The Mathematics of Machine Learning 2026

Lecture 1

Tim van Erven

University of Amsterdam

Organization

Lectures: Weekly lectures on Tuesdays:

- Weeks 6–12 from 9h00–11h00 in room SP G2.10
- Weeks 14–17, 20, 21 from 11h00–13h00 in room SP C1.112

Exercise Classes: Weekly exercise classes on Thursdays, starting in the second week of the course.

- Week 7:
  - Group B: from 11h00-13h00 in room SP B0.207
  - Group A: from 15h00-17h00 in room SP D1.110
- Weeks 8-21:
  - Groups A and B merged into single group.
  - Irregular schedule: see your UvA student schedule!

Teaching Assistants:

- David van Batenburg (before midterm)
- Federico Berton (after midterm)

Website: www.timvanerven.nl/teaching/mathml2026/
Canvas: canvas.uva.nl/courses/49659

Outline

1. Organization
2. General Introduction
    a) Supervised learning: classification, regression (overfitting 1)
    b) Simplistic: linear regression for classification (overfitting 2)
    c) Nearest neighbor classification (overfitting 3)

Supervised Learning

Training data: $T = \begin{pmatrix} Y_1 \\ X_1 \end{pmatrix}, \ldots, \begin{pmatrix} Y_N \\ X_N \end{pmatrix}$

Teacher shows us what desired response Y_i is for input X_i.

X_i: feature vectors
Y_i: class / response variable

Goal: learn function $\hat{f} : \mathcal{X} \to \mathcal{Y}$ from **model** $\mathcal{F}$ = some set of functions

Supervised Learning

Training data: $T = \begin{pmatrix} Y_1 \\ X_1 \end{pmatrix}, \ldots, \begin{pmatrix} Y_N \\ X_N \end{pmatrix}$

Teacher shows us what desired response Y_i is for input X_i.

X_i: feature vectors
Y_i: class / response variable

Goal: learn function $\hat{f} : \mathcal{X} \to \mathcal{Y}$ from **model** $\mathcal{F}$ = some set of functions

Evaluate $\hat{f}$ on **test data**:

- new X from same source
- predict corresponding Y by $\hat{Y} = \hat{f}(X)$

Supervised Learning

Training data: $T = \begin{pmatrix} Y_1 \\ X_1 \end{pmatrix}, \ldots, \begin{pmatrix} Y_N \\ X_N \end{pmatrix}$

Teacher shows us what desired response Y_i is for input X_i.

X_i: feature vectors
Y_i: class / response variable

Goal: learn function $\hat{f} : \mathcal{X} \to \mathcal{Y}$ from **model** $\mathcal{F}$ = some set of functions

Evaluate $\hat{f}$ on **test data**:

- new X from same source
- predict corresponding Y by $\hat{Y} = \hat{f}(X)$

Assume:

- (X_i, Y_i) pairs are independent samples from same probability distribution P^*.
- Avoid further assumptions on P^*. Can be very complicated!

Supervised Learning: Classification

$$\begin{pmatrix} G_1 \\ X_1 \end{pmatrix}, \begin{pmatrix} G_2 \\ X_2 \end{pmatrix}, \ldots, \begin{pmatrix} G_N \\ X_N \end{pmatrix}$$

Classification: G is a categorical variable

- E.g. yes/no or apple/orange/pear

Supervised Learning: Classification

$$\begin{pmatrix} G_1 \\ X_1 \end{pmatrix}, \begin{pmatrix} G_2 \\ X_2 \end{pmatrix}, \ldots, \begin{pmatrix} G_N \\ X_N \end{pmatrix}$$

Classification: G is a categorical variable

- E.g. yes/no or apple/orange/pear

$G \to Y$:

yes $\to 1$	yes $\to +1$
no $\to 0$	no $\to -1$

---

apple $\to 1$	apple $\to \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$	
orange $\to 2$	orange $\to \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$	"dummy coding" "one-hot coding"
pear $\to 3$	pear $\to \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$	

Spam Classification

Get Messages | Write | Chat | Address Book | Tag | Quick Filter | Search... <⌘K>

From Google Corporation© <claudio.santoriello@finanzaefuturo.it>
Subject **Powered by Google**
Reply to Google Corporation© <mr.jonesbradley@foxmail.com>
To

Reply | Forward | Junk | Delete | More

08/10/15 15:39

Dear Google User,

You have been selected as a Google Ambassador for using Google services. Find attached letter for more details and Processing of your claims.

Best Regards,

Matt Brittin
Chairman of the Board and Managing Director,
Google United Kingdom

1 attachment: Google Notification Letter.pdf 367 KB

Save

?

[1]

[1] Spam can image by Bodo Akdeniz from https://commons.wikimedia.org/wiki/File:Spam_2.jpg

Spam Classification

Y_i : 'spam' or 'no spam'

X_i : e-mails (pre-processed)

- **Examples for $\mathcal{F}$**:
  - Trivial: $\mathcal{F} = \{$always say 'spam', always say 'no spam'$\}$
  - $\mathcal{F} =$ all functions of the form

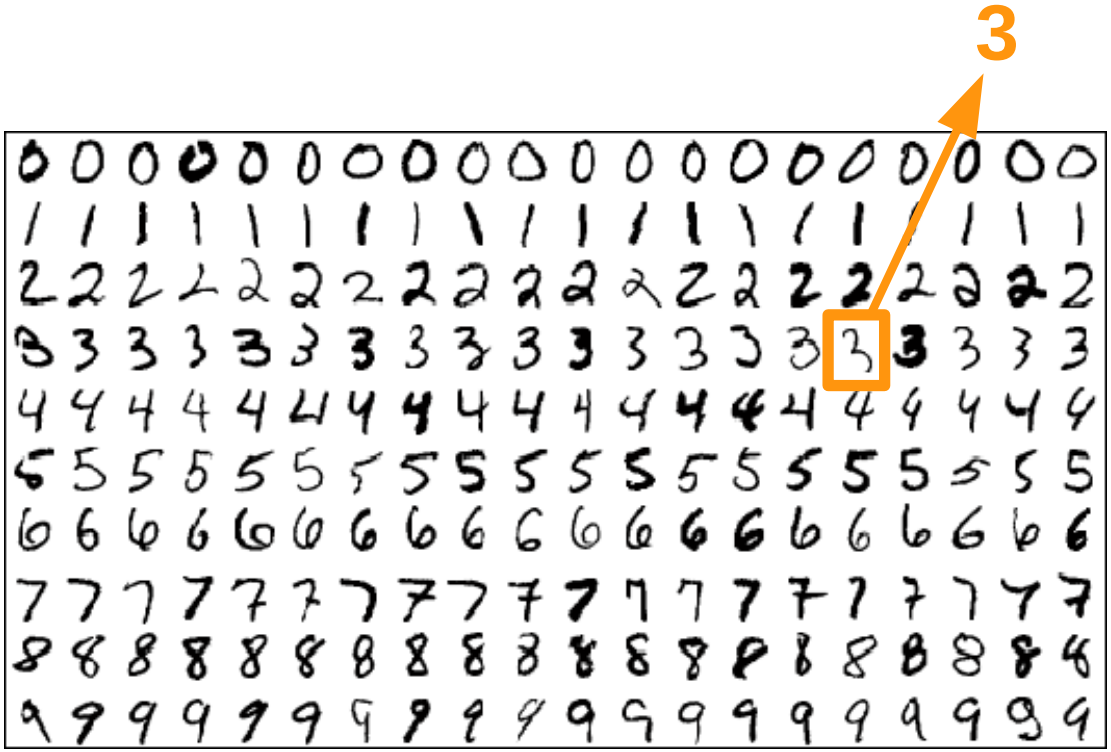
$$\text{IF } \%(\text{word 1}) > \theta_1 \text{ OR } \%(\text{word 2}) < \theta_2$$
$$\text{THEN } \text{'spam'}; \text{ ELSE } \text{'no spam'}$$

- **Empirical Risk Minimization (ERM)**:
  - Pick $\hat{f} \in \mathcal{F}$ with smallest nr. of mistakes on $\mathcal{T}$

Handwritten Digit Classification

$Y_i : 0, 1, \ldots, 9$

X_i : picture of one digit $= k \times k$ matrix of grey values



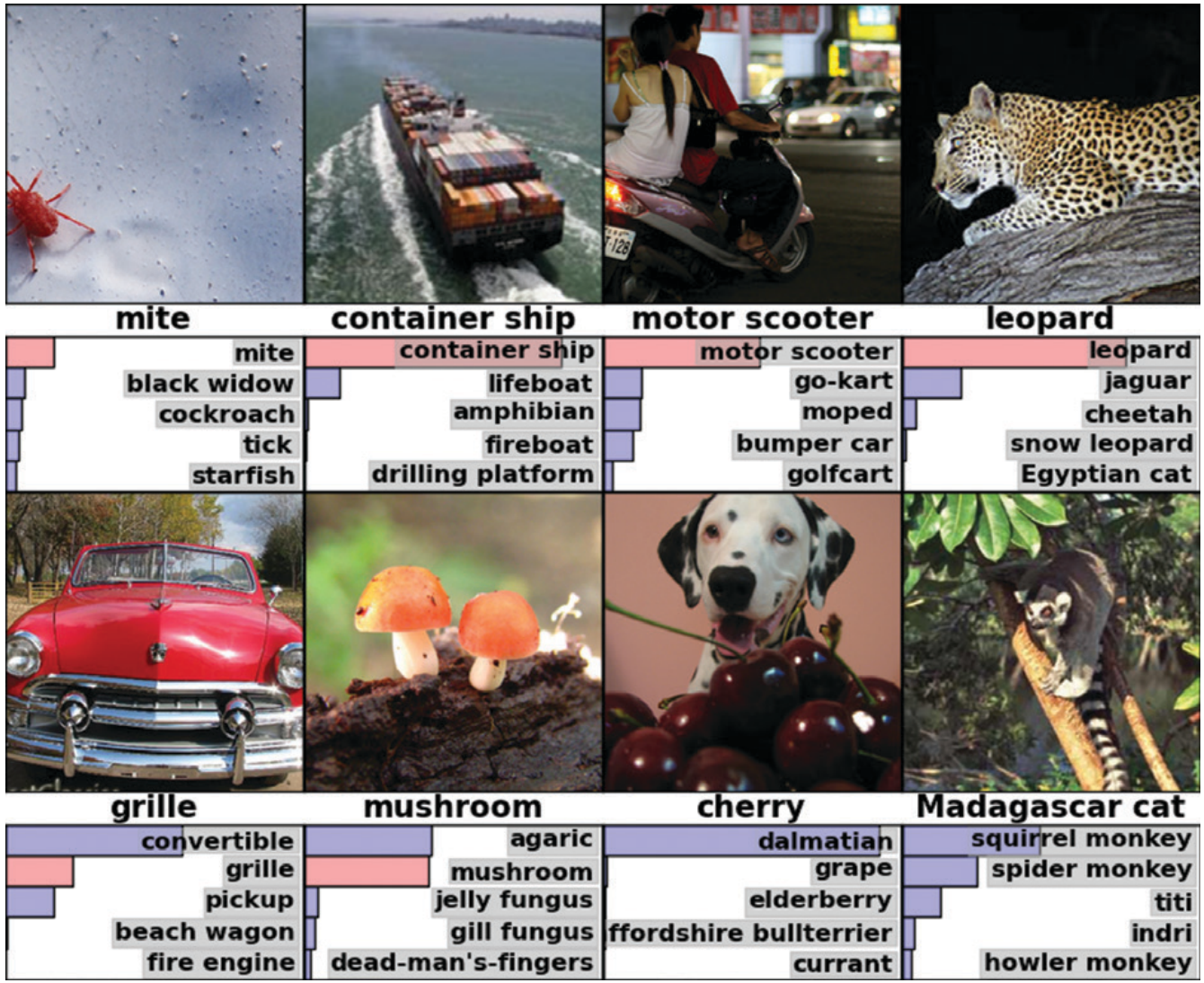
- **Nearest Neighbour**:
  - Measure distance of new picture to all pictures in $\mathcal{T}$
  - Choose same class as closest picture

LaTeX Symbol Classification

Detexify website: `detexify.kirelabs.org/`

What is Y_i?
What is X_i?

Imagenet Classification



Krizhevsky, Sutskever, Hinton, **ImageNet Classification with Deep Convolutional Neural Networks**, NeurIPS 2012

Supervised Learning: Linear Regression

$$\begin{pmatrix} Y_1 \\ X_1 \end{pmatrix}, \begin{pmatrix} Y_2 \\ X_2 \end{pmatrix}, \ldots, \begin{pmatrix} Y_N \\ X_N \end{pmatrix}$$

Linear Regression:

- Y_i is real-valued
- $X_i \in \mathbb{R}^p$ (convert any categorical feature $X_{i,j}$ by one-hot coding)
- Model $\mathcal{F}$ can be represented as linear functions:

$$f_\beta(X_i) = \beta_0 + \sum_{j=1}^{p} X_{i,j}\beta_j \qquad \beta = (\beta_0, \beta_1, \ldots, \beta_p)$$

Supervised Learning: Linear Regression

$$\begin{pmatrix} Y_1 \\ X_1 \end{pmatrix}, \begin{pmatrix} Y_2 \\ X_2 \end{pmatrix}, \ldots, \begin{pmatrix} Y_N \\ X_N \end{pmatrix}$$

Linear Regression:

- Y_i is real-valued
- $X_i \in \mathbb{R}^p$ (convert any categorical feature $X_{i,j}$ by one-hot coding)
- Model $\mathcal{F}$ can be represented as linear functions:

$$f_\beta(X_i) = \beta_0 + \sum_{j=1}^{p} X_{i,j}\beta_j \qquad \beta = (\beta_0, \beta_1, \ldots, \beta_p)$$

- Trick: Extend X_i by setting $X_{i,0} = 1$.
  Then simple notation: $f_\beta(X_i) = X_i^\intercal \beta$.

Supervised Learning: Linear Regression

$$\begin{pmatrix} Y_1 \\ X_1 \end{pmatrix}, \begin{pmatrix} Y_2 \\ X_2 \end{pmatrix}, \ldots, \begin{pmatrix} Y_N \\ X_N \end{pmatrix}$$

Linear Regression:

- Y_i is real-valued
- $X_i \in \mathbb{R}^p$ (convert any categorical feature $X_{i,j}$ by one-hot coding)
- Model $\mathcal{F}$ can be represented as linear functions:

$$f_\beta(X_i) = \beta_0 + \sum_{j=1}^{p} X_{i,j}\beta_j \qquad \beta = (\beta_0, \beta_1, \ldots, \beta_p)$$

  - Trick: Extend X_i by setting $X_{i,0} = 1$.
    Then simple notation: $f_\beta(X_i) = X_i^\mathsf{T}\beta$.

NB **Random design:** X_i are random samples from P^*, **not** set by experimenter

Supervised Learning: Linear Regression

$$\begin{pmatrix} Y_1 \\ X_1 \end{pmatrix}, \begin{pmatrix} Y_2 \\ X_2 \end{pmatrix}, \ldots, \begin{pmatrix} Y_N \\ X_N \end{pmatrix}$$

Linear Regression:

- Y_i is real-valued
- $X_i \in \mathbb{R}^p$ (convert any categorical feature $X_{i,j}$ by one-hot coding)
- Model $\mathcal{F}$ can be represented as linear functions:

$$f_\beta(X_i) = \beta_0 + \sum_{j=1}^{p} X_{i,j}\beta_j \qquad \beta = (\beta_0, \beta_1, \ldots, \beta_p)$$

- Trick: Extend X_i by setting $X_{i,0} = 1$.
  Then simple notation: $f_\beta(X_i) = X_i^\intercal \beta$.

Least squares: Choose $\hat{\beta}$ to minimize

$$\mathrm{RSS}(\beta) = \sum_{i=1}^{N} (Y_i - f_\beta(X_i))^2 = \sum_{i=1}^{N} (Y_i - X_i^\intercal \beta)^2$$

Regression: Prostate Cancer

Goal: predict level of PSA (prostate specific antigen) for men with prostate cancer

- Y_i: log PSA
- X_i: vector with **97 clinical measures**, including
  - log cancer volume
  - log prostate weight
  - Gleason score
  - ...

(see book for more details)

Important Trick: Make More Features

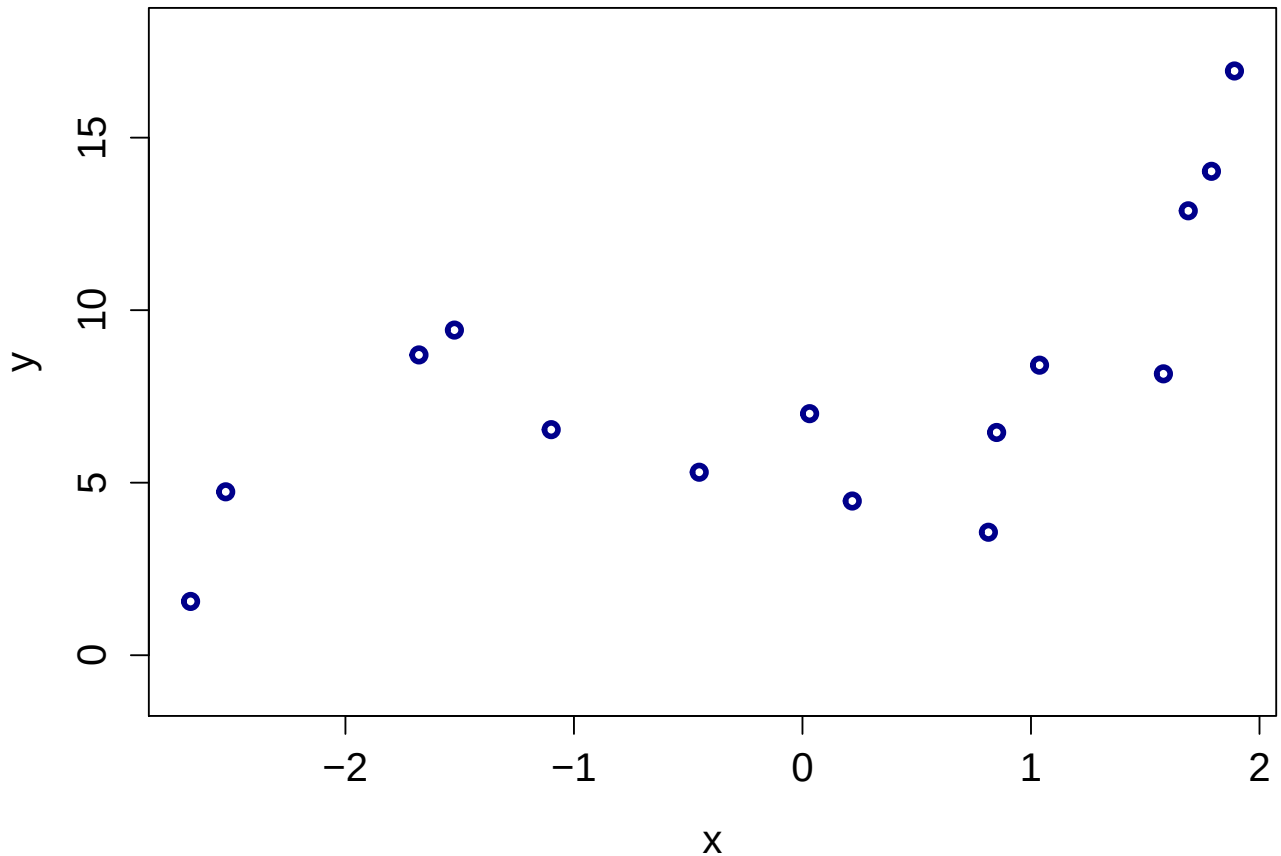
- Transform X_i into $\tilde{X}_i = \phi(X_i)$ for **feature map** $\phi : \mathbb{R}^p \to \mathbb{R}^m$
- Example with $X_i \in \mathbb{R}$ (polynomials):

$$\tilde{X}_{i,0} = 1$$
$$\tilde{X}_{i,1} = X_i$$
$$\tilde{X}_{i,2} = X_i^2$$
$$\tilde{X}_{i,3} = X_i^3$$
etc.

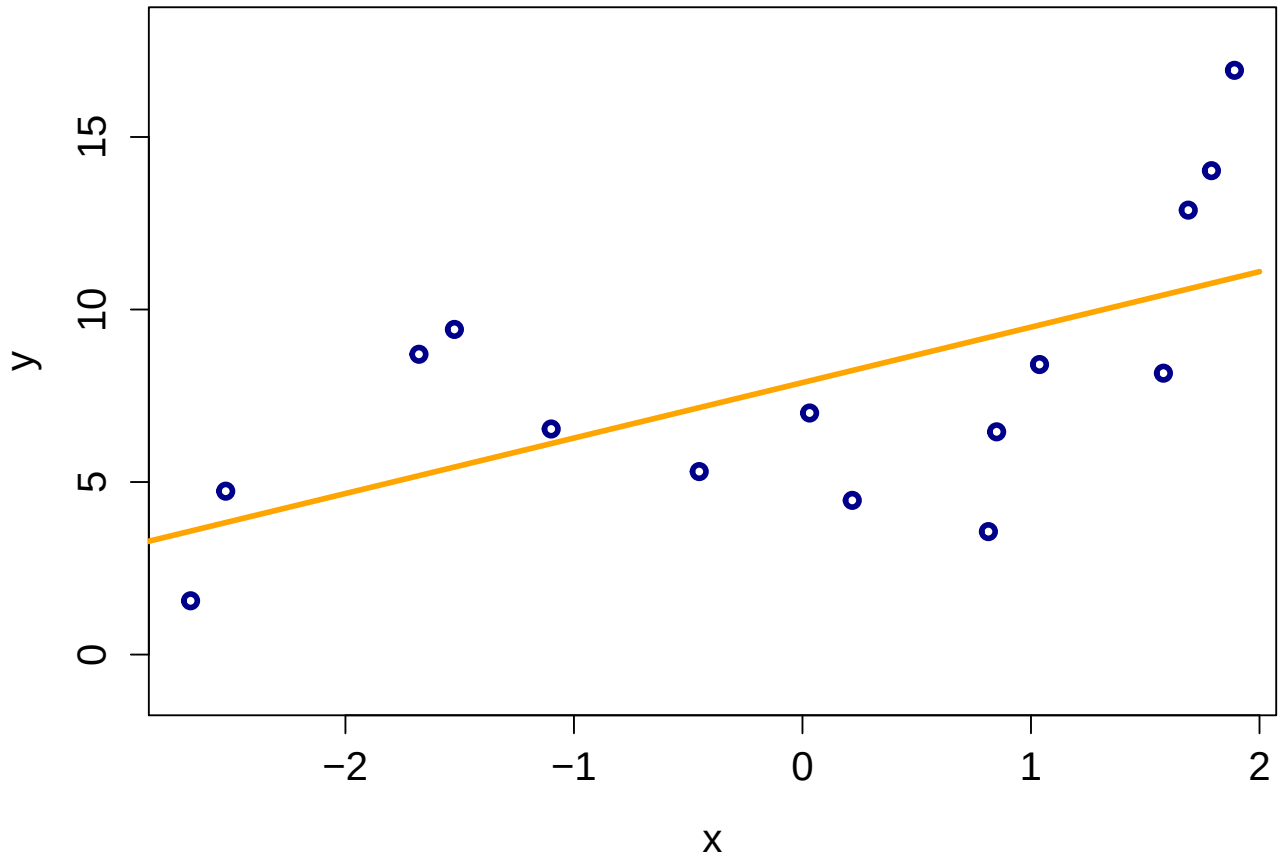
- Model **linear in new features**, but not in original features!

- We will see some methods that take this idea to extremes:
  - Support vector machines create an infinite number of features
  - Neural networks learn multiple layers of feature transformations that build on top of the previous layer

Simulated Data

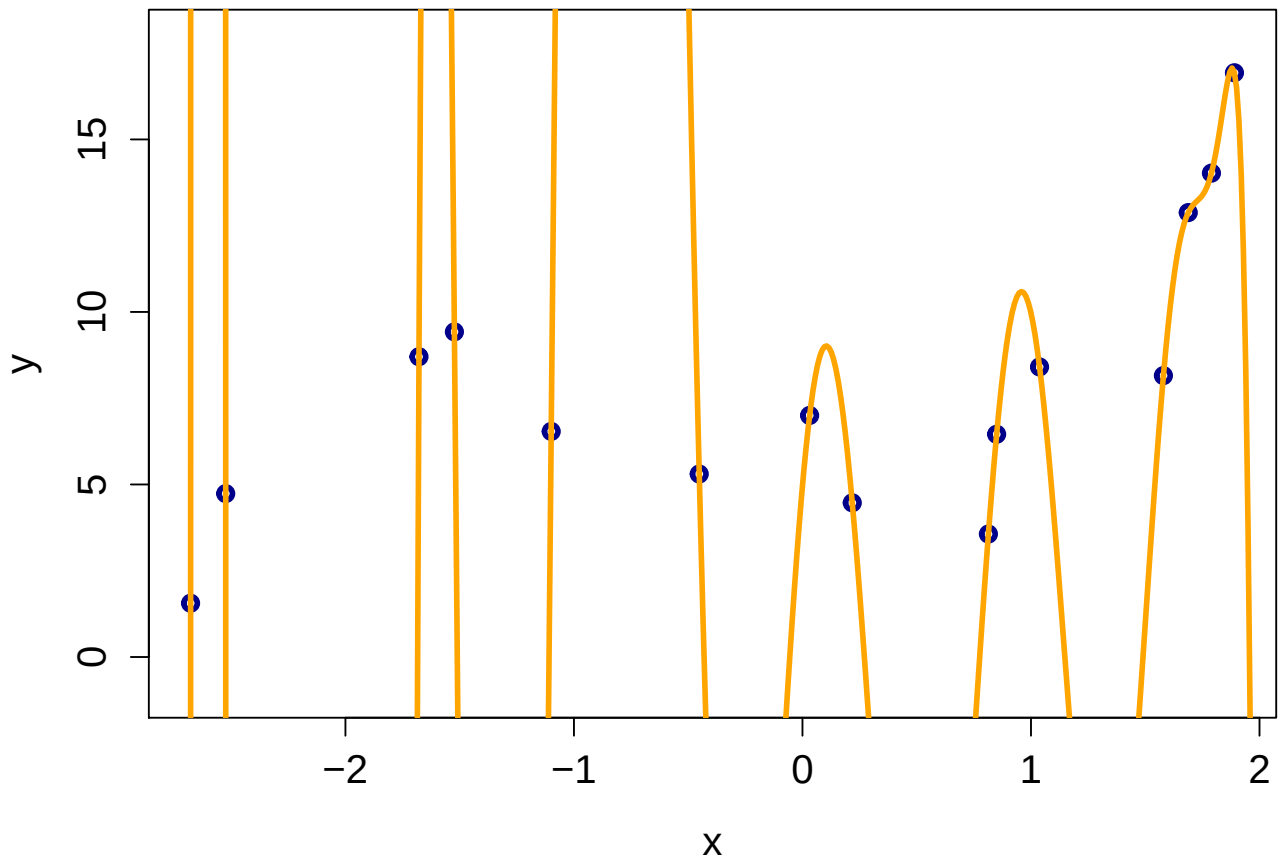


1-st Degree Polynomial



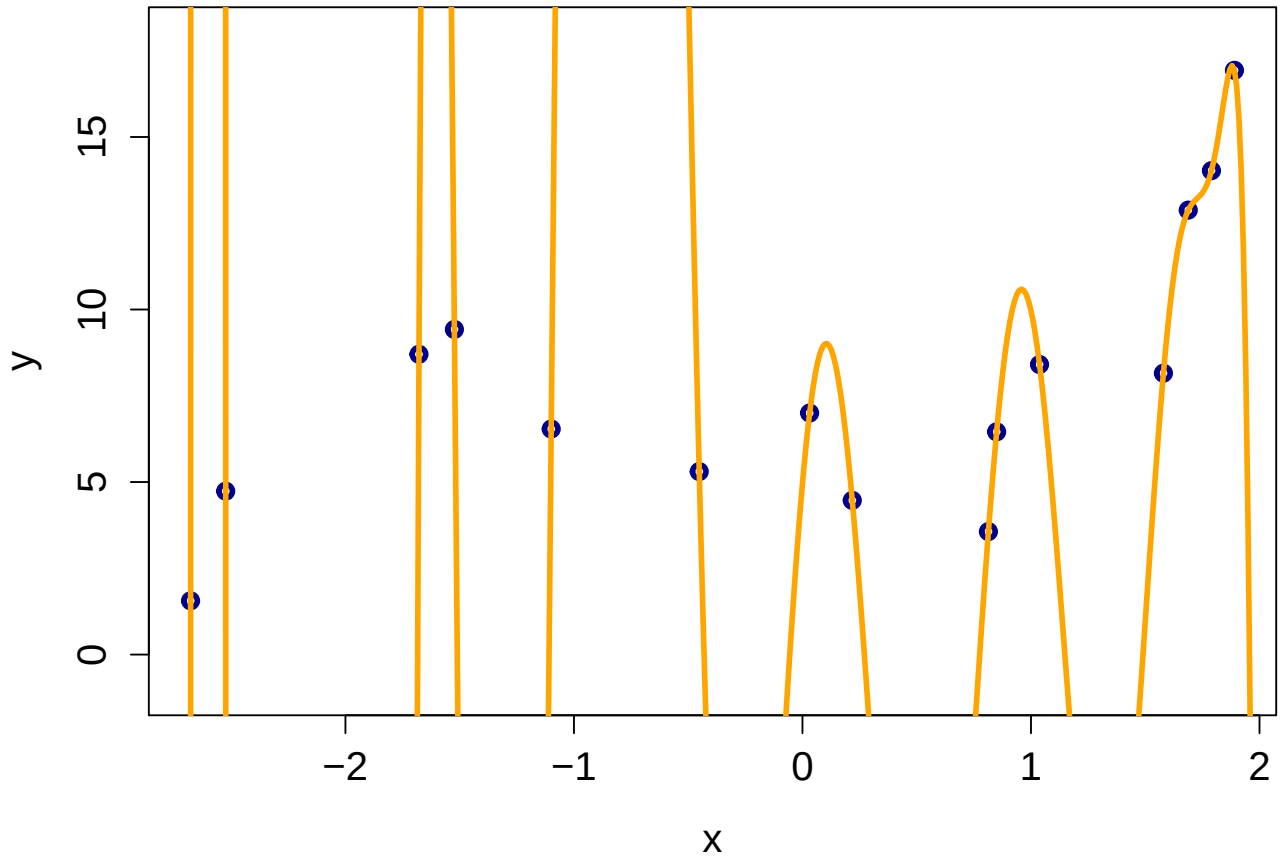
Underfitting: too few parameters

14-th Degree Polynomial



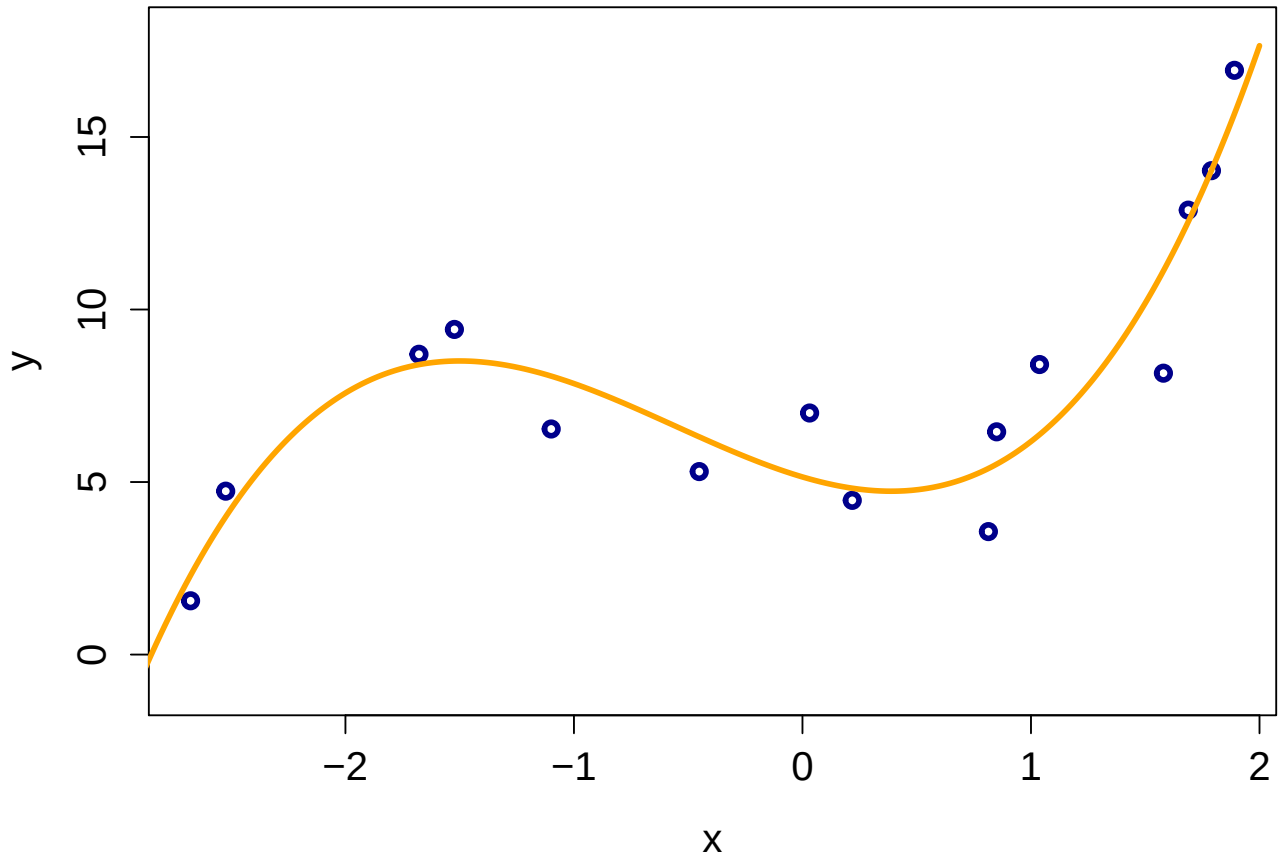
- Fits the data perfectly!
- Smaller error on the data, so must be better, right?

14-th Degree Polynomial



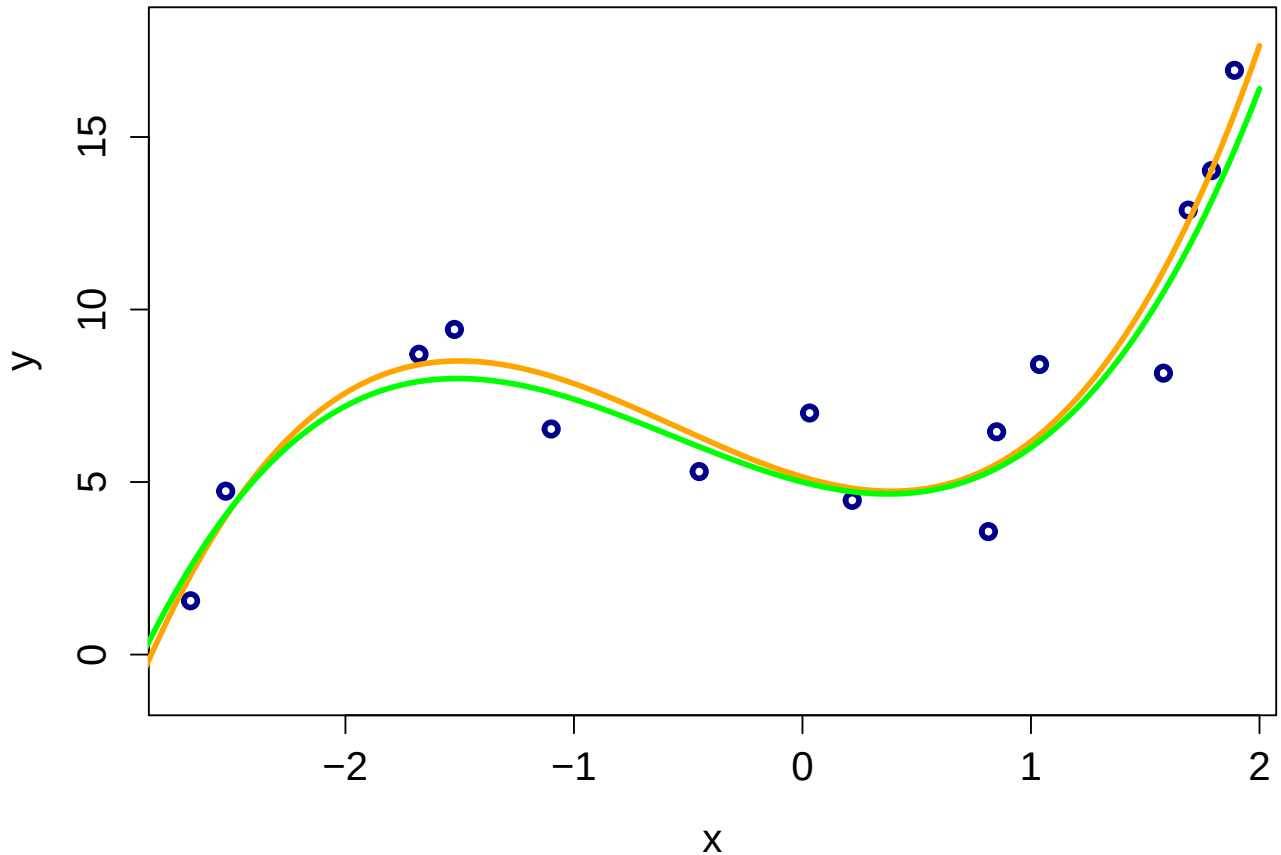
Overfitting: too many parameters!

3-rd Degree Polynomial



Intermediate number of parameters

3-rd Degree Polynomial



Intermediate number of parameters

Example: Linear Regression for Classification I

NB This is just a first example method. Logistic regression is better!

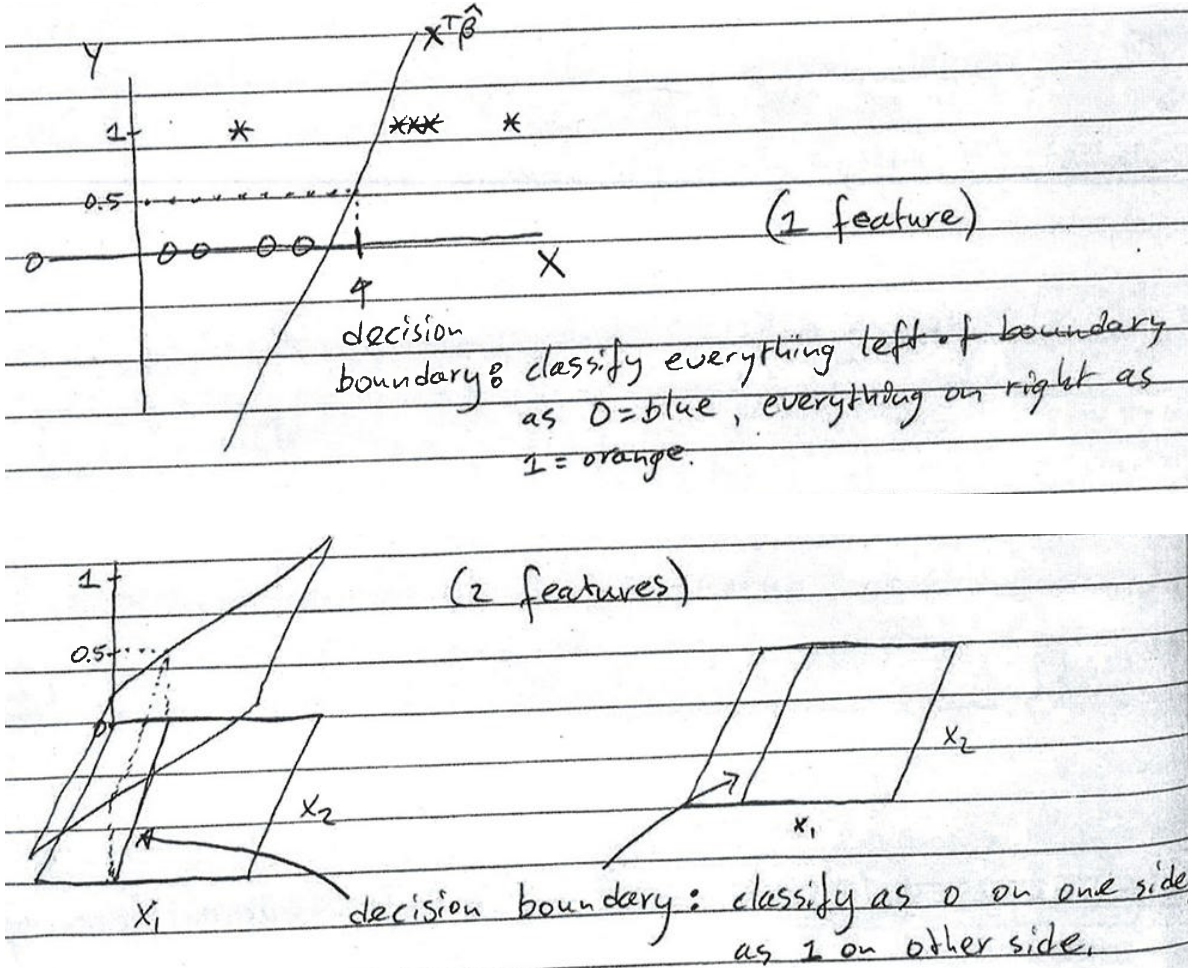
- $G_i \in \{\text{blue}, \text{orange}\} \rightarrow Y_i \in \{0, 1\}, \qquad X_i \in \mathbb{R}^2$

Simple idea:

- Use linear regression on this data
- Convert back to binary predictions:

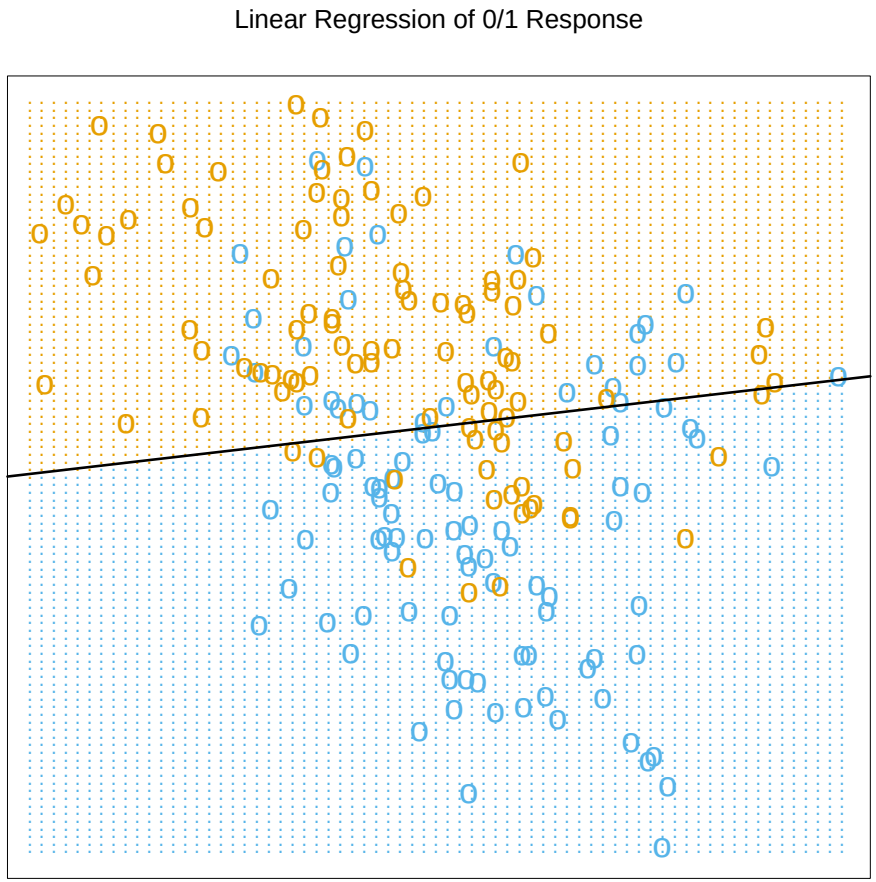
$$X^\intercal \beta > 0.5 \Rightarrow \hat{f}(X) = 1$$
$$X^\intercal \beta \leq 0.5 \Rightarrow \hat{f}(X) = 0$$

Example: Linear Regression for Classification II



Decision boundary: Classify everything on one side as 0, and on the other side as 1

Example: Linear Regression for Classification III



- Many errors on training data. Avoidable?
- Maybe if we construct more features:
  X_1, X_2, X_1^2, X_2^2, X_1X_2, $X_1^2X_2$, ...
- Too few: underfitting; too many: overfitting

K-Nearest Neighbor Classification I

To classify new X:

- Ask K **points** in training data T **closest to** X their class label Y
- Choose **most common class**
- Demo: www.timvanerven.nl/teaching/knn-visualization/

K-Nearest Neighbor Classification I

To classify new X:

- Ask K **points** in training data T **closest to** X their class label Y
- Choose **most common class**
- Demo: www.timvanerven.nl/teaching/knn-visualization/

For two classes: $Y_i \in \{0, 1\}$

- Estimate $\Pr(Y = 1|X)$ by **empirical average in region** around X:

$$\hat{p} = \frac{1}{K} \sum_{i:X_i \in N_K(X)} Y_i$$

- $\hat{p} > 0.5 \Rightarrow$ predict 1, $\quad \hat{p} \leq 0.5 \Rightarrow$ predict 0

K-Nearest Neighbor Classification I

To classify new X:

- Ask K **points** in training data T **closest to** X their class label Y
- Choose **most common class**
- Demo: www.timvanerven.nl/teaching/knn-visualization/

For two classes: $Y_i \in \{0, 1\}$

- Estimate $\Pr(Y = 1|X)$ by **empirical average in region** around X:

$$\hat{p} = \frac{1}{K} \sum_{i:X_i \in N_K(X)} Y_i$$

- $\hat{p} > 0.5 \Rightarrow$ predict 1, $\quad \hat{p} \leq 0.5 \Rightarrow$ predict 0

- K **small:** very flexible decision boundary, but very unstable estimates based on few neighbors (risks overfitting)
- K **large:** stable estimates, but inflexible decision boundary (risks underfitting)

K-Nearest Neighbor Classification I

To classify new X:

- Ask K **points** in training data T **closest to** X their class label Y
- Choose **most common class**
- Demo: www.timvanerven.nl/teaching/knn-visualization/

For two classes: $Y_i \in \{0, 1\}$

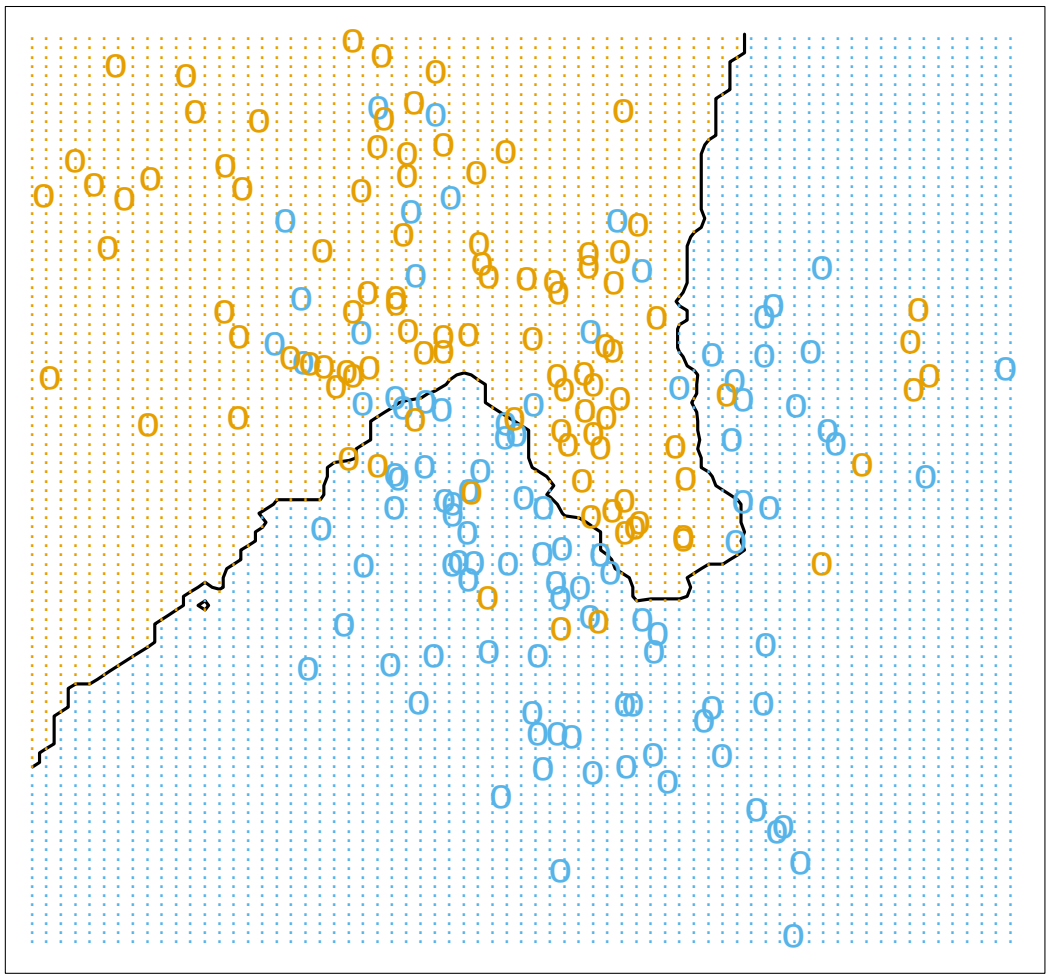
- Estimate $\Pr(Y = 1|X)$ by **empirical average in region** around X:

$$\hat{p} = \frac{1}{K} \sum_{i:X_i \in N_K(X)} Y_i$$

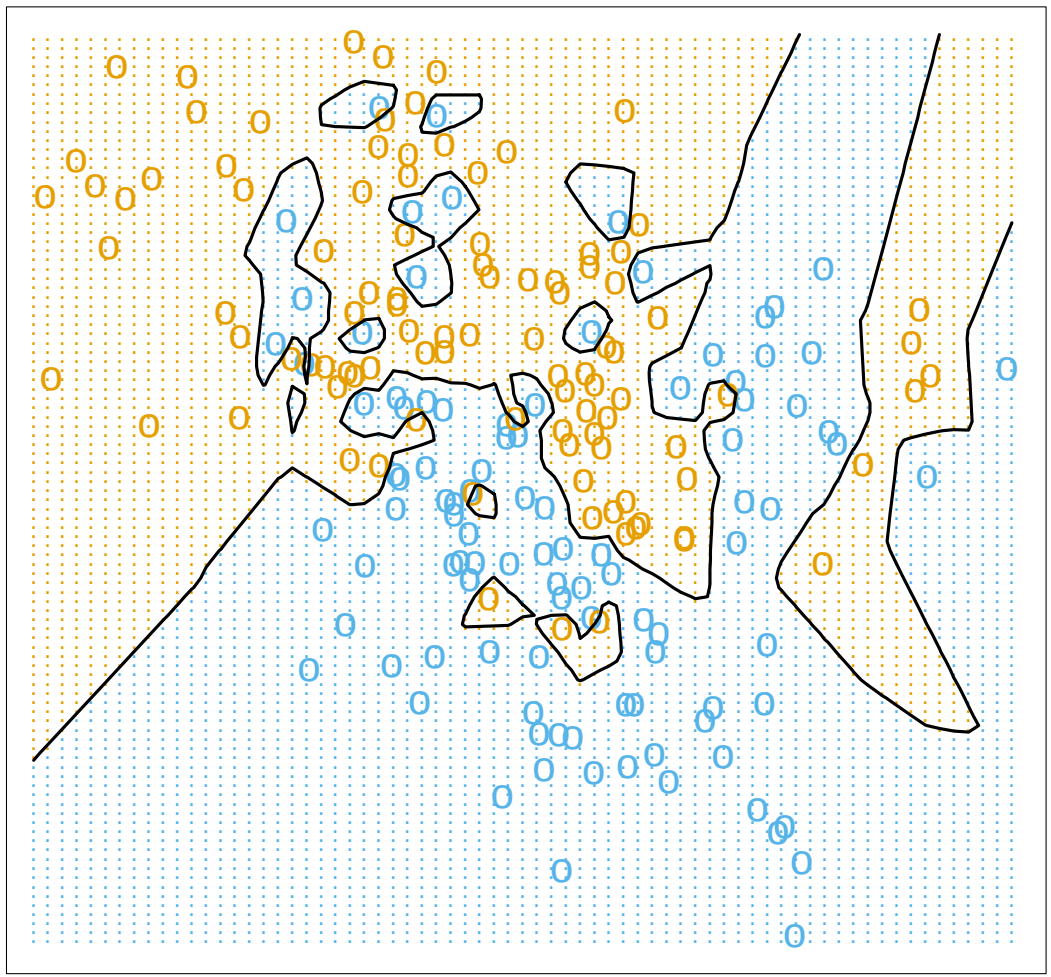
- $\hat{p} > 0.5 \Rightarrow$ predict 1, $\quad \hat{p} \leq 0.5 \Rightarrow$ predict 0

- Model $\mathcal{F}$ is **implicit**!
- Back of the envelope calculation:
  - If neighbors do not overlap: $\frac{N}{K}$ neighborhoods $\approx$ effective nr. of parameters, because estimate $\hat{p}$ for each neighborhood

K-Nearest Neighbor Classification II

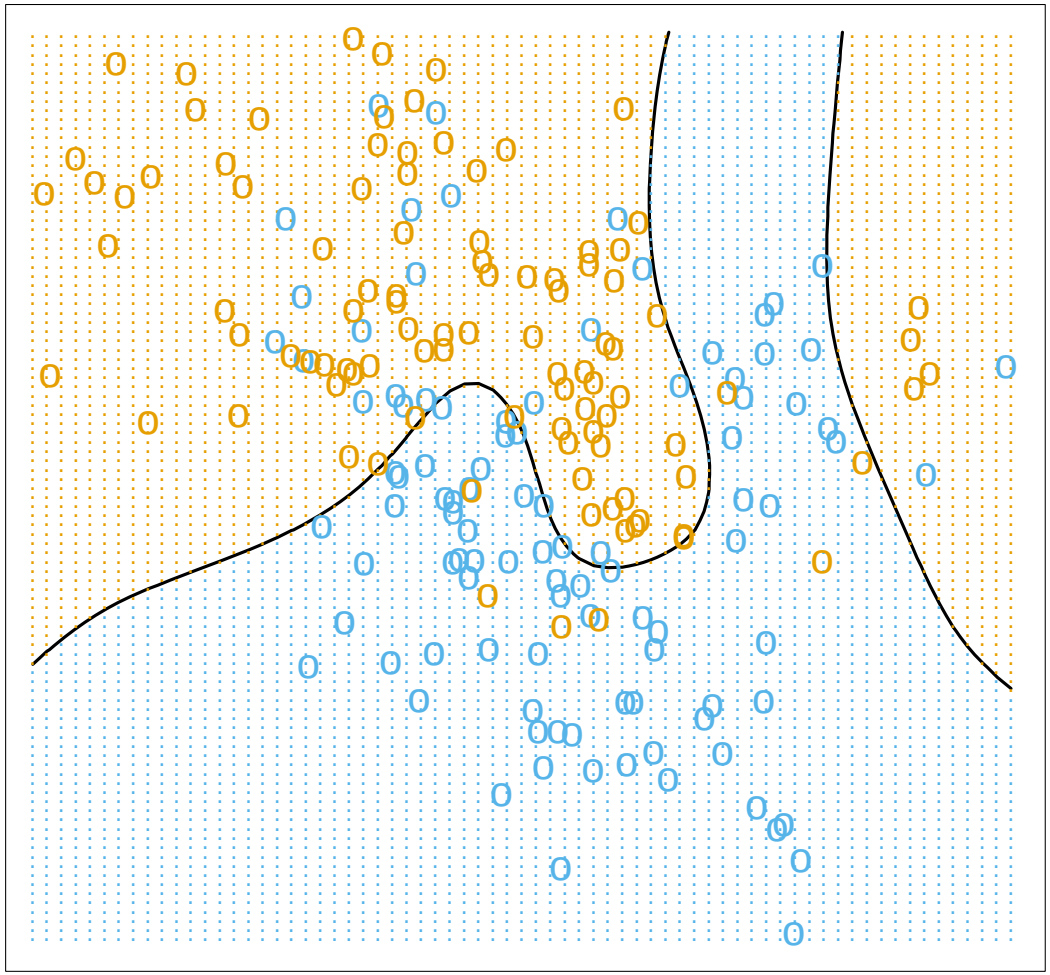


K-Nearest Neighbor Classification II

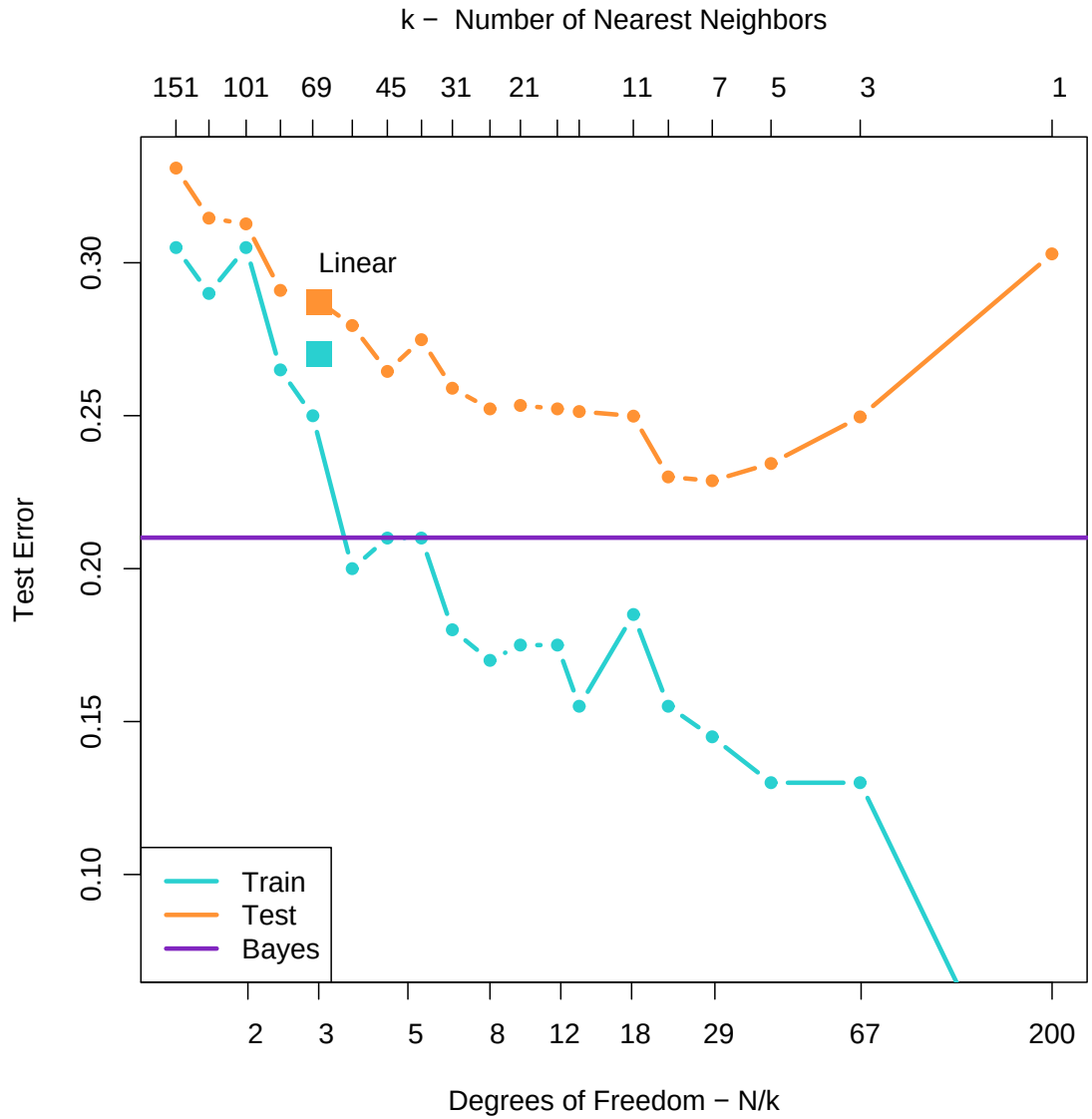


K-Nearest Neighbor Classification II

Bayes Optimal Classifier

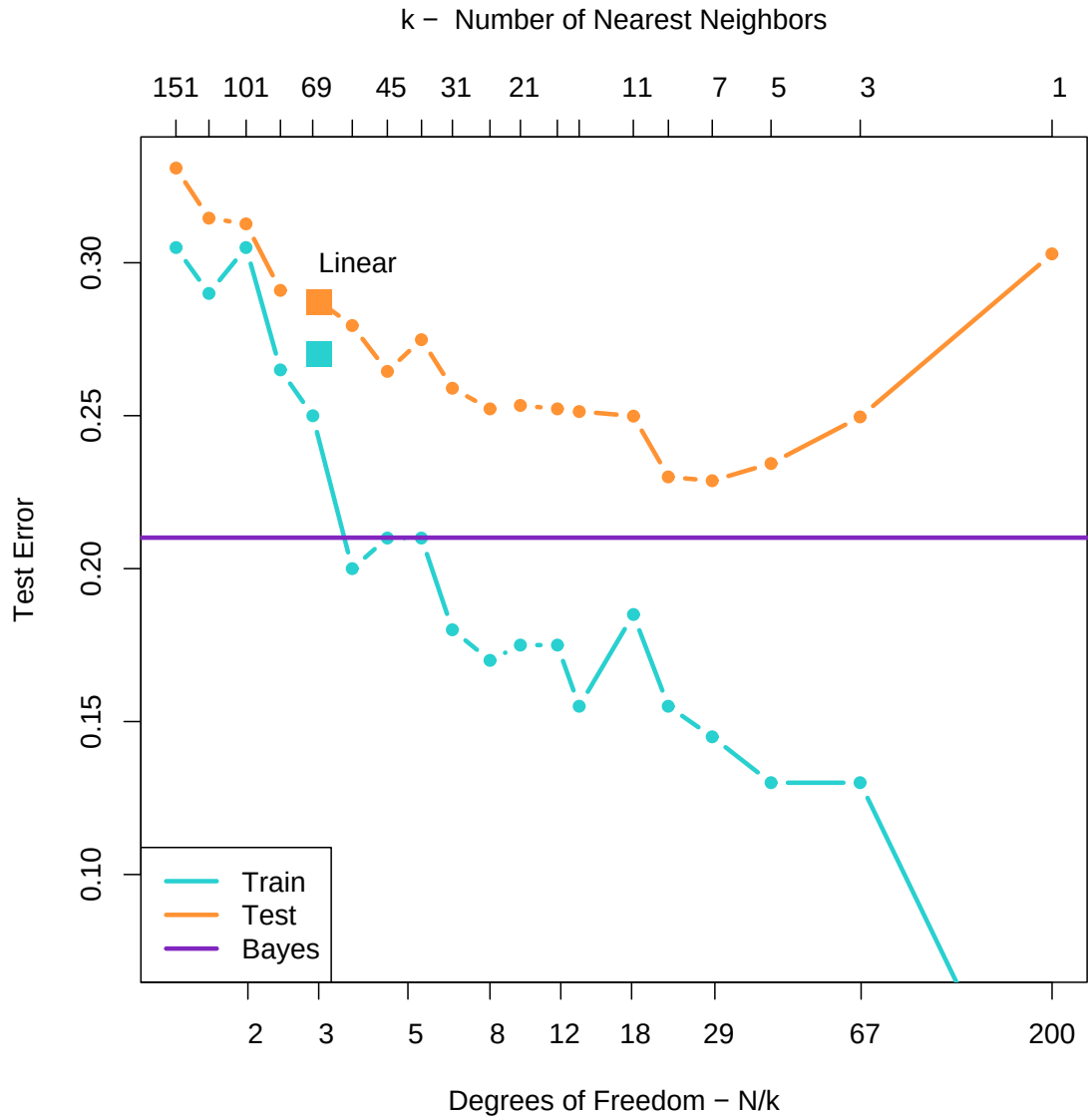


K-Nearest Neighbor Classification II



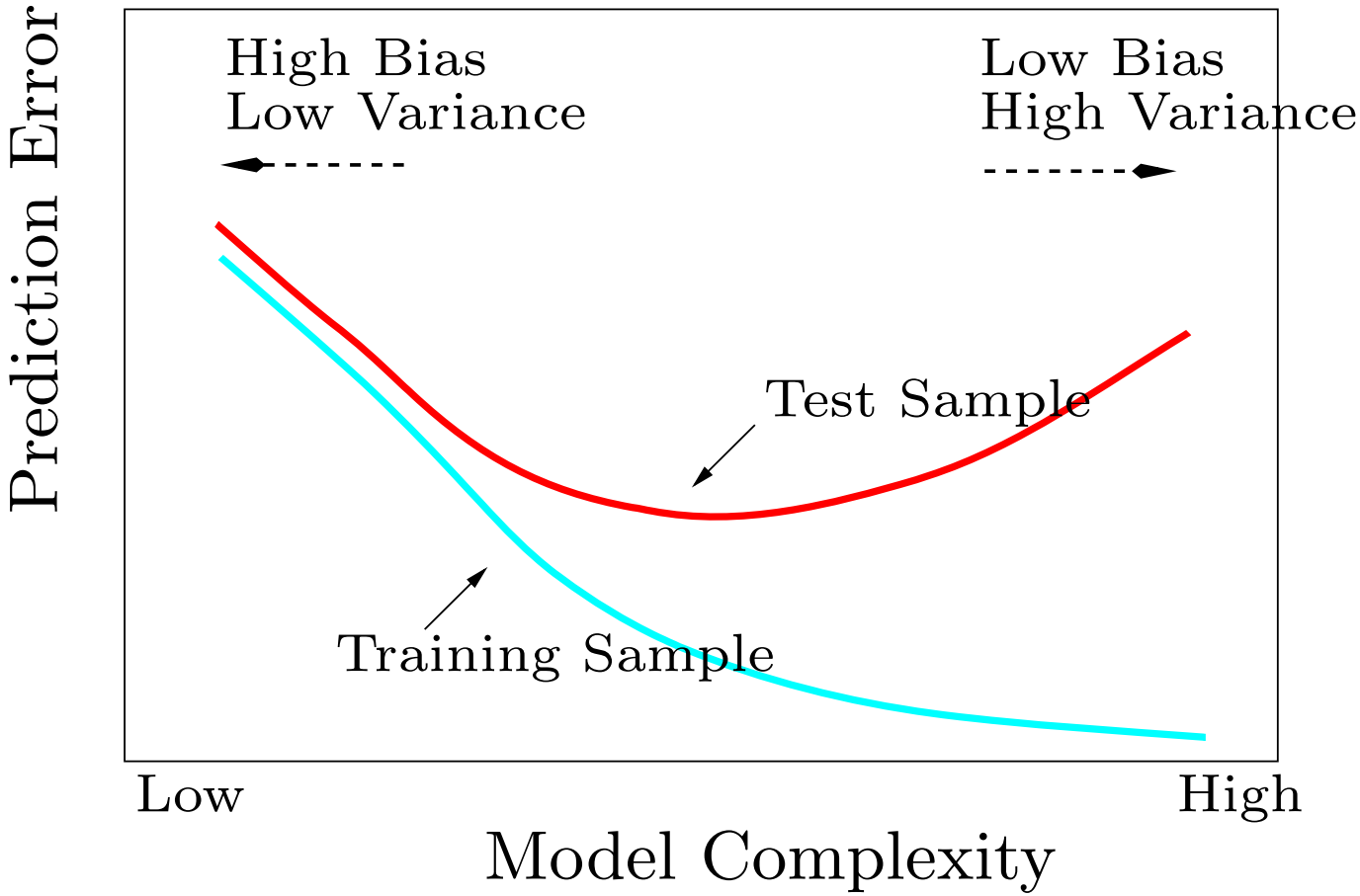
What is **best** K?

K-Nearest Neighbor Classification II



What is **best K?** Need **statistical decision theory**!

Model Complexity



Why Take This Course?

Knowledge Motivation:

- Machine learning methods are behind all modern AI
- Machine learning is one of the fastest growing fields in science; applications are **changing the world**
- Know the ideas behind it!
- See math in context!
- Perfect background for a theorem-proof style course: Machine Learning Theory in the master.

Practical Motivation:

- Many machine learning methods available as software packages, but without understanding, it is easy to make mistakes (overfitting, underfitting, ...)
  - Which ones should you use?
  - What do they do?
  - What are the key ideas behind them, and their main properties?
  - What do their settings mean?
- All methods we cover are very practical