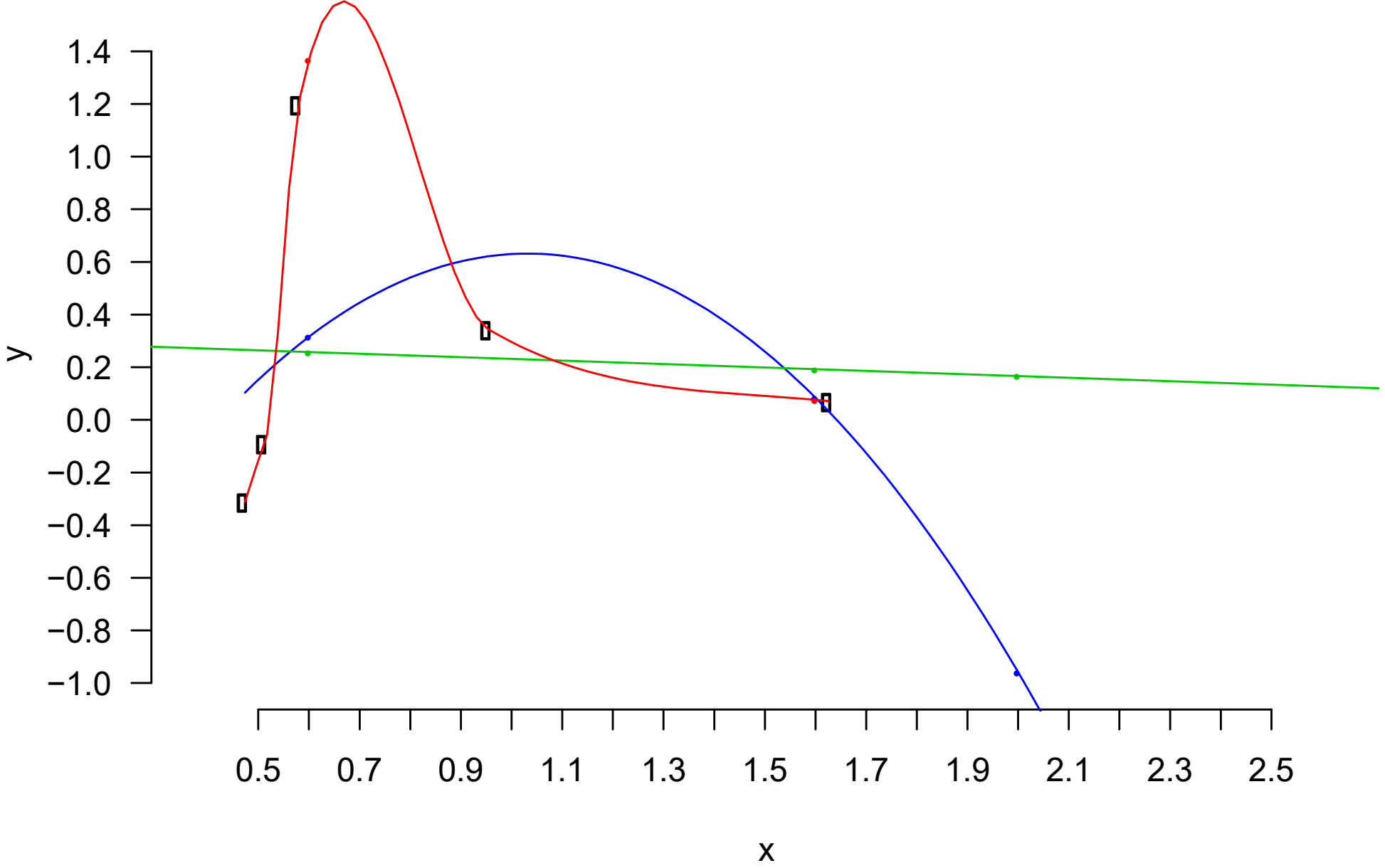

**Task**: predict (however you want): $\hat{y}$ for $x = 0.6, 1.6, 2.0$.



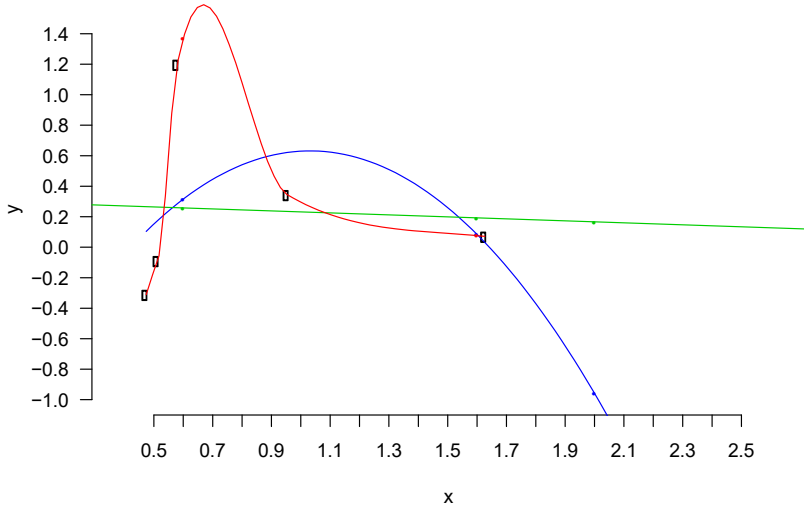

**Task**: predict (however you want): $\hat{y}$ for $x = 0.6, 1.6, 2.0$.




**Task**: predict (however you want): $\hat{y}$ for $x = 0.6, 1.6, 2.0$.




**Task**: predict (however you want): $\hat{y}$ for $x = 0.6, 1.6, 2.0$.




| **Model** | $\hat{f}(0.6)$ | $\hat{f}(1.6)$ | $\hat{f}(2.0)$ |
|---|---|---|---|
| Eyeballing | ? | ? | ? |
| Linear regression | 0.192 | 0.192 | 0.166 |
| Linear regression w/ quadratic | 0.315 | 0.084 | -0.959 |
| Nonparametric | 1.368 | 0.076 | - |

# Model performance

# Model performance

- The predictions $\hat{y} = \hat{f}(x)$ differ from the true $f\ (x)$;
- We can evaluate how much this happens “on average”.

# A few model evaluation metrics

- Mean squared error (MSE):

$$\text{MSE} = n^{-1} \sum_{i=1}^{n} (y - \hat{y})^2$$

- Root mean squared error RMSE $= \sqrt{\text{MSE}}$
- Mean absolute error (MAE:)

$$\text{MAE} = n^{-1} \sum_{i=1}^{n} |y - \hat{y}|$$

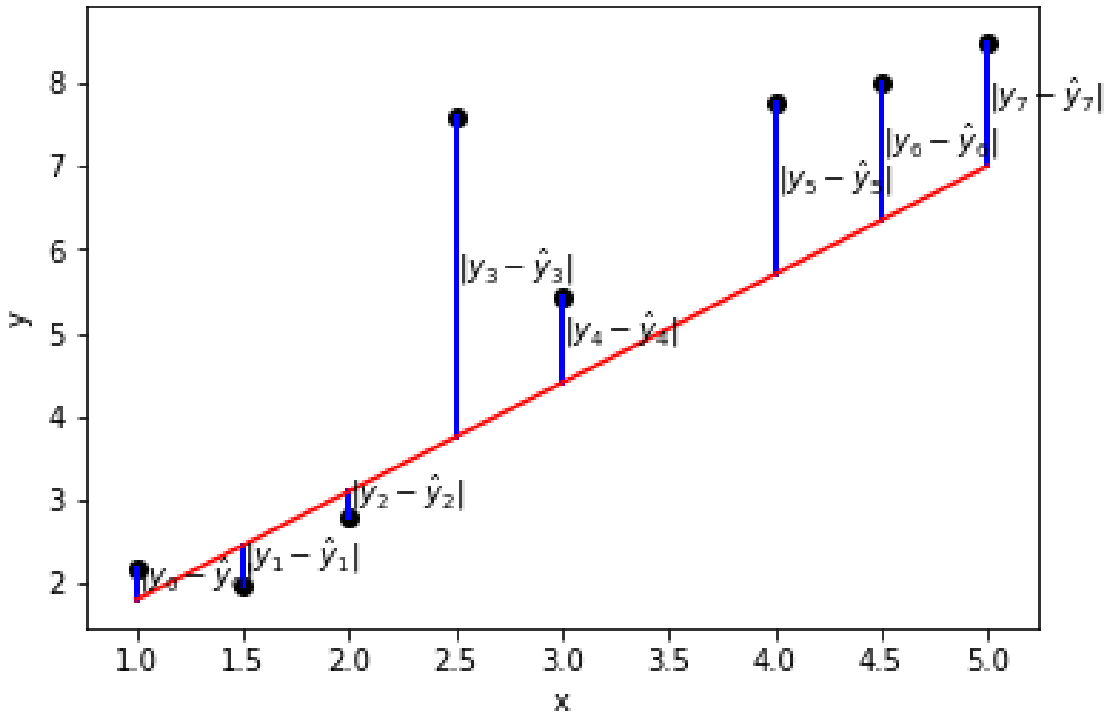
- Median absolute error (mAE):

$$\text{mAE} = \text{median}|y - \hat{y}|$$

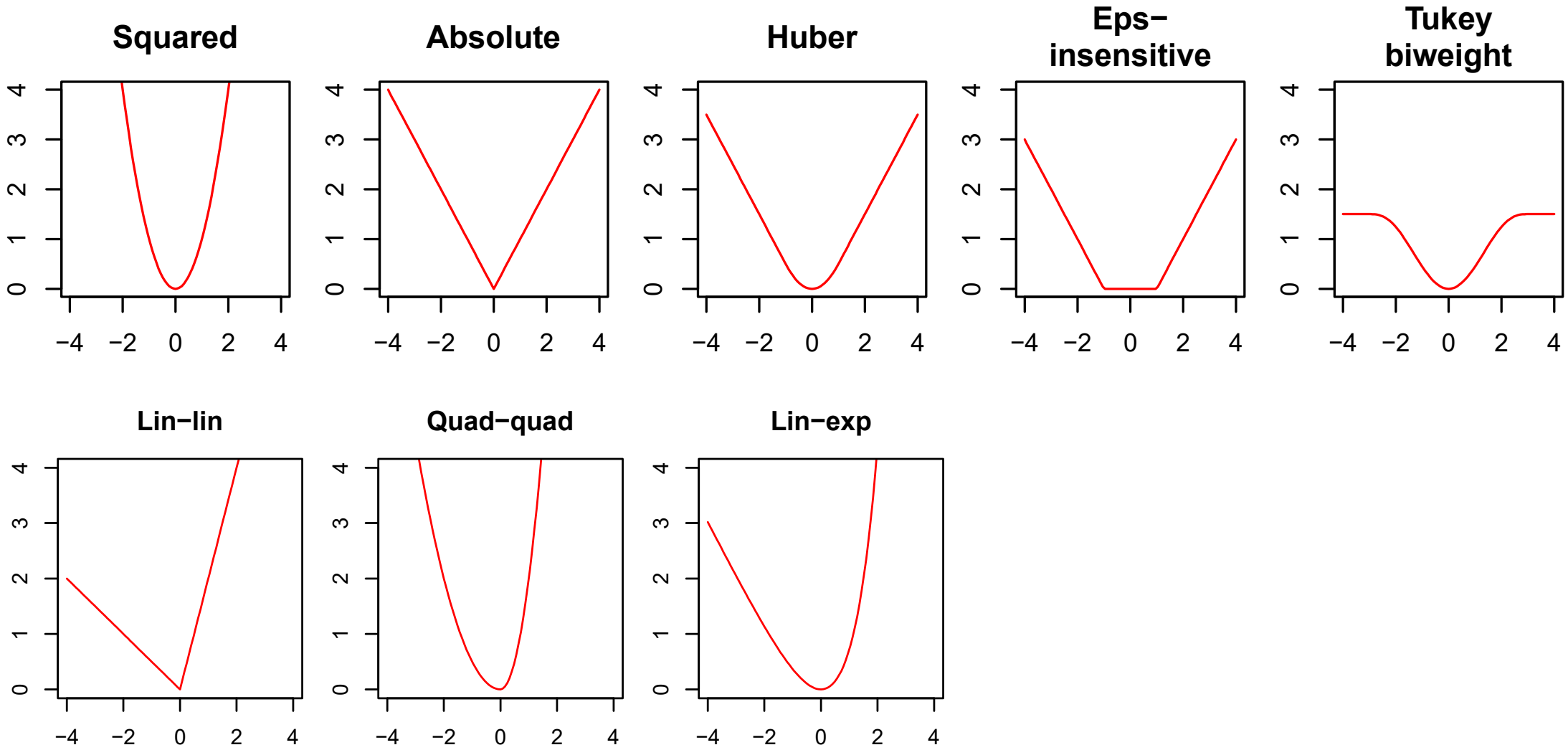
- Proportion of variance explained: ($R^2$)

$$R^2 = \text{correlation}(y, \hat{y})^2$$

- etc...



# Thinking about loss functions



# So, who won…?

- Which model appears to fit best to the training data?
- Calculate MSE for one model, relative to truth.
- Which is the best model in terms of MSE?

# And the winner is…

| **Model** | MSE | MSE (interpolation) |
|---|---|---|
| Eyeballing | ? | ? |
| Linear regression | 0.992 | 0.709 |
| Linear regression w/ quadratic | 2.410 | 0.883 |
| Nonparametric | - | 0.883 |

**Truth**: $y = \sqrt{x} + \epsilon$, where $\epsilon \sim \text{Normal}(0,1)$.

# What happened?

- There were few observations, relative to the complexity of most models (except perhaps linear regression);
- The observed data *were* a random sample from the true "data-generating process", $f(x) + \epsilon$;

BUT

- By chance, patterns appeared that are *not* in the true $f(x)$;
- The more flexible models $\hat{f}(x)$ **overfitted** these patterns.

# Thought experiment

- Imagine we had sampled another 5 observations, re-trained all of our models, and predicted again.
- Each time we remember the predictions given.
- We do this a large number of (say, 1,000,000,000) times, and then take the average for the predictions over all samples

**Questions**

1. Which model(s) would give the right prediction **on average**?
2. Which model(s) would give wildly varying predictions?
3. Which model(s) would you *guess* have the lowest MSE overall?

## Unbiased

*Model that gives the correct prediction, on average over samples from the target population*

- Unbiased in our example: nonparametric, square-root
- Biased in our example: all others

## High variance:

*Model that easily overfits accidental patterns.*

- High variance in our example: nonparam., quadratic, sq-root
- Low variance in our example: linear regression

# Bias and variance in our example

Some, possibly surprising, conclusions:

- The **best** model in one sense is the **worst** model in the other!
- The “correct” model is not the best model!

# Bias-variance tradeoff

- Flexibility → less bias
- Flexibility → more variance

*Bias and variance are implicitly linked because they are both affected by* **model complexity** (“flexibility”, “capacity”, “effective number of parameters”)

# What do you mean by “complexity”?

- Amount of information in data absorbed into model;
- Amount of compression performed on data by model;
- Effective number of parameters

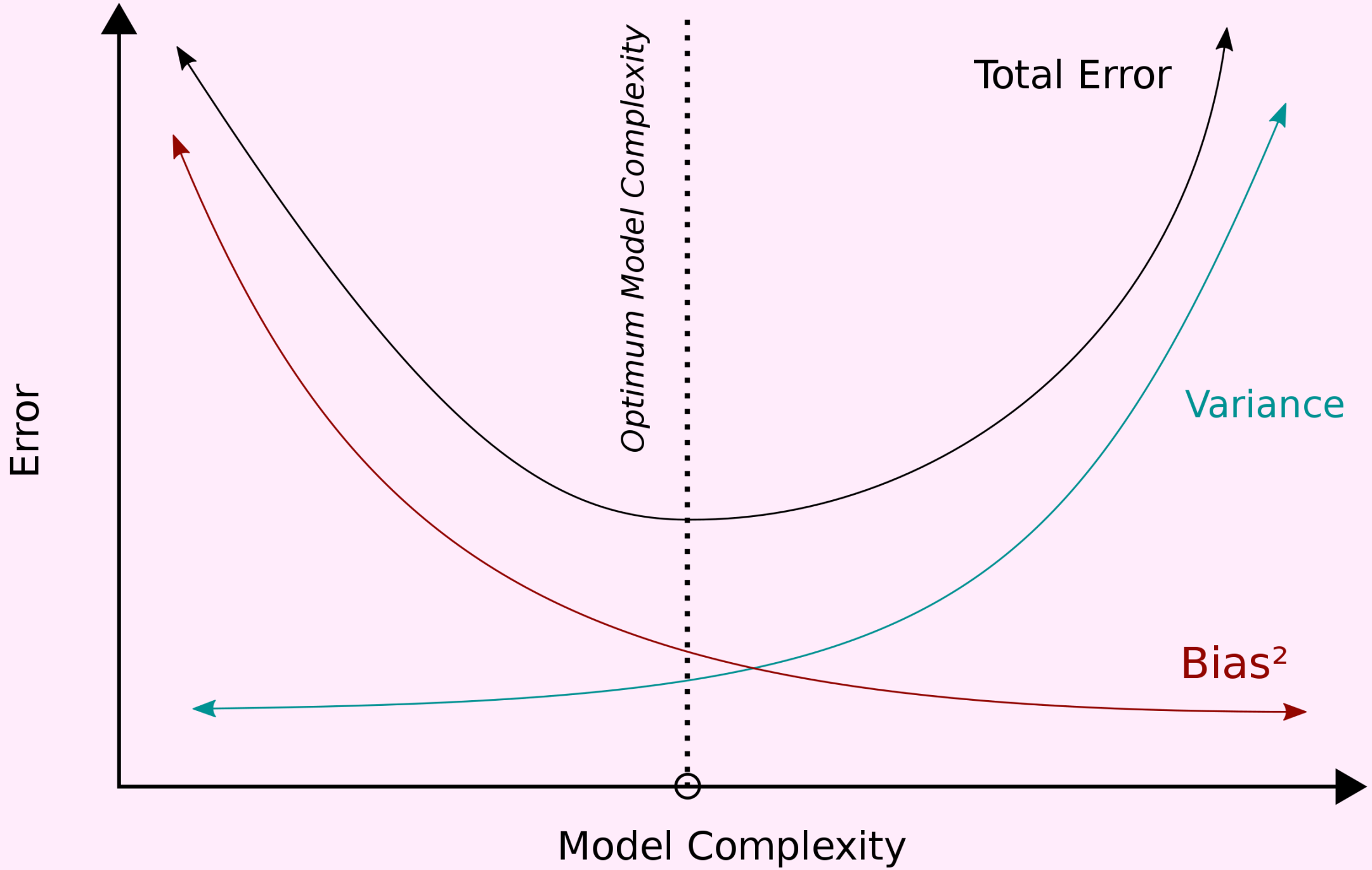
Examples of things that make model **more “complex”**:

- More predictors in linear regression
- Higher-order polynomial in linear regression ($x^2, x^3, x^4$, etc.);
- Smaller k in kNN
- ...

**Question**:

Does the bias-variance tradeoff occur with n = 5?

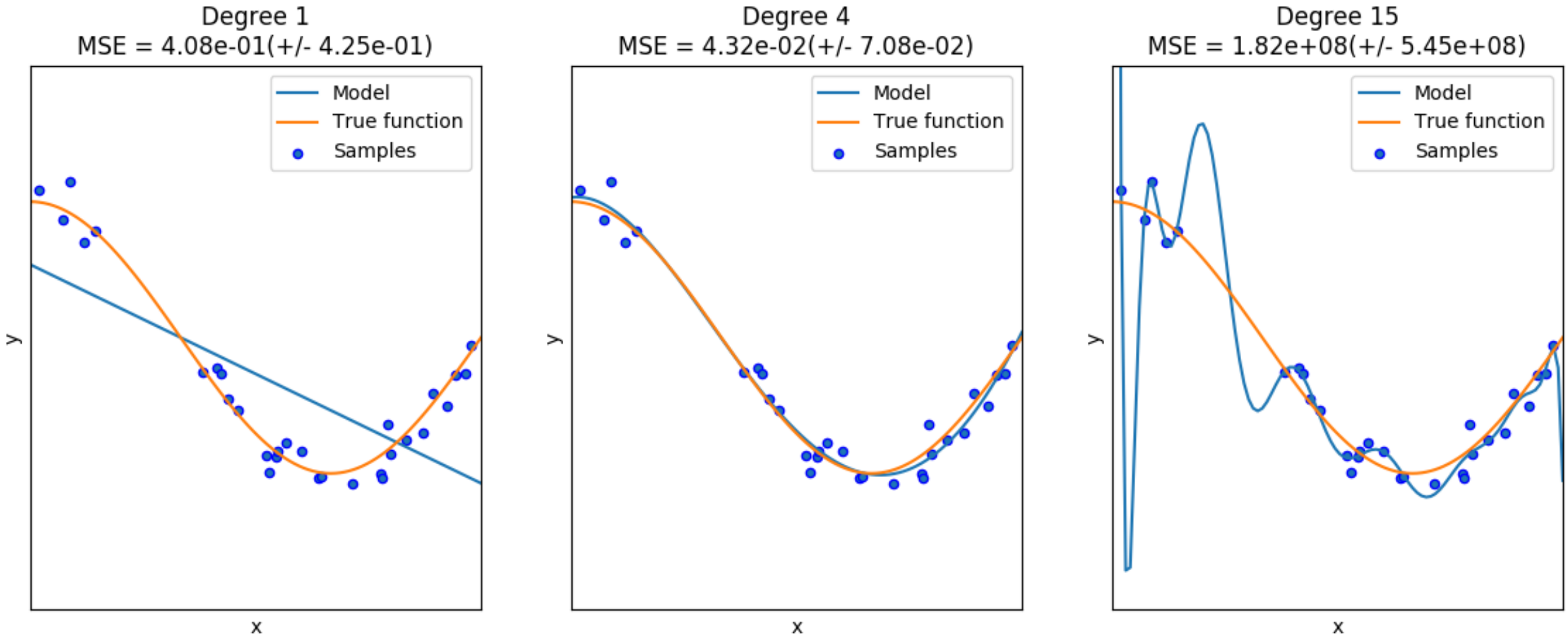
Does the bias-variance tradeoff occur with n = 5,000,000,000?


Error
Optimum Model Complexity
Total Error
Variance
Bias²
Model Complexity

$$E(\text{MSE}) = \text{Bias}^2 + \text{Variance} + \sigma$$

Population mean squared error is squared bias PLUS model variance PLUS irreducible variance.

(The E means “on average over samples from the target population”).


Degree 1
MSE = 4.08e-01(+/- 4.25e-01)
Model
True function
Samples
y
x
Degree 4
MSE = 4.32e-02(+/- 7.08e-02)
Model
True function
Samples
y
x
Degree 15
MSE = 1.82e+08(+/- 5.45e+08)
Model
True function
Samples
y
x

# Back to reality!

**Problem**:

- Wait, we don’t actually have the population!
- And our training data were already used to train the model…

**Solution**:

- Instead, we will take a new, pristine, sample from population:
- The **test data**

# The train-val-test paradigm

# Train/dev/test

**Training data:**

Observations used to train ("fit", "estimate") $\hat{f}(\boldsymbol{x})$

**Validation data** (or "dev" data):

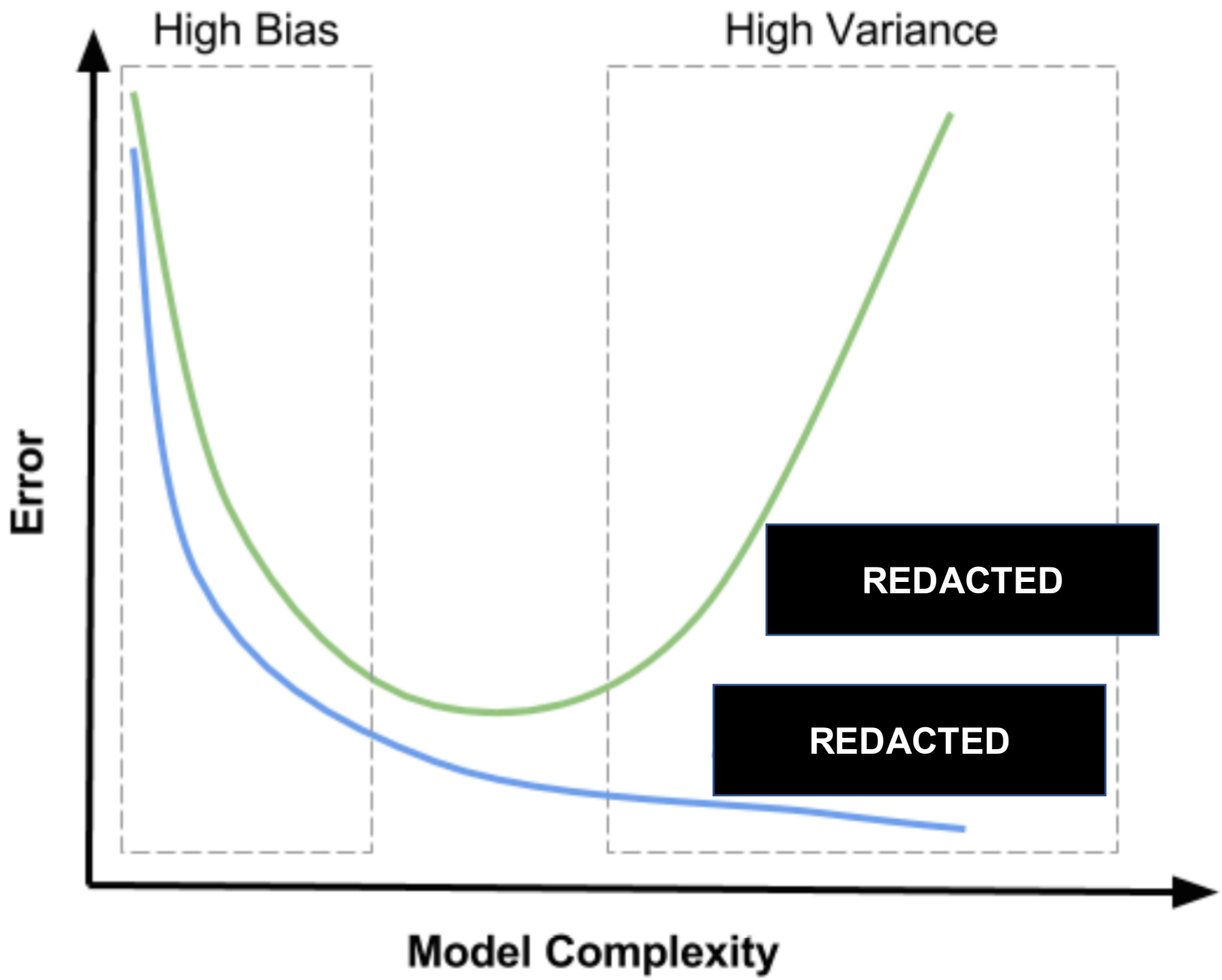
New observations from the same source as training data

Used several times to select model complexity)

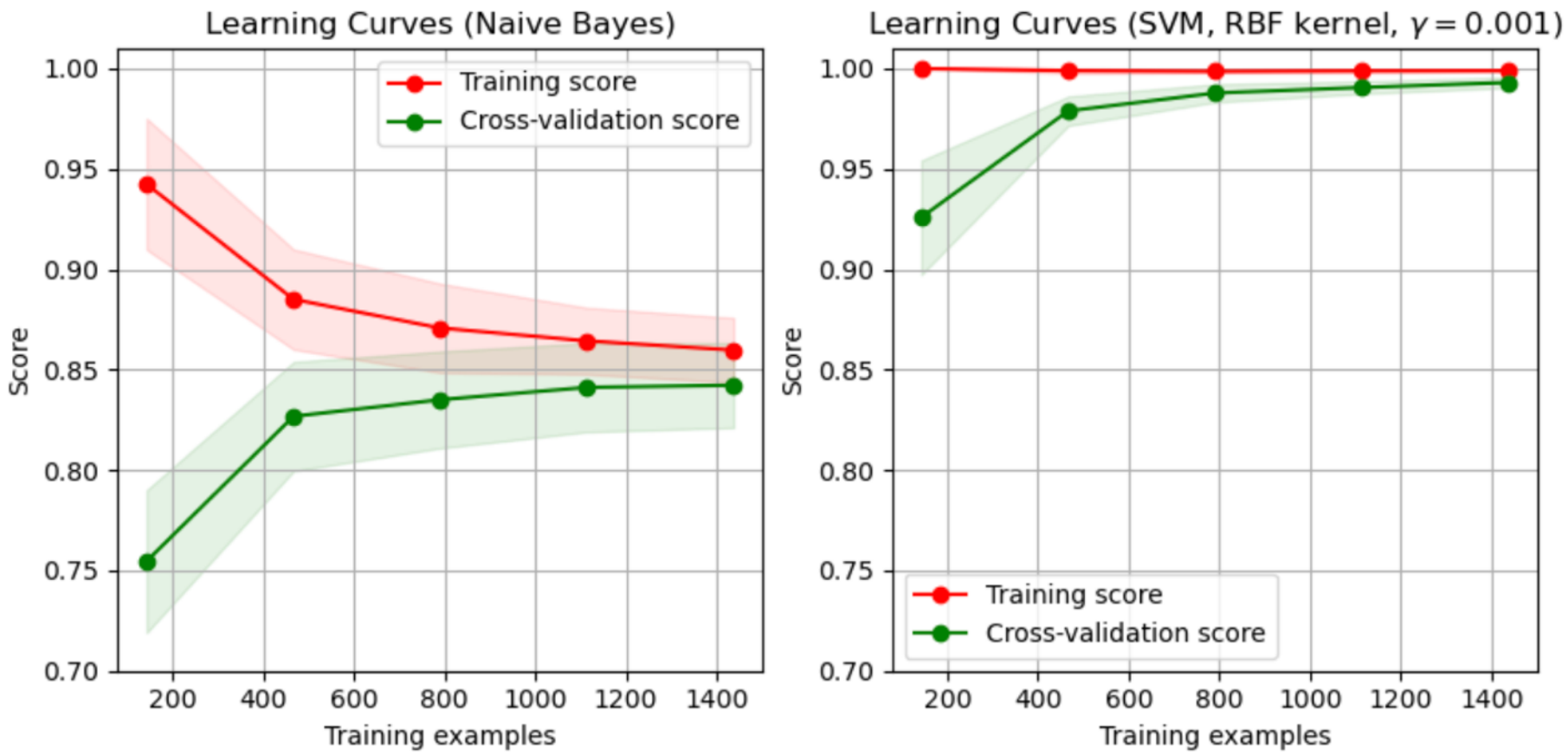
**Test data:**

New observations from the intended prediction situation

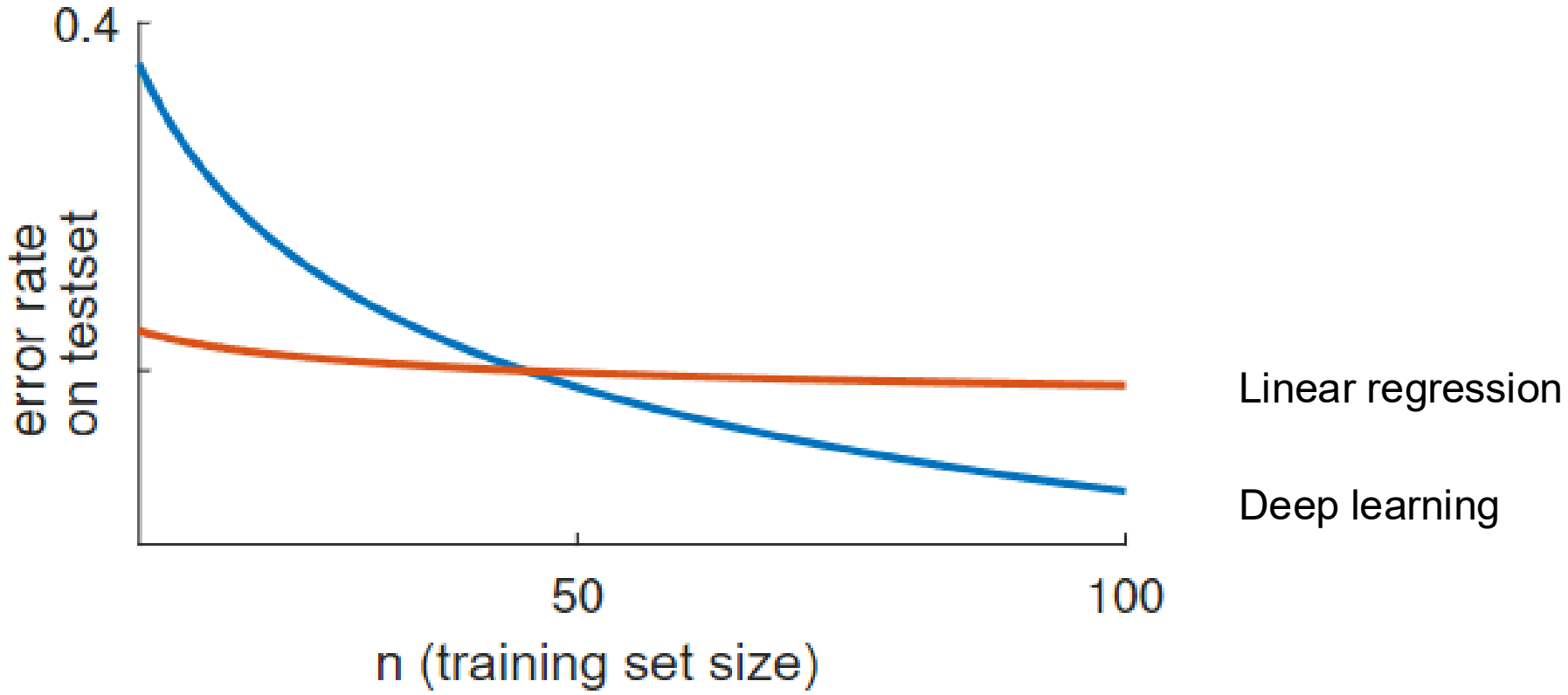
*Question: Why don't these give the same average MSE?*


High Bias
High Variance
Error
REDACTED
REDACTED
Model Complexity

# **Learning curves:** *n* vs. performance



# Learning curves




Viering & Loog (2021). *The Shape of Learning Curves: a Review*. https://arxiv.org/pdf/2103.10948.pdf

# Conclusion

- Choose your **data**, **goal**, and **performance metric** with care
- Bias and variance trade off, in **theory** and **practice**;
- We try to estimate this using **train/dev/test** paradigm;
- Getting good test data is difficult problem;
- **Cross-validation** is a useful alternative to separate dev set;
- Beware that *any* procedure that makes decisions based on the data requires validation!
- **Machine learning is not (only) about models/algorithms. It is just as much about data, the reality of your task, etc.**