



GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

Deep Learning

Lecture 3: Machine learning basics

Alexander Ecker
Institut für Informatik, Uni Göttingen



<https://alexanderecker.wordpress.com>

– Credit: slides largely based on Ian Goodfellow's and Chris Bishop's slides –

Homework assignments

First homework assignment will be on Stud.IP on Friday Nov. 29 (tomorrow)

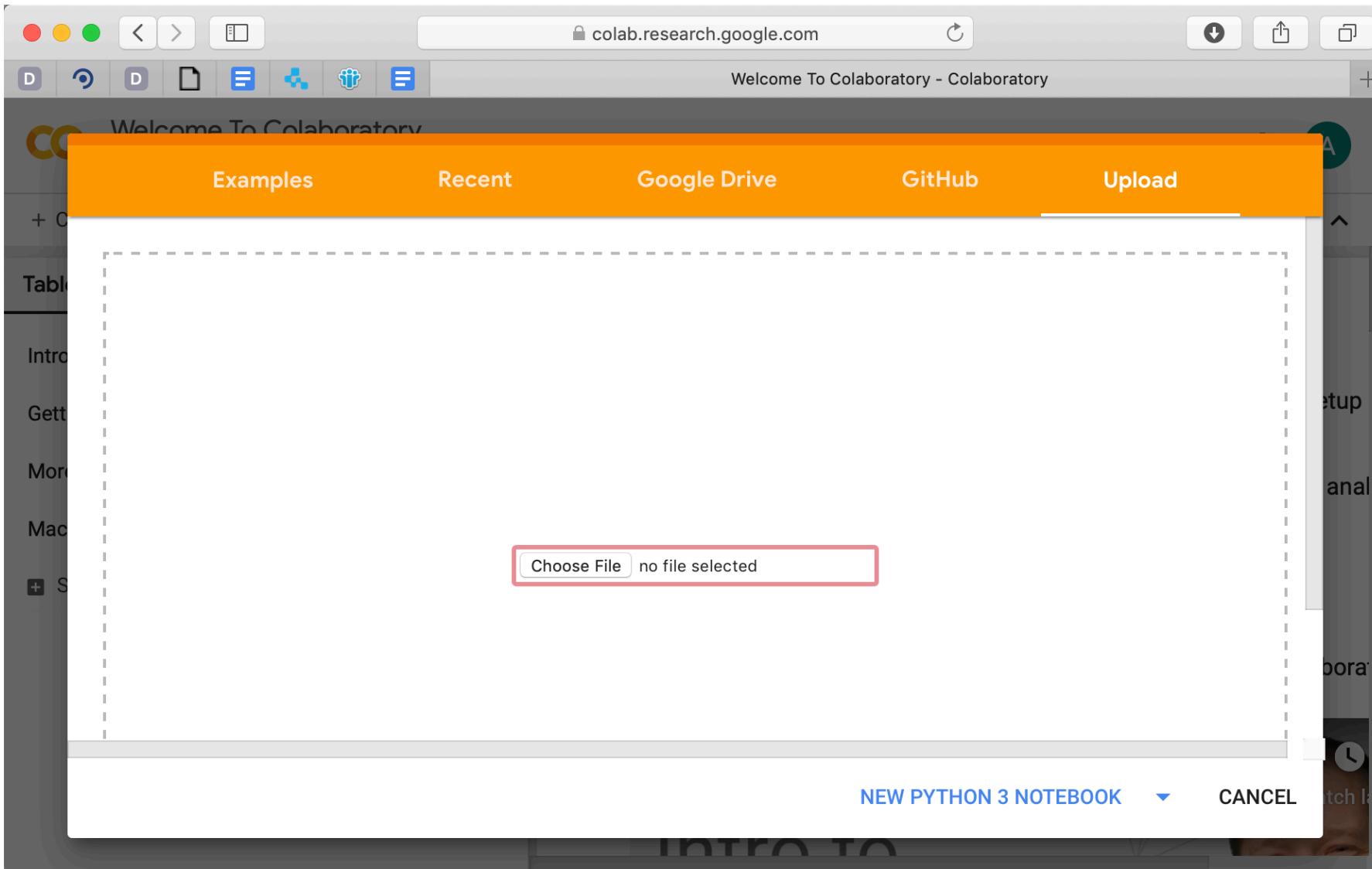
Form teams of 2–3 students

- No individuals please
- You'll present your solution to the tutors in tutorial (5 min)
- It's ok to split work, but everybody needs to be able to explain everything

Submission

- Upload complete Jupyter Notebook including output on Stud.IP (see following slides)
- Deadline for upload: Dec. 16 at noon

Google colab



Uploading your solution on Stud.IP (1/4)

The screenshot shows the Georg-August-Universität Göttingen website with the Stud.IP module active. The top navigation bar includes links for Veranstaltungen, Übersicht, Verwaltung, Forum, Teilnehmende, Dateien (which is circled in red), Ablaufplan, Informationen, Literatur, Wiki, and Mehr ...

The main content area displays information for a course titled "Vorlesung: Deep Learning - Kurzinfo". On the left, there are sections for "Kurzinfo" (with "Details" selected), "Teilen" (with a link to copy the event URL), and "UniVZ". The right side shows "Grunddaten" (Basic Data) and "Zeit / Veranstaltung" (Time / Event). It lists two weekly schedules: Monday 14:00 - 15:00 and Thursday 14:00 - 16:00, both in Room MN09, Hörsaal 32, GZG. It also mentions a one-time session on Monday, 28.10.19, from 13:00 to 14:00. The "Nächster Termin" (Next Term) is listed as Thursday, 28.11.2019, from 14:00 to 16:00. The "DozentInnen" (Teachers) section lists Prof. Dr. Alexander Ecker.

Uploading your solution on Stud.IP (1/4)

The screenshot shows the Georg-August-Universität Göttingen Stud.IP interface. The top navigation bar includes the university logo, language selection (English), a search bar, and user account information. Below the bar, a menu bar offers links to Veranstaltungen, Übersicht, Verwaltung, Forum, Teilnehmende, Dateien (which is selected), Ablaufplan, Informationen, Literatur, Wiki, and Mehr ...

The main content area displays a list of files for a lecture titled "Vorlesung: Deep Learning". The list includes:

Typ	Name	Gruppe	Autor/-in	Datum	Aktionen
Slides	Submission of Homework 1		Ecker, Alexander	vor 21 Tagen	⋮
DL19_Topics.pdf	DL19_Topics.pdf	76.7 kB	Ecker, Alexander	vor 12 Stunden	⋮

A red arrow points to the second row, which is highlighted with a red circle. The "Submission of Homework 1" file was uploaded by Ecker, Alexander, and is dated "vor 21 Tagen".

On the left side, there are buttons for "Aktionen" (New folder, Add file) and "Dateien hochladen" (Upload files). At the bottom, there are buttons for "Herunterladen" (Download), "Verschieben" (Move), "Kopieren" (Copy), and "Löschen" (Delete).

Uploading your solution on Stud.IP (1/4)

The screenshot shows the Georg-August-Universität Göttingen Stud.IP interface. The top navigation bar includes the university logo, language selection (English), a search bar, and user account information. Below the bar, a menu bar offers links to Veranstaltungen, Übersicht, Verwaltung, Forum, Teilnehmende, Dateien (which is selected), Ablaufplan, Informationen, Literatur, Wiki, and Mehr ...

The main content area displays a section titled "Vorlesung: Deep Learning - Dateien". On the left, there's a sidebar with actions like "Ordner bearbeiten", "Neuer Ordner", and "Datei hinzufügen". The main area shows a submission form for "Submission of Homework 1", which is due by 16.12.2019 at 12:00:00. It states that this is a homework folder where only files can be uploaded. A red arrow points to the "Datei hinzufügen" button, which is highlighted with a red circle.

Uploading your solution on Stud.IP (4/4)

Important!

Notebook must be executable from top to bottom without errors

File name: names of the authors, e.g. `ecker_smith_mueller.ipynb`

Upload only one file per team

Topics today

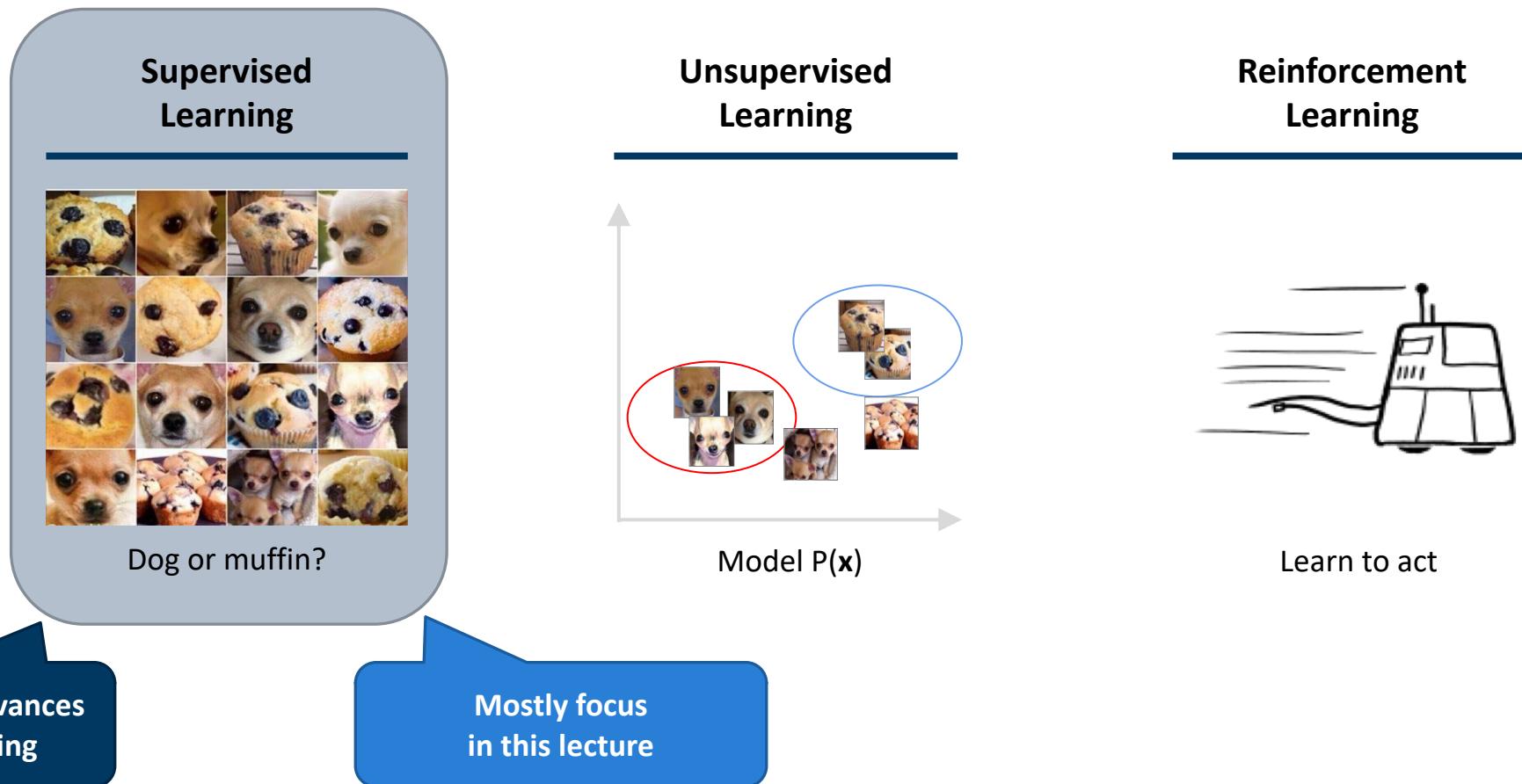
Review of ML basics – case study: polynomial curve fitting

Logistic regression

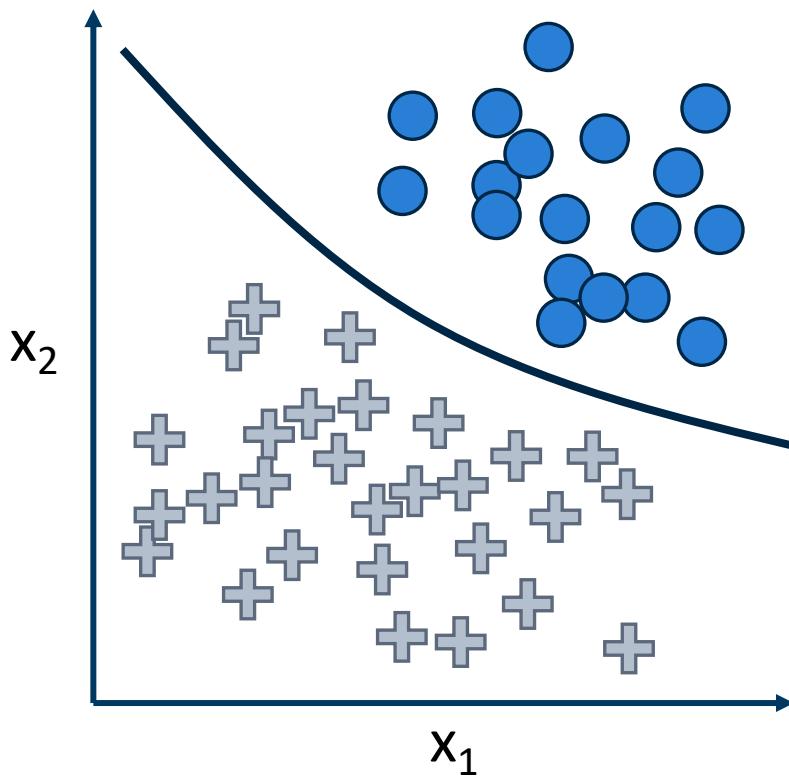
Gradient descent

Machine learning

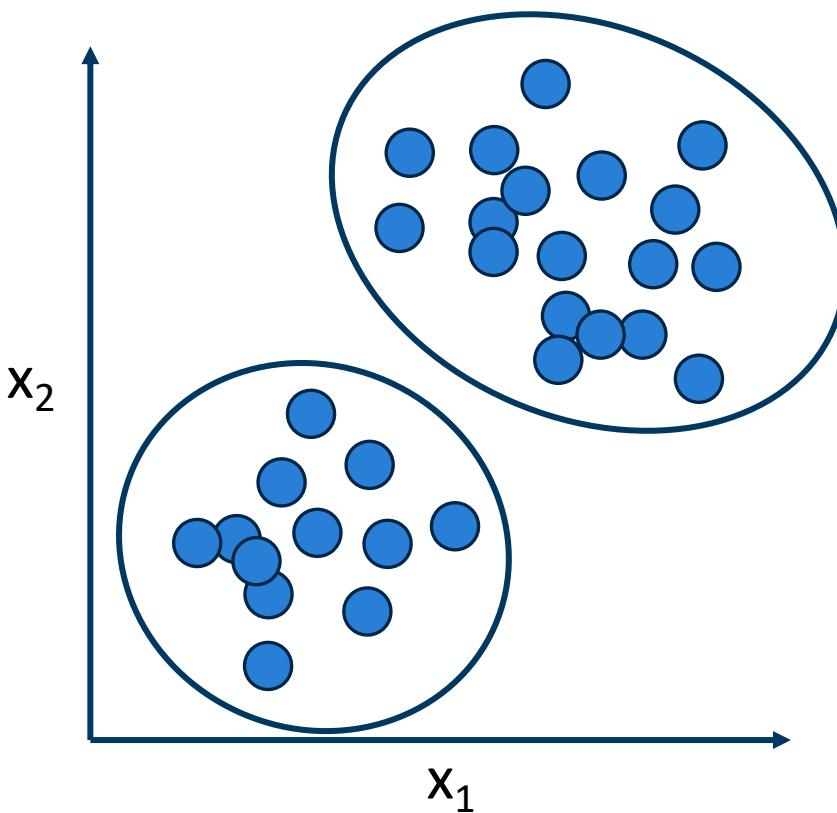
Three main forms of machine learning



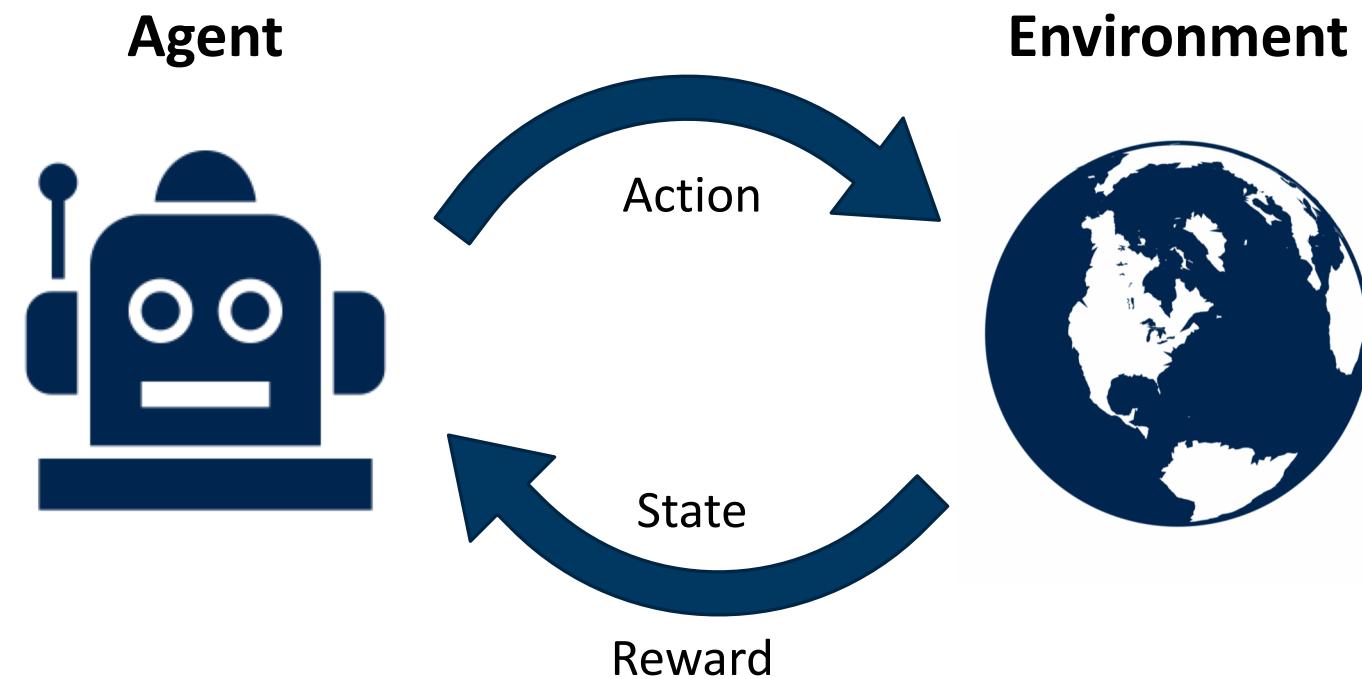
Supervised learning



Unsupervised learning



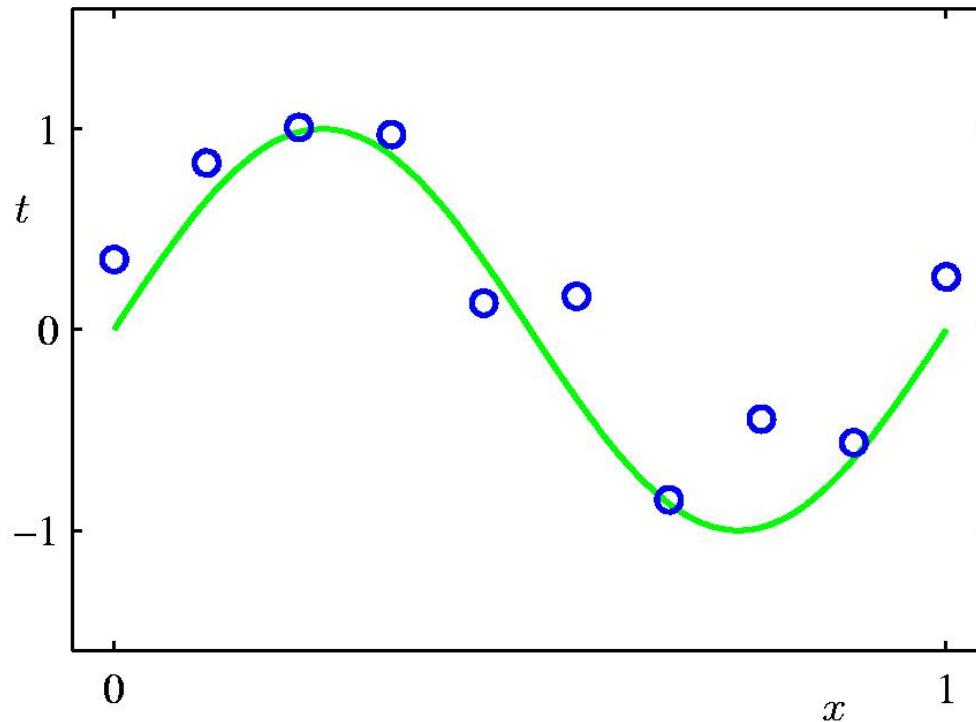
Reinforcement learning



Case study

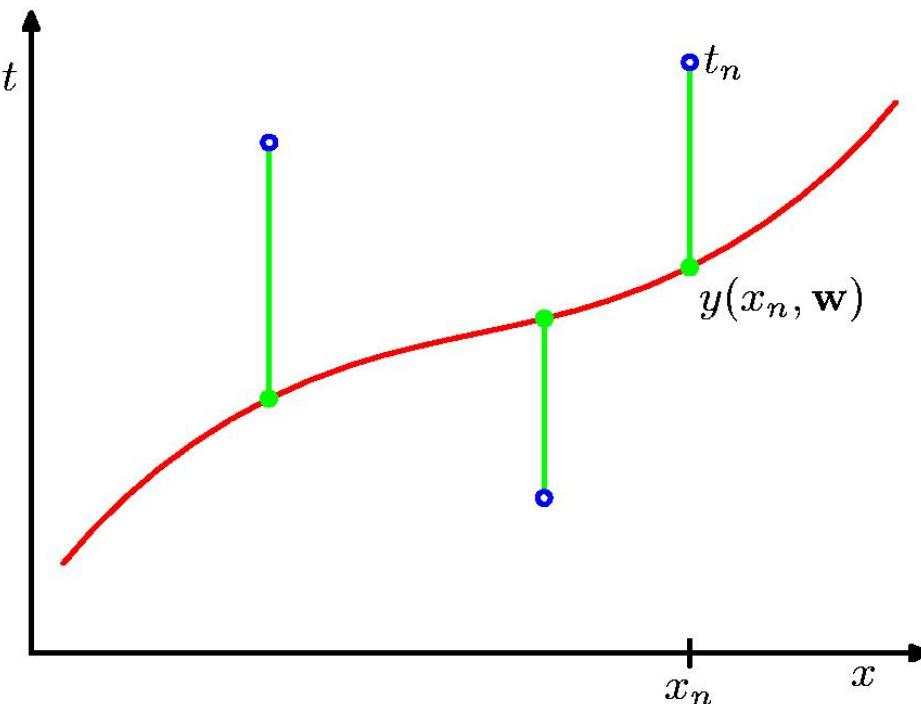
POLYNOMIAL CURVE FITTING

Polynomial Curve Fitting



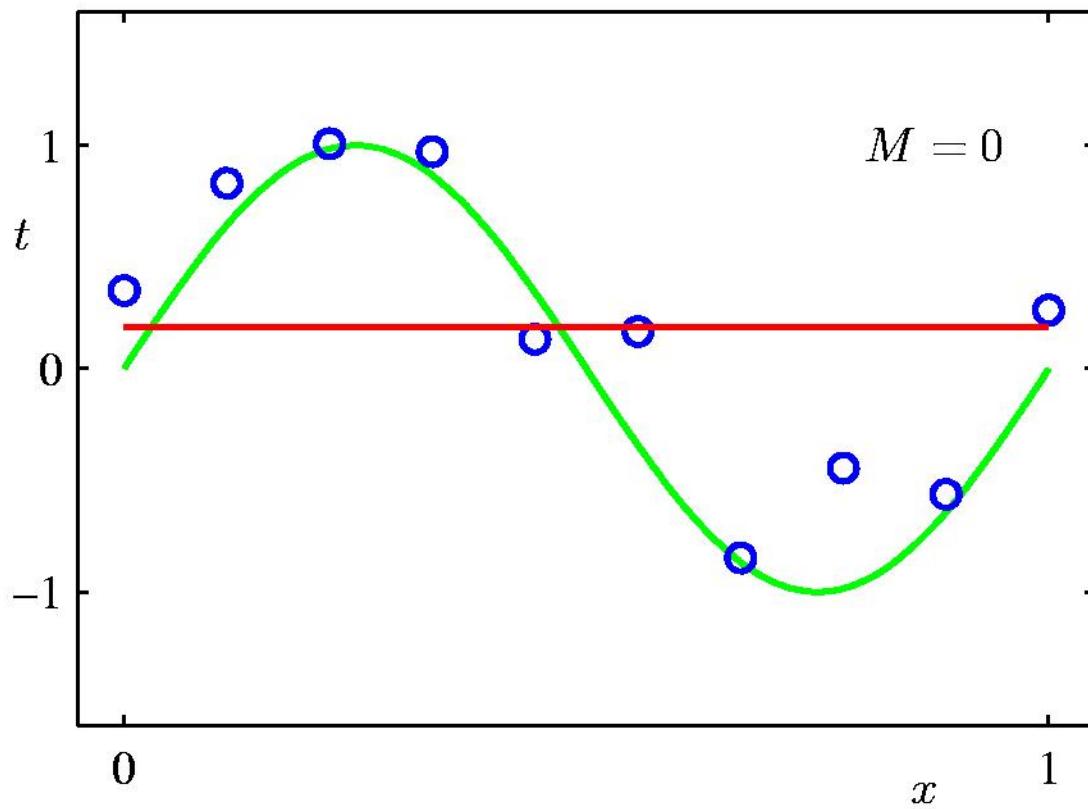
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \dots + w_M x^M = \sum_{j=0}^M w_j x^j$$

Sum-of-Squares Error Function

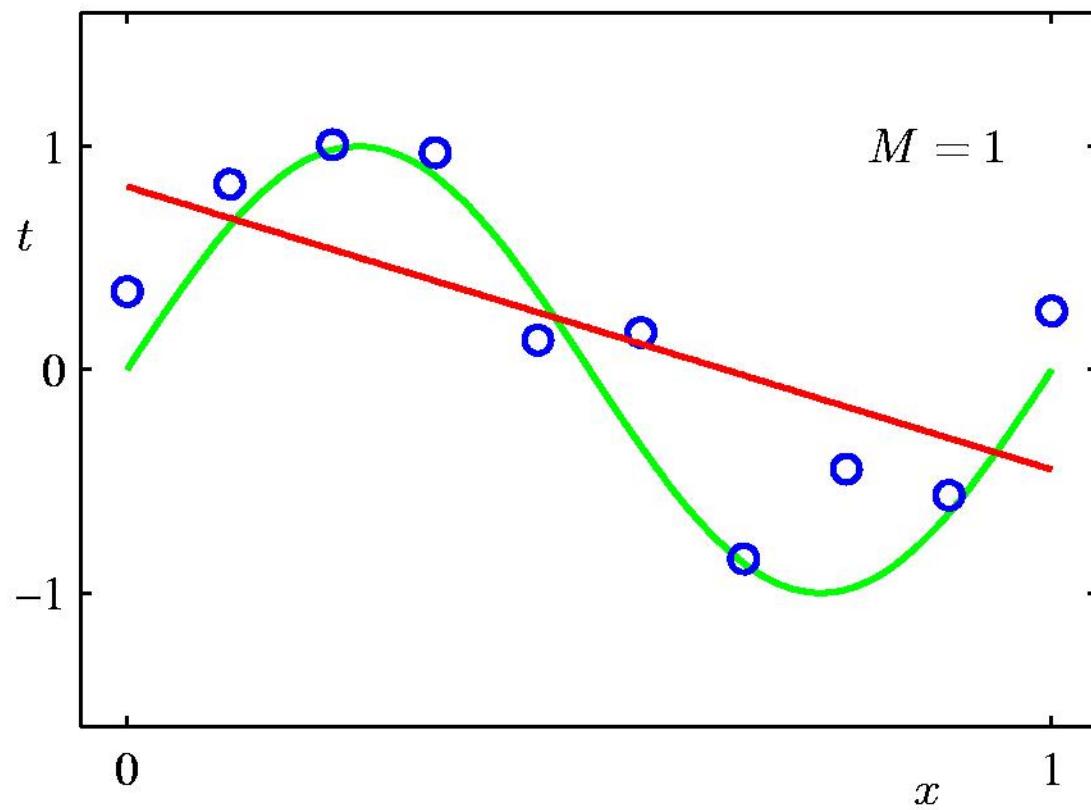


$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2$$

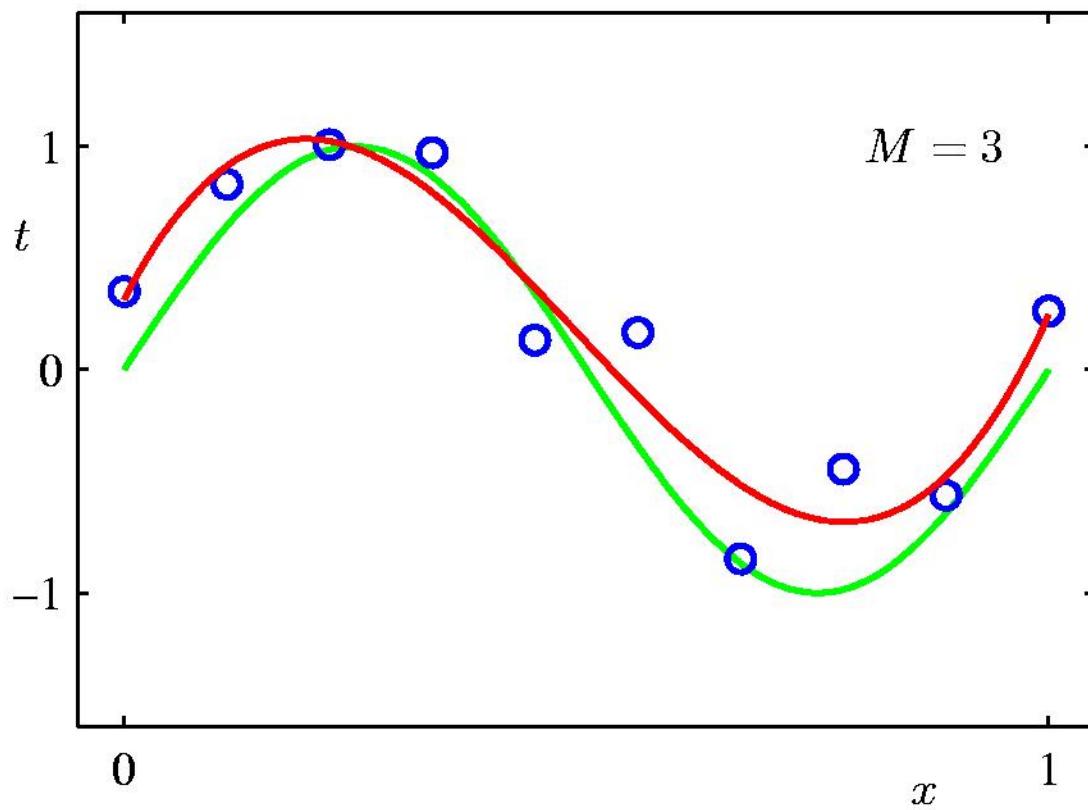
0th Order Polynomial



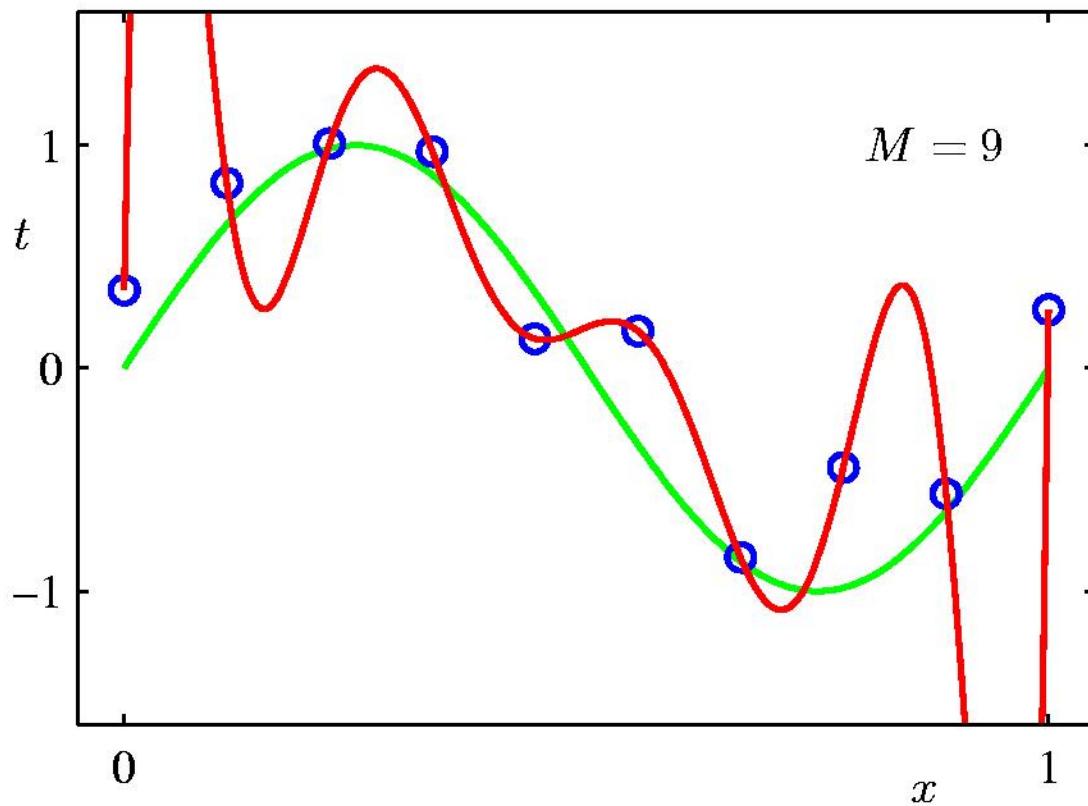
1st Order Polynomial



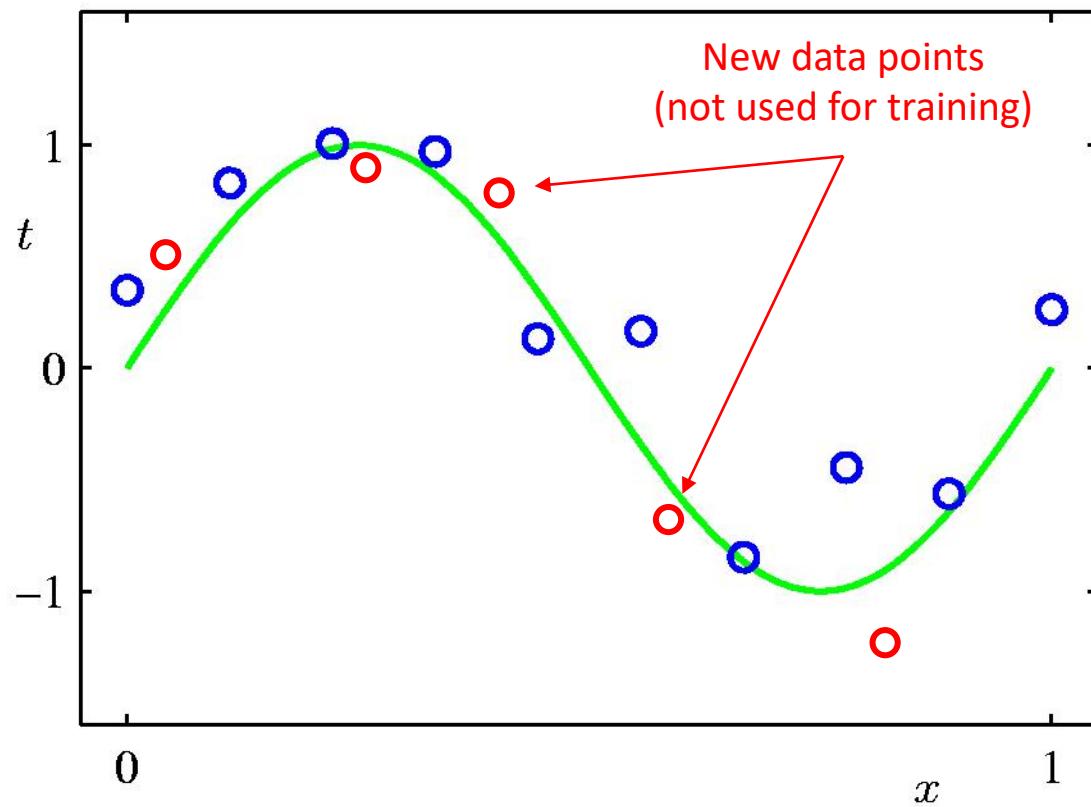
3rd Order Polynomial



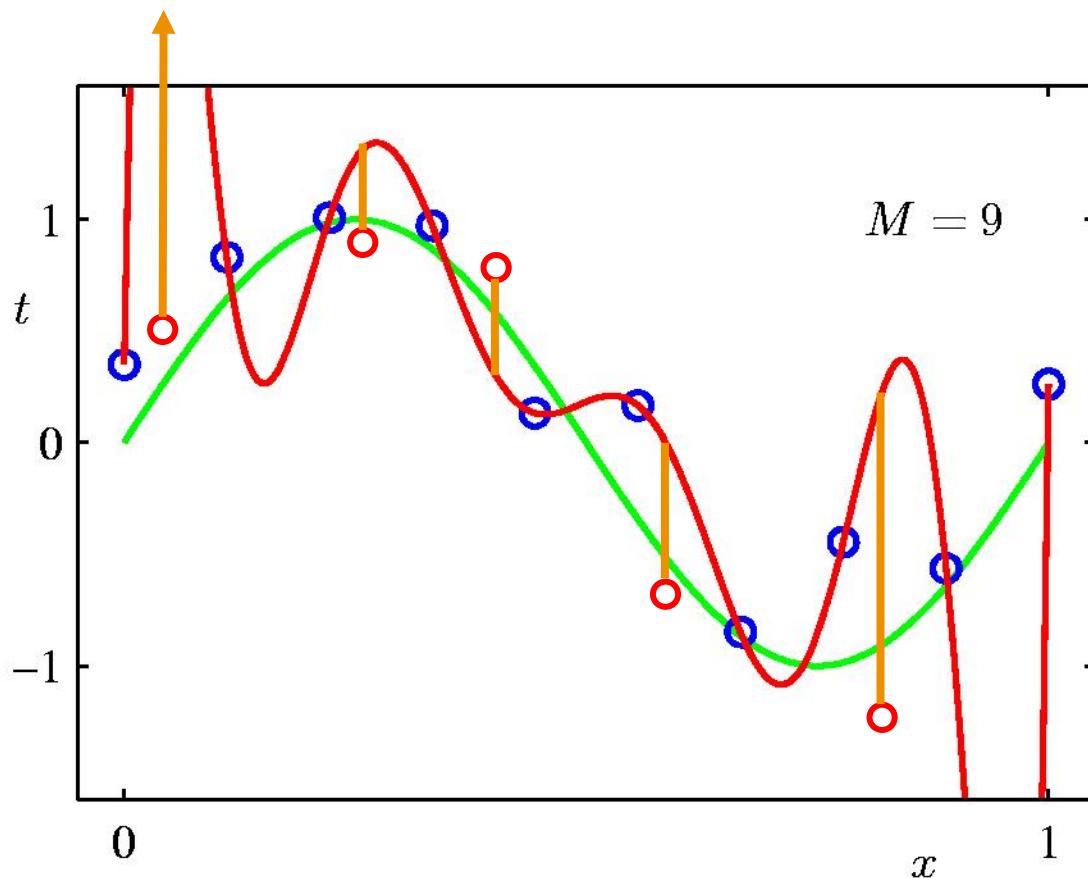
9th Order Polynomial



Goal: Generalization to new data

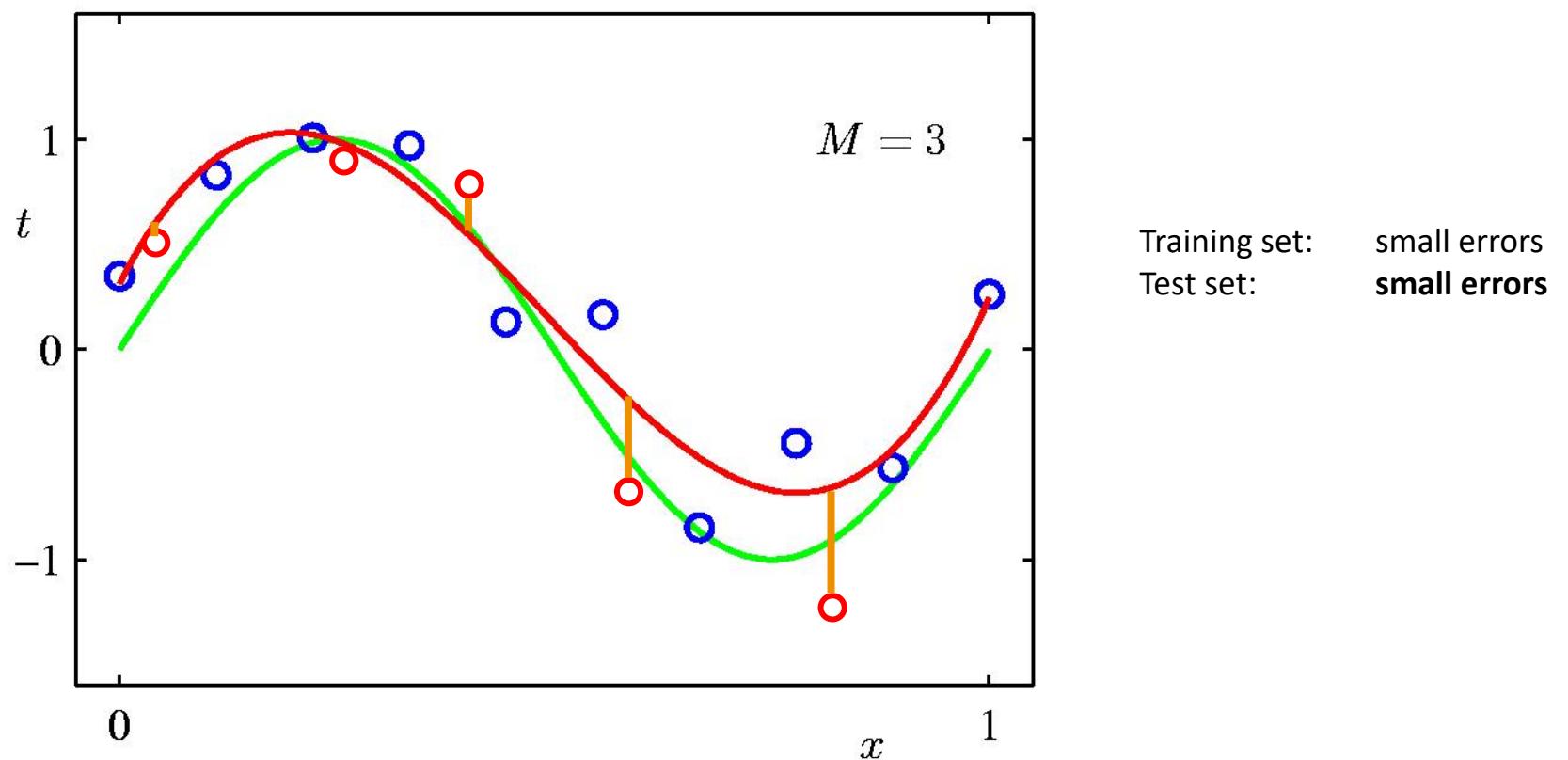


9th Order Polynomial

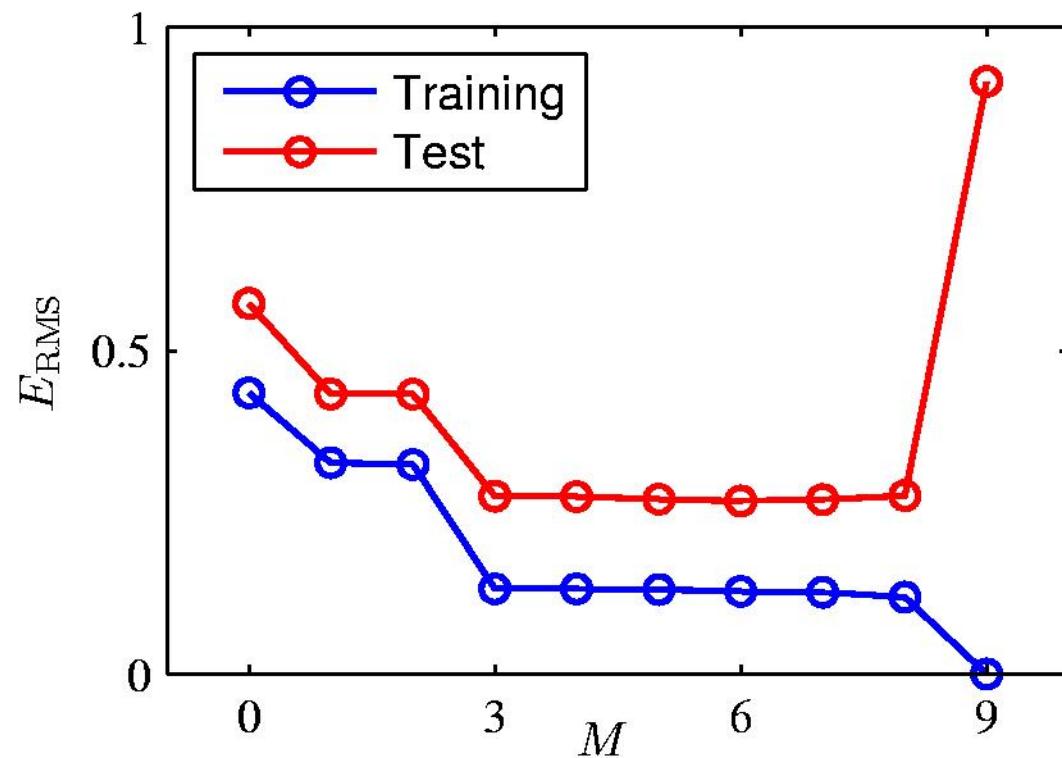


Training set: perfect fit
Test set: **large errors!**

3rd Order Polynomial



Overfitting



Root-Mean-Square (RMS) Error: $E_{\text{RMS}} = \sqrt{2E(\mathbf{w}^*)/N}$

Polynomial Coefficients

	$M = 0$	$M = 1$	$M = 3$	$M = 9$
w_0^*	0.19	0.82	0.31	0.35
w_1^*		-1.27	7.99	232.37
w_2^*			-25.43	-5321.83
w_3^*			17.37	48568.31
w_4^*				-231639.30
w_5^*				640042.26
w_6^*				-1061800.52
w_7^*				1042400.18
w_8^*				-557682.99
w_9^*				125201.43

Goal of ML: risk minimization

Assuming

- **Hypothesis** $h: \mathcal{X} \rightarrow \mathcal{Y}$ from a hypothesis space \mathcal{H}
- **Prediction** $\hat{y} = h(x)$
- **Training samples** (x_n, y_n) drawn independently and identically from joint probability density $P(x, y)$
- **Loss function** $L(\hat{y}, y): \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$
- **Risk** $R: \mathcal{H} \mapsto \mathbb{R}$, the expected loss $R(h) = \mathbb{E}[L(h(x), y)]$

Goal of supervised learning:

Risk minimization:

$$\arg \min_{h \in \mathcal{H}} R(h)$$

Empirical risk minimization

In practice, risk $R(h)$ cannot be computed!

The distribution $P(x, y)$ is unknown to the learning algorithm

Typical solution:

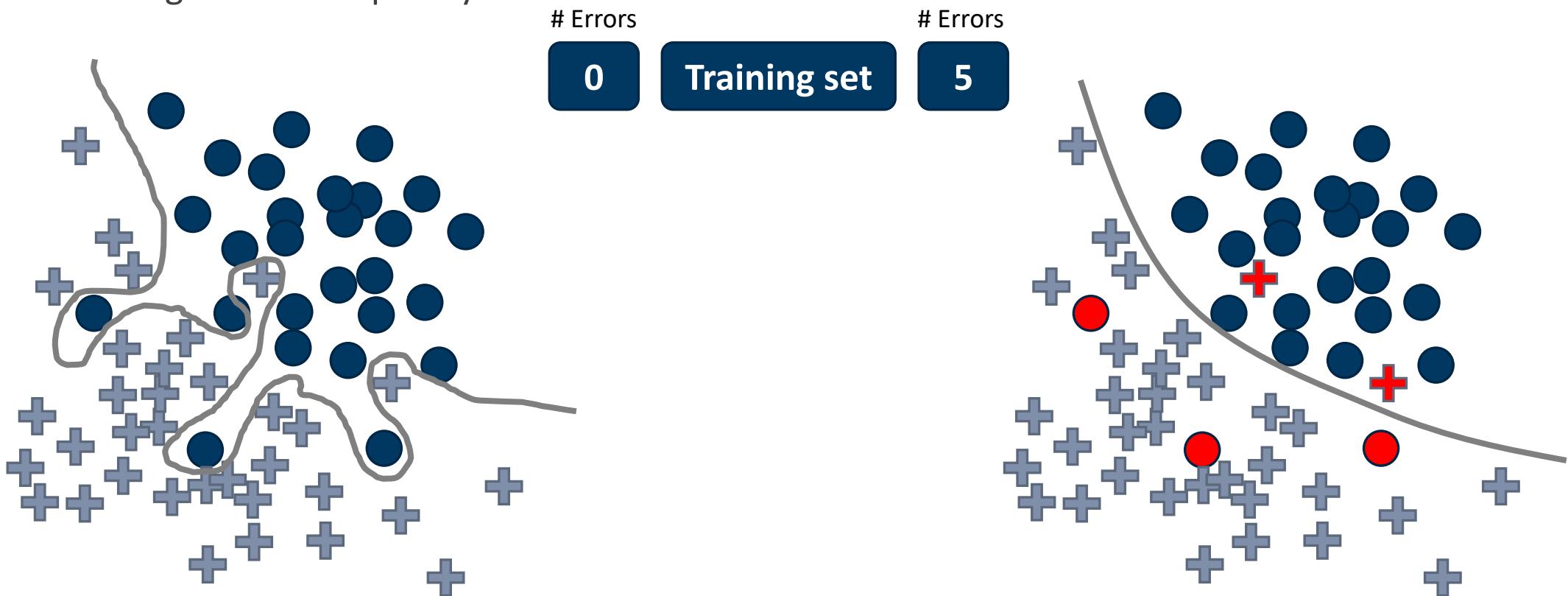
Empirical risk minimization (ERM):

$$R_{\text{emp}}(h) = \frac{1}{N} \sum_{n=1}^N L(h(x_n), y_n)$$

$$\hat{h} = \arg \min_{h \in \mathcal{H}} R_{\text{emp}}(h)$$

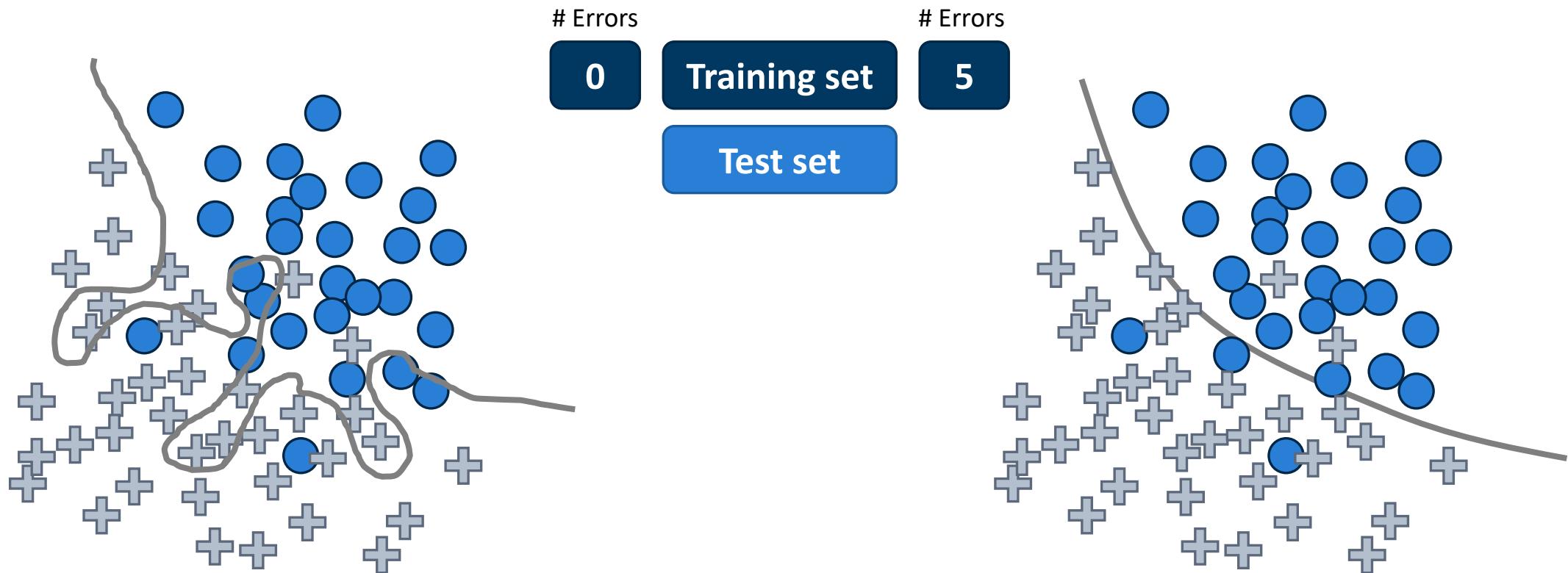
Problem with Empirical Risk Minimization

If hypothesis space \mathcal{H} is not constrained, empirical risk R can be made arbitrarily small by increasing model complexity



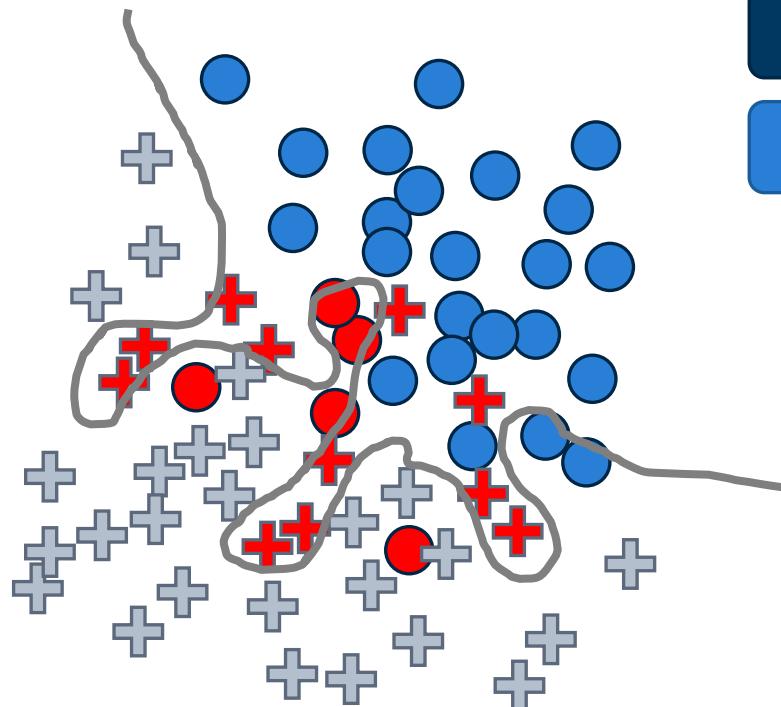
Problem with Empirical Risk Minimization

What happens when we deploy our classifier and test it on new data?

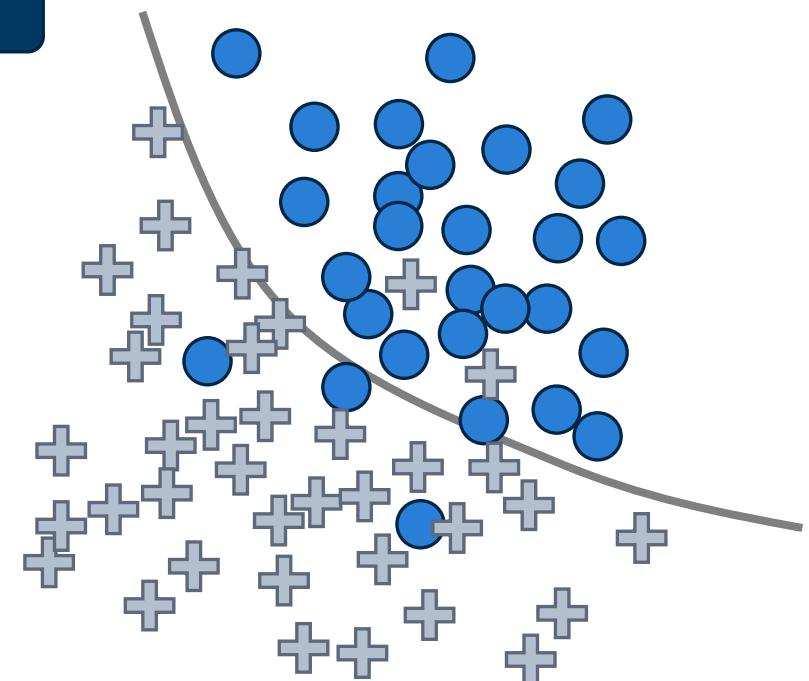


Problem with Empirical Risk Minimization

Does not generalize!

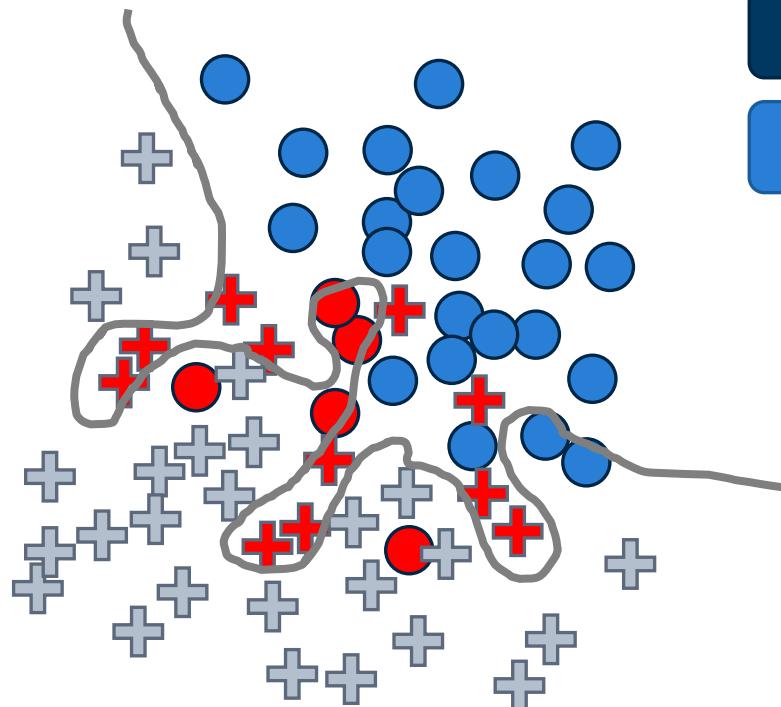


# Errors	Training set	Test set
0		
16		
	Training set	
	Test set	
5		



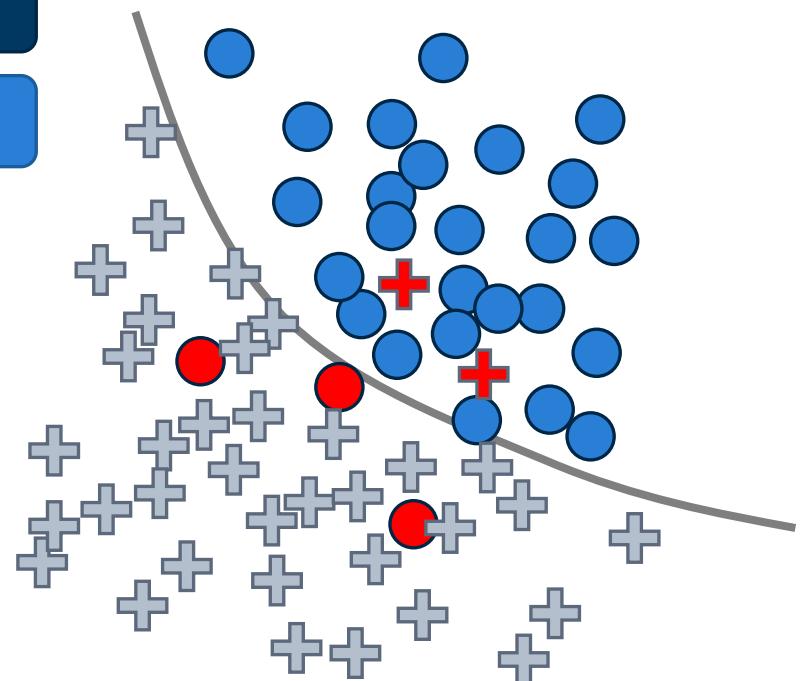
Problem with Empirical Risk Minimization

Does not generalize!



# Errors	Training set	# Errors
0	Training set	5
16	Test set	5

Generalizes well



What does this mean in practice?

Goal: Generalization

Machine learning is about generalization, not about optimization

CONTROL MODEL COMPLEXITY

A number of techniques exist:

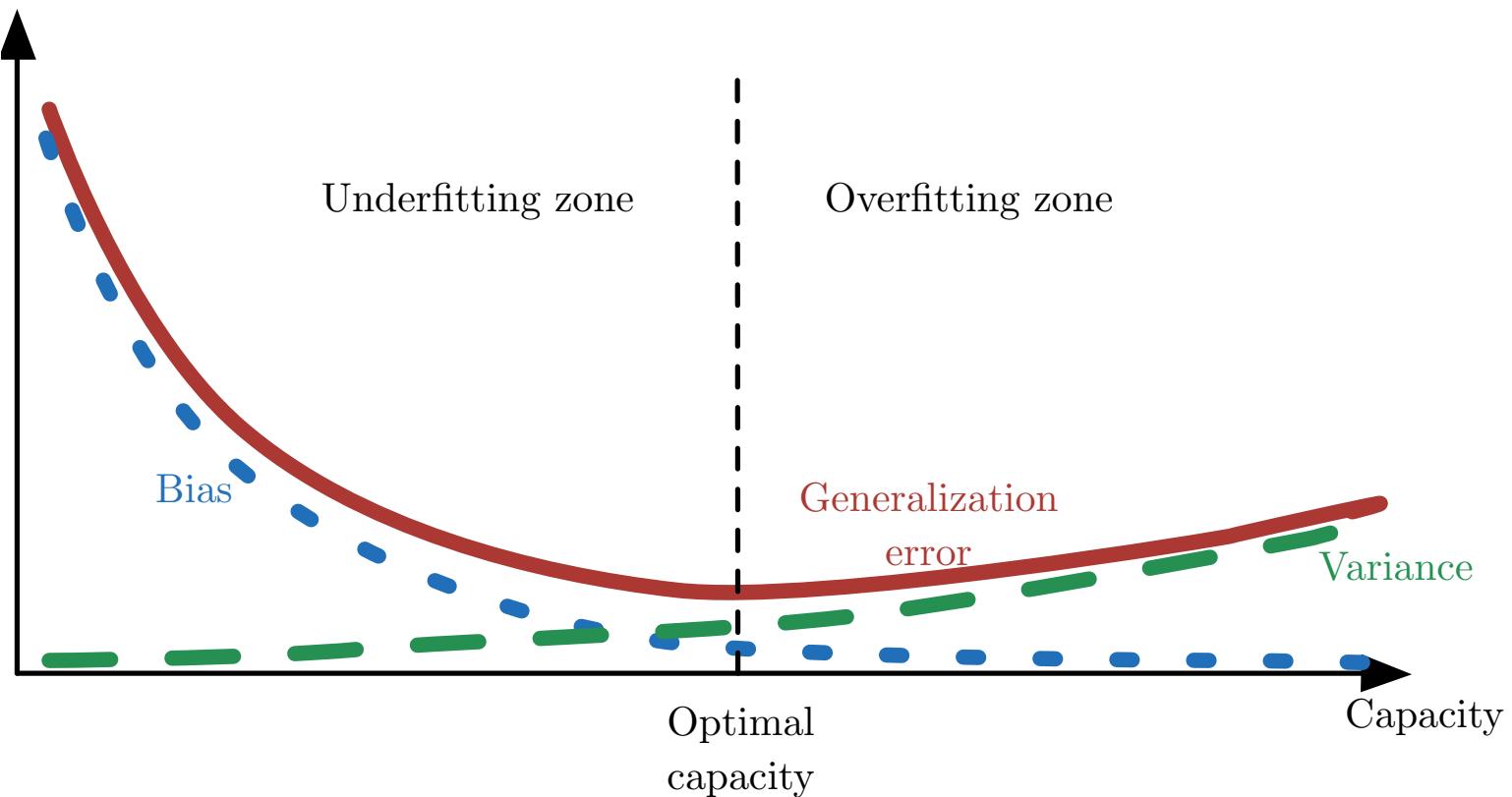
- Reduce number of parameters
- Regularization
- Early stopping

ESTIMATE GENERALIZATION ERROR

Split dataset into

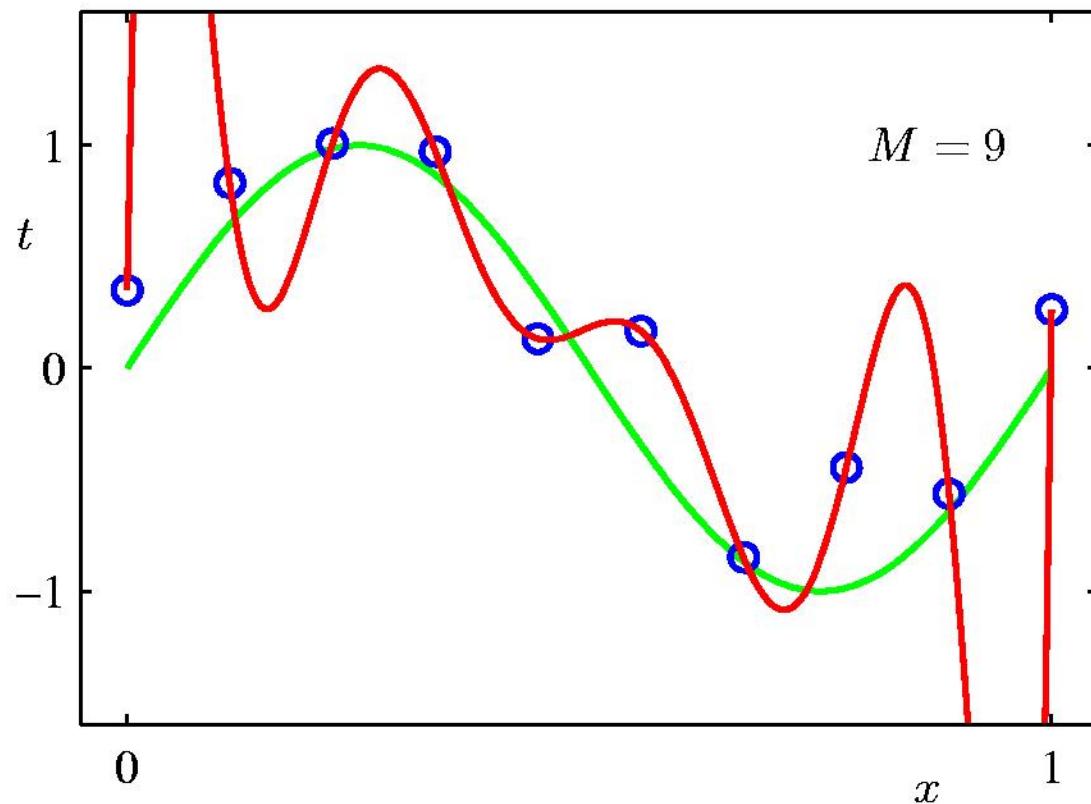
- **Training set** used to train the algorithm using empirical risk minimization (ERM)
- **Test set** used to estimate the true risk

Bias and variance

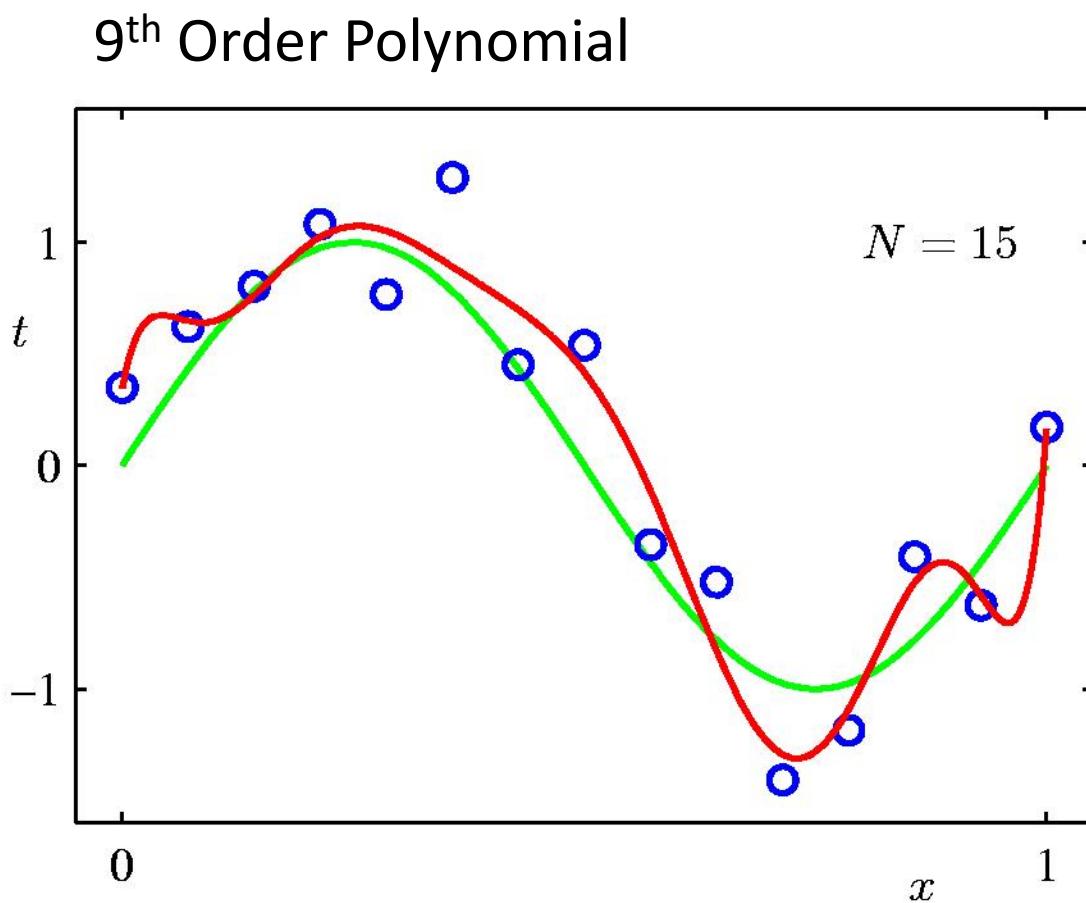


Data Set Size: $N = 10$

9th Order Polynomial

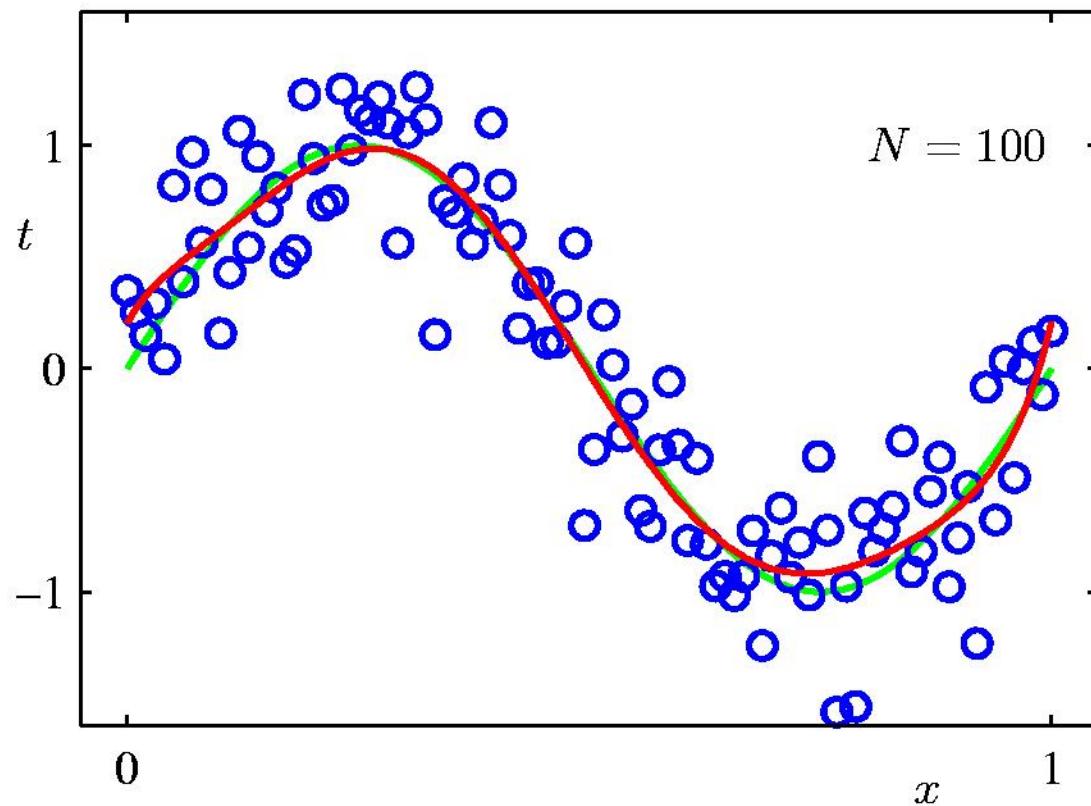


Data Set Size: $N = 15$



Data Set Size: $N = 100$

9th Order Polynomial



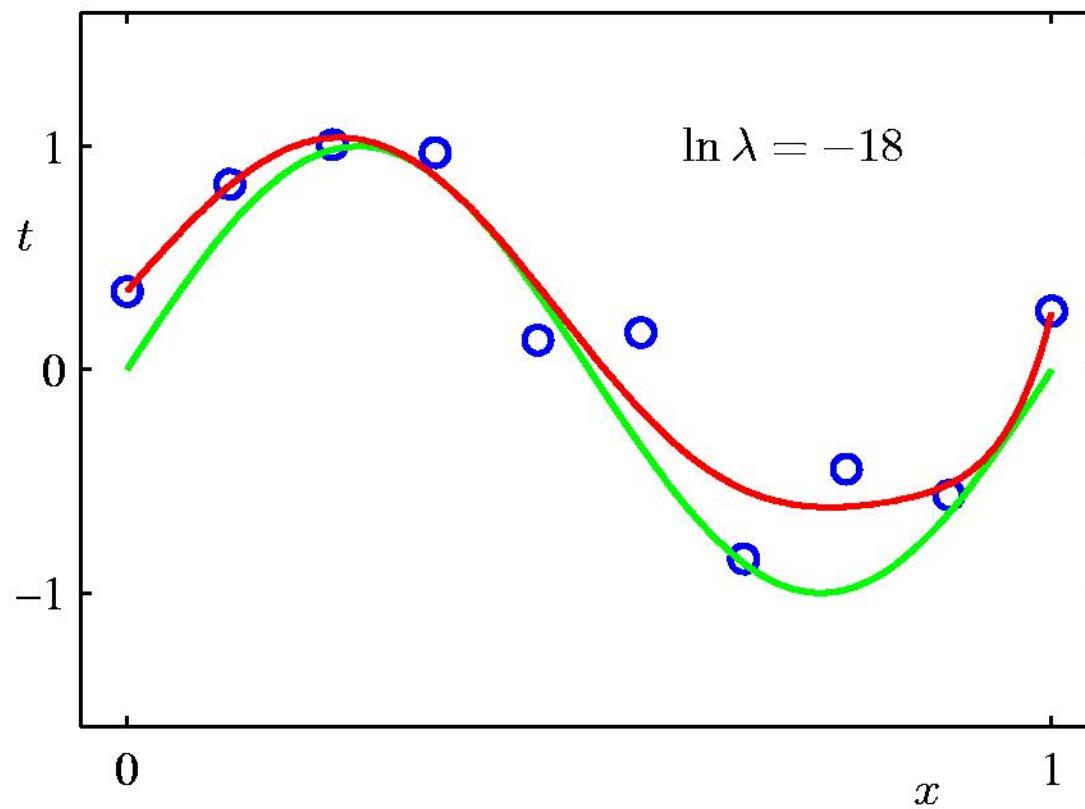
Regularization

Regularization

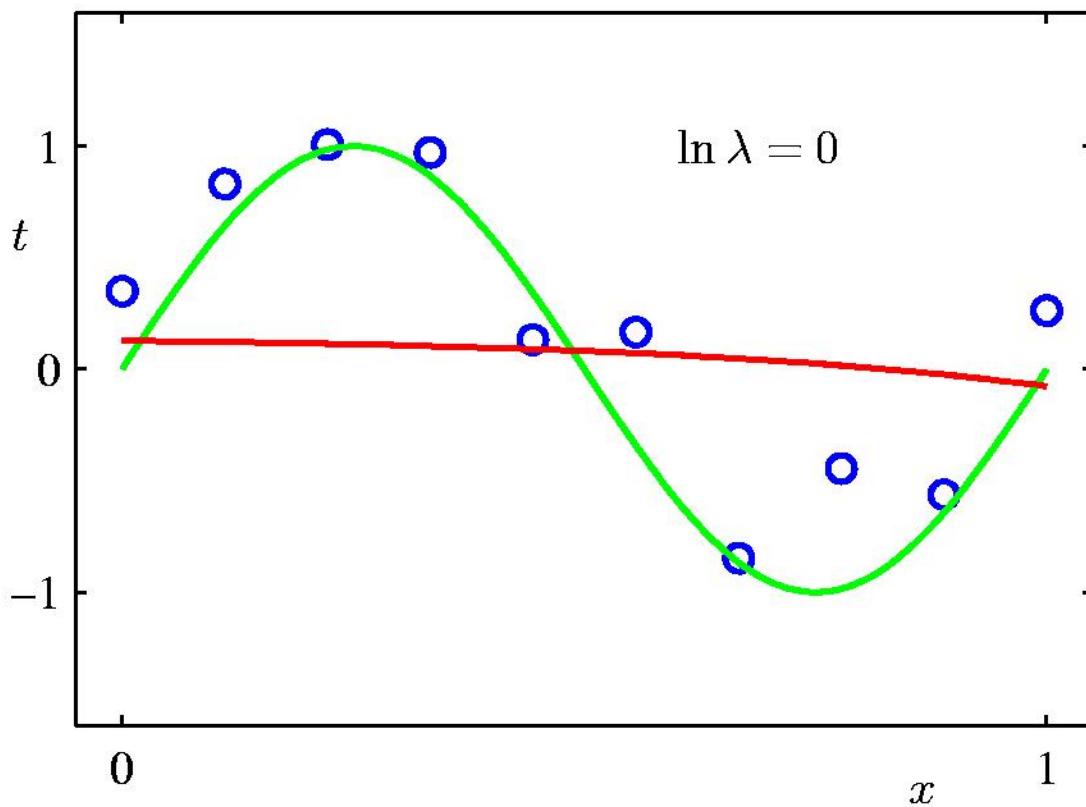
Penalize large coefficient values

$$\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

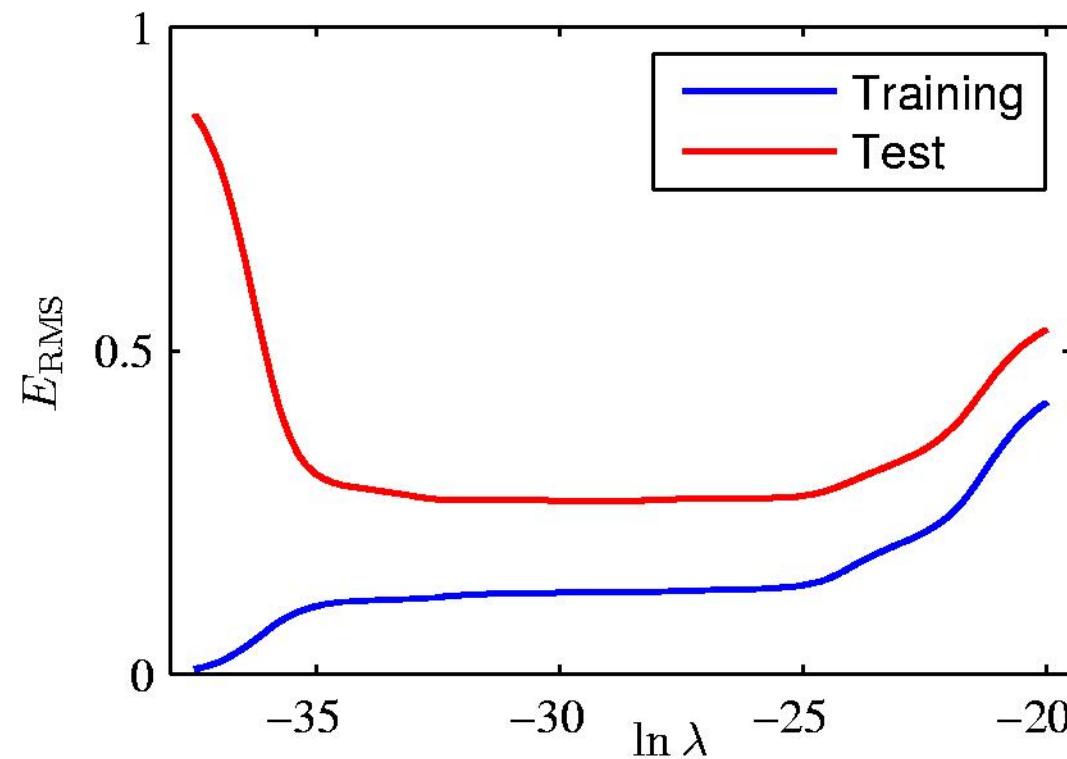
Moderate Regularization



Regularization too strong



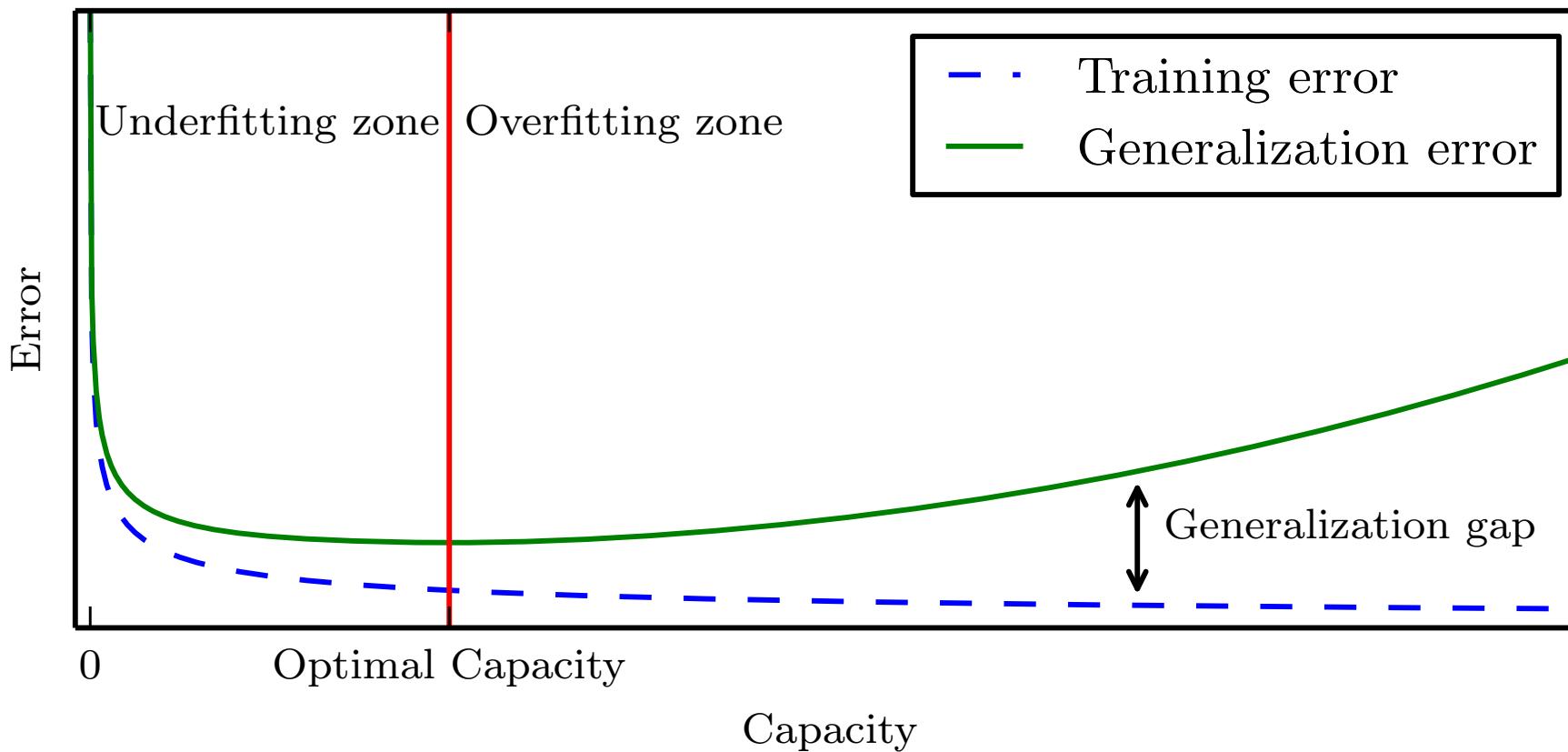
Squared error vs. regularization strength



Polynomial Coefficients

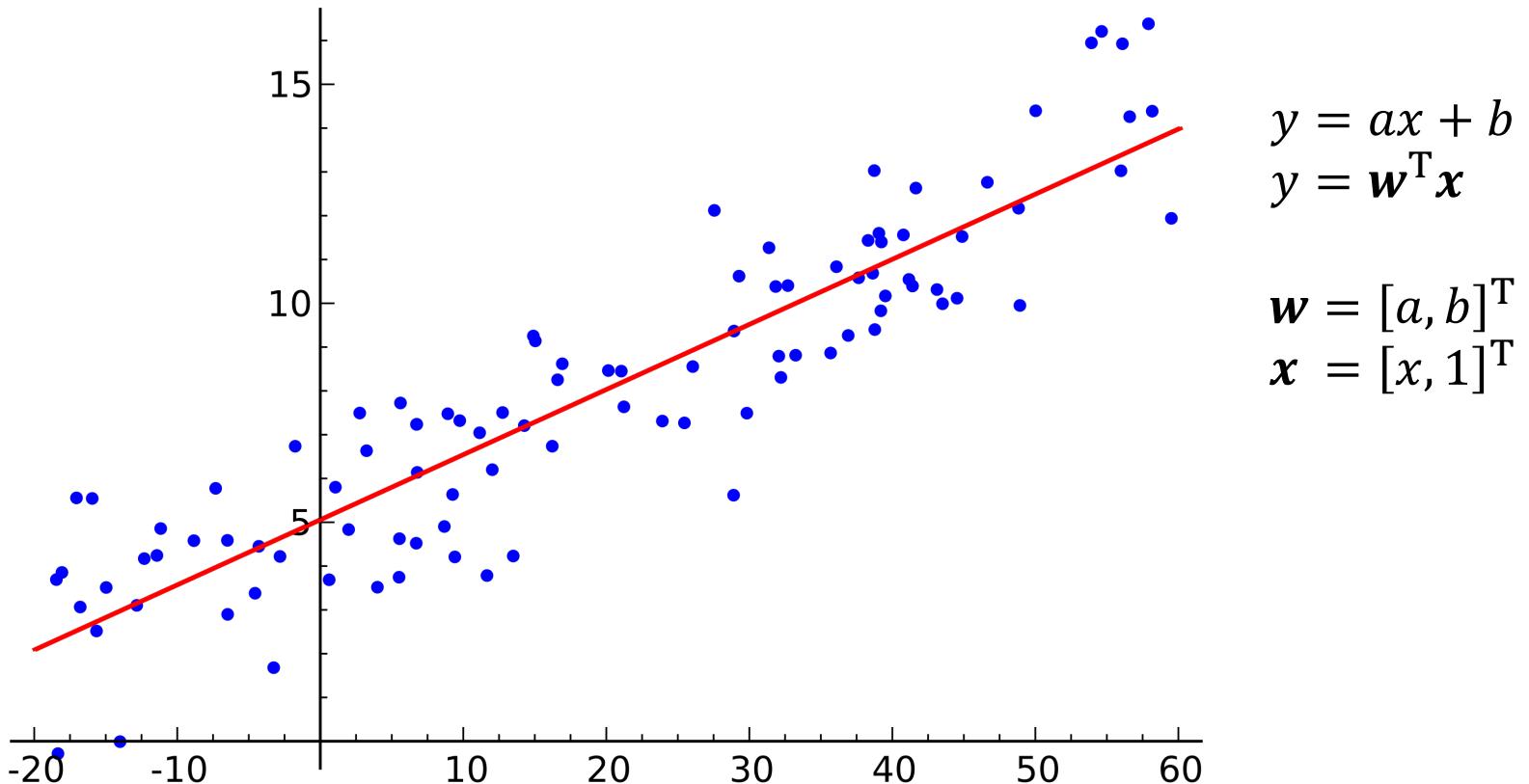
	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
w_0^*	0.35	0.35	0.13
w_1^*	232.37	4.74	-0.05
w_2^*	-5321.83	-0.77	-0.06
w_3^*	48568.31	-31.97	-0.05
w_4^*	-231639.30	-3.89	-0.03
w_5^*	640042.26	55.28	-0.02
w_6^*	-1061800.52	41.32	-0.01
w_7^*	1042400.18	-45.95	-0.00
w_8^*	-557682.99	-91.53	0.00
w_9^*	125201.43	72.68	0.01

Generalization and capacity



Linear regression

Linear regression



Maximum Likelihood and Least Squares (1)

Assume observations from a deterministic function $t = \mathbf{w}^T \mathbf{x} + \epsilon$ with added Gaussian noise:

$$p(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1})$$

which is the same as saying,

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|\mathbf{w}^T \mathbf{x}, \beta^{-1})$$

Given observed inputs, $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]$, and targets, $\mathbf{t} = [t_1, \dots, t_N]$, we obtain the likelihood function

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^N \mathcal{N}(t_n | \mathbf{w}^T \mathbf{x}_n, \beta^{-1})$$

Maximum Likelihood and Least Squares (2)

Taking the logarithm, we get

$$\begin{aligned}\log p(t|\mathbf{X}, \mathbf{w}, \beta) &= \sum_{n=1}^N \mathcal{N}(t_n | \mathbf{w}^\top \mathbf{x}_n, \beta^{-1}) \\ &= \frac{N}{2} \log \beta - \frac{N}{2} \log 2\pi - \beta L(\mathbf{w})\end{aligned}$$

where

$$L(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^\top \mathbf{x}_n\}^2$$

is the sum-of-squares error.

Maximum Likelihood and Least Squares (3)

Computing the gradient and setting it to zero yields

$$\nabla_{\mathbf{w}} \log p(\mathbf{t}|\mathbf{w}, \beta) = \beta \sum_{n=1}^N \{t_n - \mathbf{w}^T \mathbf{x}_n\} \mathbf{x}_n^T = 0$$

Solving for \mathbf{w} , we get

$$\mathbf{w}_{\text{ML}} = \boxed{(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{t}}$$

The Moore-Penrose
pseudo-inverse, Φ^\dagger .

Probabilistic view on regularization

Bayles rule

$$P(\mathbf{w}|\mathbf{x}) = \frac{P(\mathbf{x}|\mathbf{w})P(\mathbf{w})}{P(\mathbf{x})}$$

Maximum a-posteriori (MAP) estimate of \mathbf{w} :

$$\arg \max_{\mathbf{w}} P(\mathbf{w}|\mathbf{x})$$

$$P(\mathbf{w}|\mathbf{x}) \propto P(\mathbf{x}|\mathbf{w})P(\mathbf{w})$$

Linear regression with Gaussian prior on weights

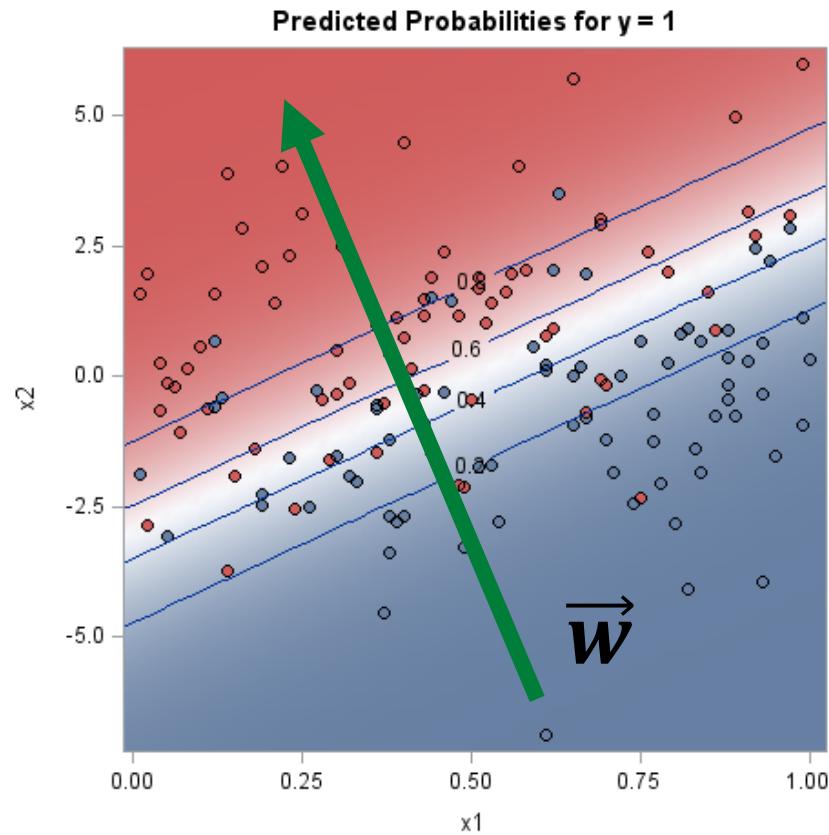
Suppose $P(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$

Find MAP by minimizing $L(\mathbf{w}) - \log P(\mathbf{w}|\mathbf{x})$

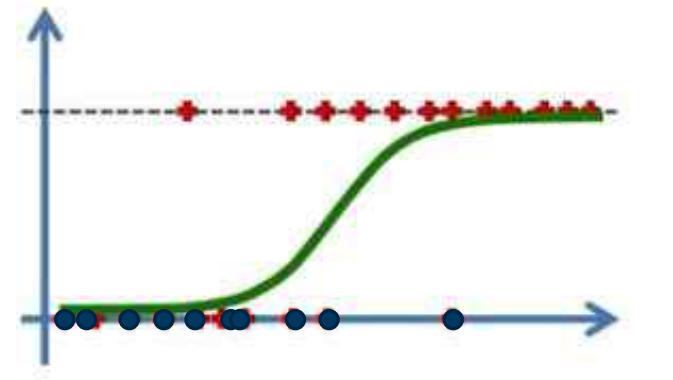
$$\begin{aligned}-\log P(\mathbf{w}|\mathbf{x}) &= -\log P(\mathbf{x}|\mathbf{w}) - \log \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}) \\&= -\log P(\mathbf{x}|\mathbf{w}) - \log \left(\exp \left(-\frac{\mathbf{w}^\top \mathbf{w}}{2\sigma^2} \right) \right) \\&= -\frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^\top \mathbf{x}_n\}^2 - \frac{1}{2\sigma^2} \|\mathbf{w}\|_2^2\end{aligned}$$

Logistic regression

Logistic regression



Project onto \vec{w}



Logistic regression model

Model the probability of ‘on’ ($y = 1$) as

$$P(y = 1|x, w) := \sigma(w^T x) := \frac{1}{1 + \exp(-w^T x)} := p$$

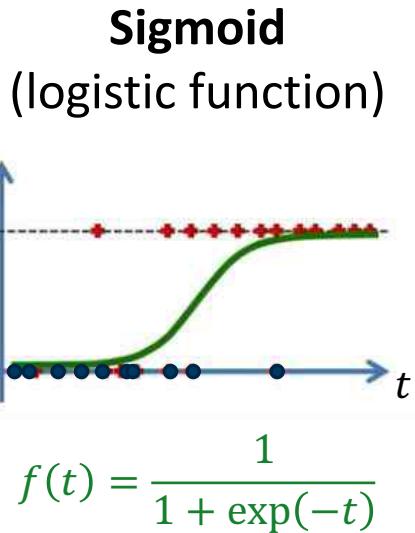
Probability of ‘off’ ($y = 0$):

$$P(y = 0|x, w) = 1 - p = 1 - \sigma(w^T x) = \sigma(-w^T x)$$

Loss function (negative log-likelihood):

$$P(Y|X, w) = \prod_n p_n^{y_n} (1 - p_n)^{1 - y_n}$$

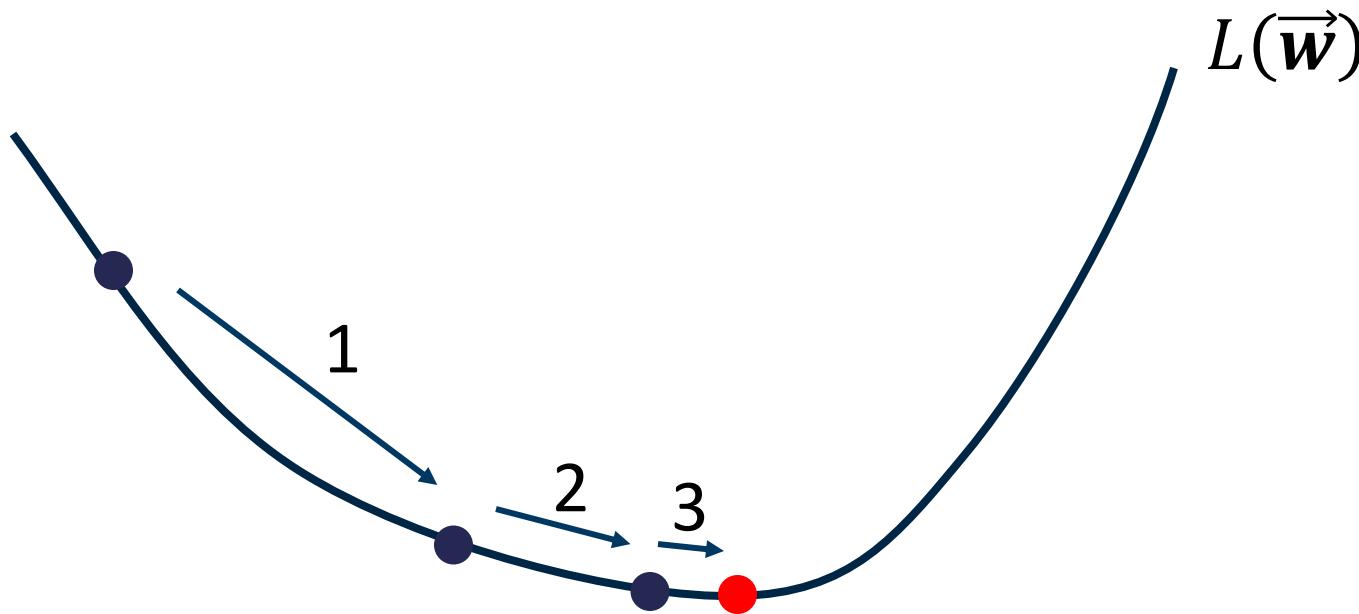
$$L = -\log P(Y|X, w) = -\sum_n y_n \log \sigma(w^T x_n) + (1 - y_n) \log \sigma(-w^T x_n)$$



Solving logistic regression

Optimal solution can be found via *iterative reweighted least squares*

Convex loss function → any gradient-based optimizer will find optimum



Gradient descent

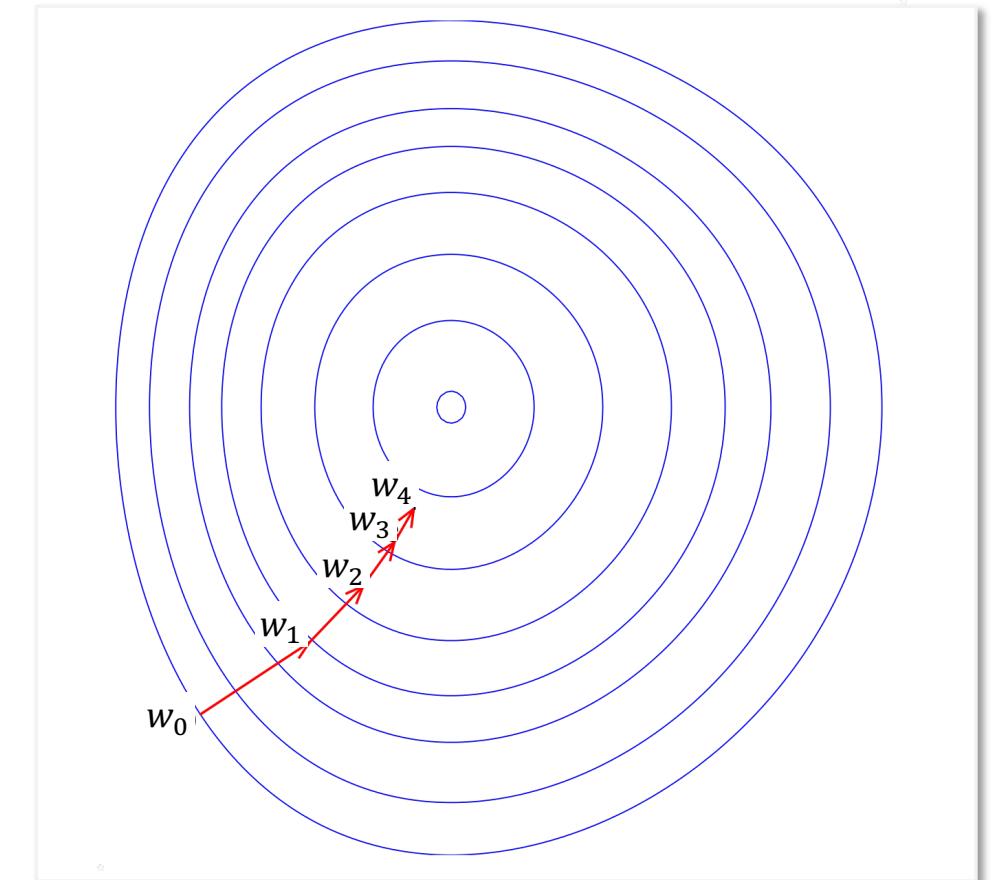
We use gradient descent to minimize the loss

Start with a random guess for the parameters w

Iteratively update w by a small step in the direction of the gradient of the loss function

$$w_{t+1} \leftarrow w_t - \lambda \nabla_w L(w_t)$$

Learning rate

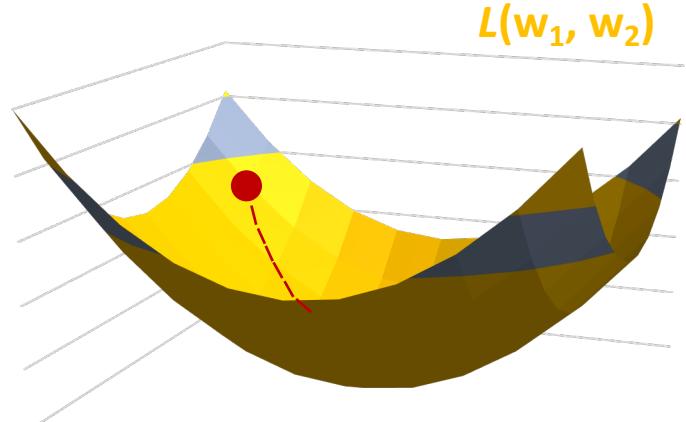


Source: Wikipedia

The three steps of gradient descent

Intuition

Iteratively move along the direction of steepest descent.



Application in an example

Repeat until no further improvement

1

Initialize with random weights and biases

2

Compute gradient of the loss function

3

Update weights and biases by taking a small step in the direction of the gradient

$$\mathbf{w} = \begin{bmatrix} 4,8 \\ -2,3 \\ 0,7 \\ \vdots \\ -0,2 \\ 1,5 \end{bmatrix}$$



$$-\nabla L(\mathbf{w}) = \begin{bmatrix} -0,7 \\ 1,2 \\ 5,3 \\ \vdots \\ -1,7 \\ -0,4 \end{bmatrix}$$



$$\mathbf{w} = \begin{bmatrix} 4,1 \\ -1,1 \\ 6,0 \\ \vdots \\ -1,9 \\ 1,1 \end{bmatrix}$$

Questions?
