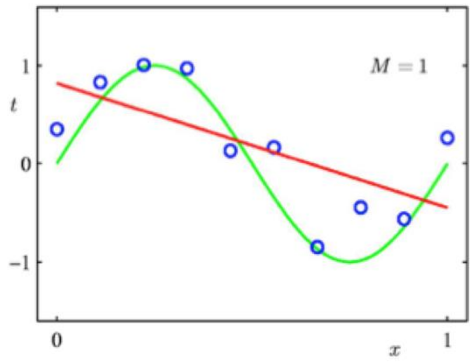
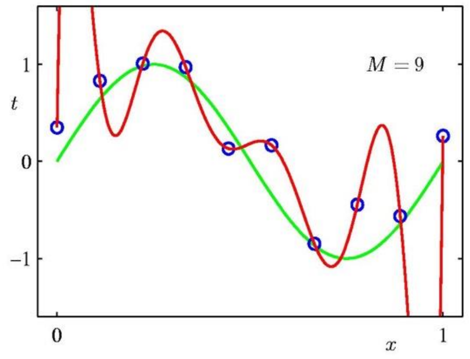
Machine Learning
Overfitting, Model Complexity, Regularization

Berrin Yanikoglu
Sabancı Üniversitesi

Overfitting and Model Complexity

- Imagine that we have some training data and we want to learn the underlying function between the independent variable x and the target values t.
- We can fit polynomials in varying degrees: lines to higher degree polynomials.
  - Higher degrees make the polynomial very capable to bend/flex to match the data as it has many parameters to change/adapt.
- However having zero train error does not mean the model (high order polyn.) will also have high generalization performance.
- In fact, a simpler model that has a similar performance compared to a more complex model, is often preferred (more on these later and more formally).



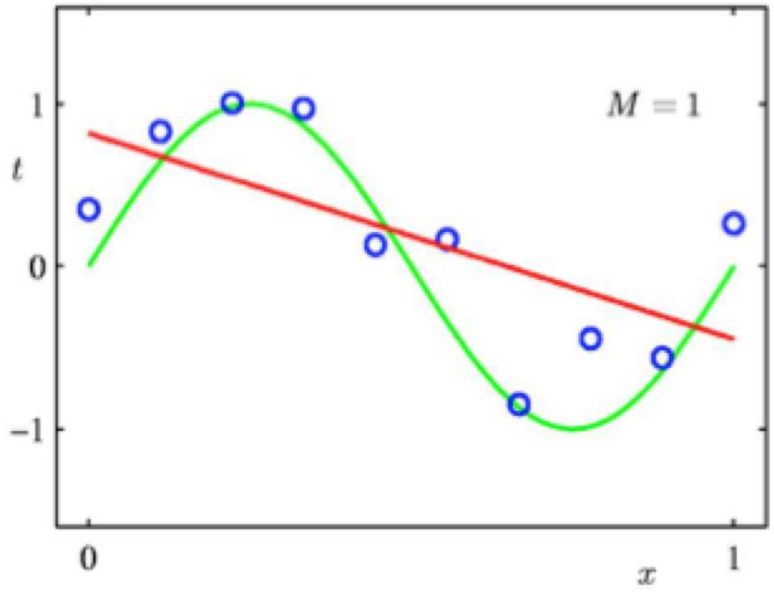


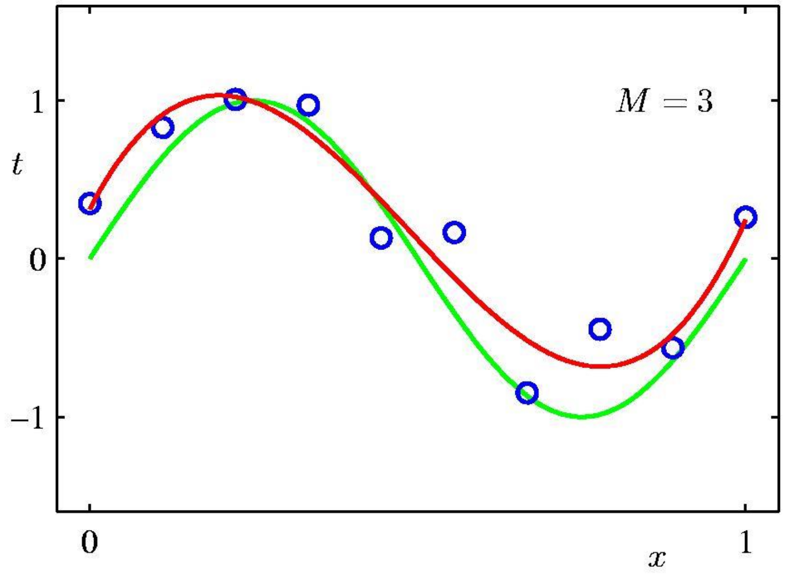
1st Order Polynomial

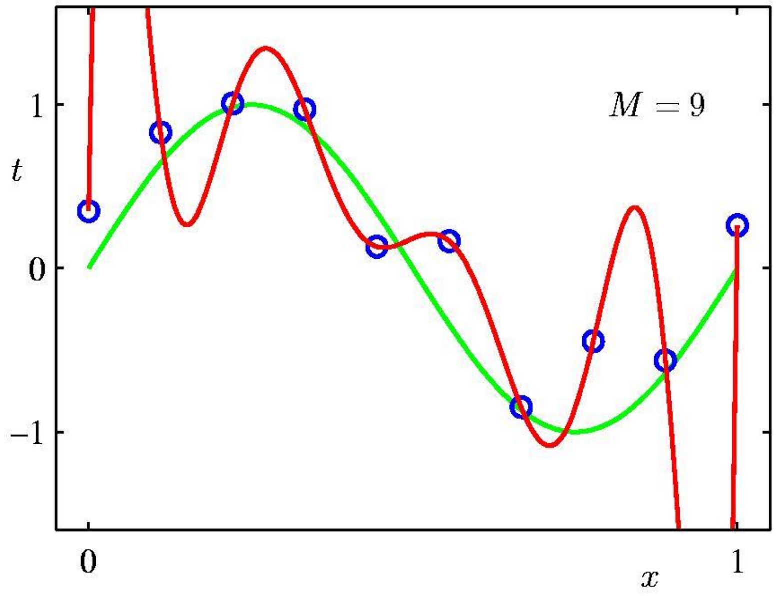
- Let’s represent this function with polynomials of varying degree:

$$y(x) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \underset{j=0}{\overset{M}{\square}} w_j x^j$$

1st Order Polynomial



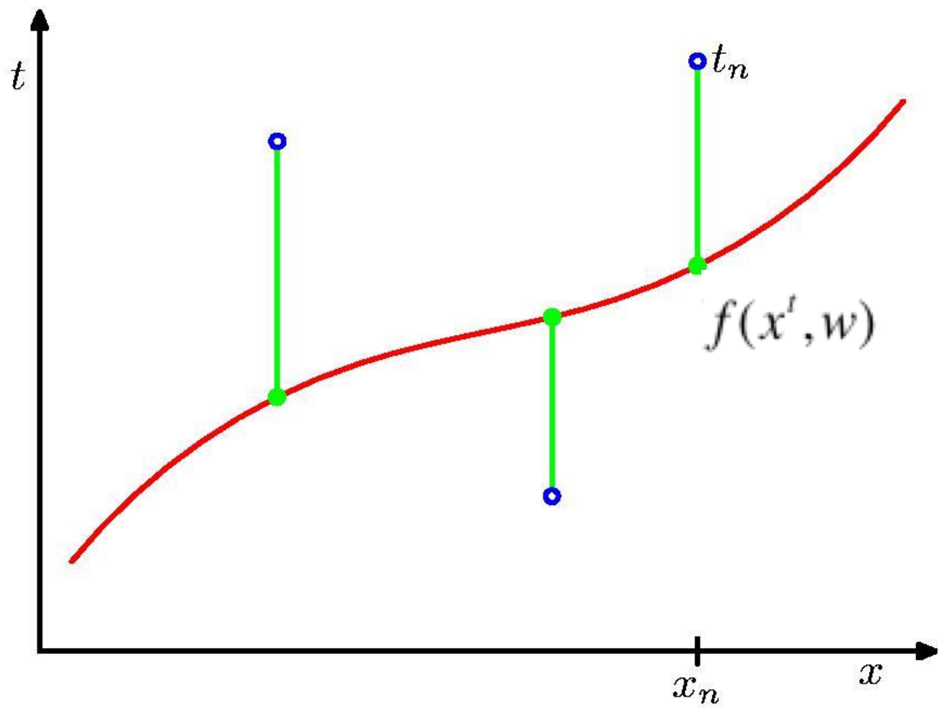




- We do not know yet which is the best model, maybe the 9th degree polynomial after all.

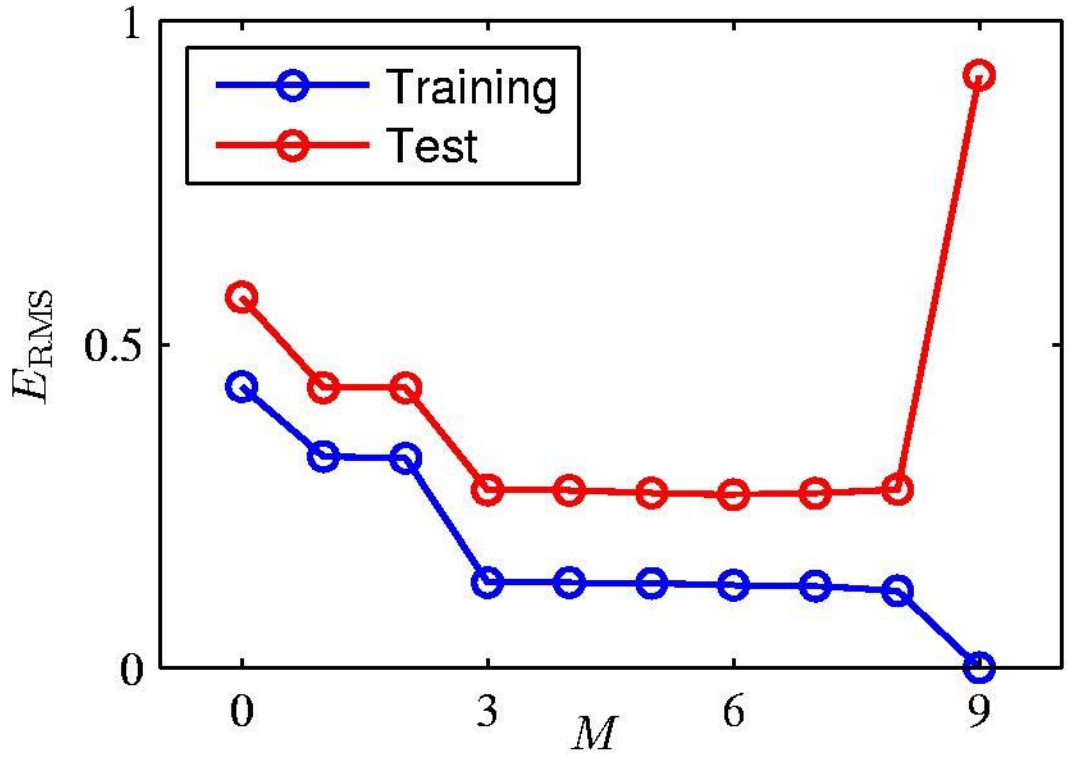
- The main/typical approach is to use the validation set performance to decide which model to choose.

Sum-of-Squares Error Function



$$E(\mathbf{w}) = \frac{1}{2} \mathop{\square}\limits_{i=1}^{N} (y(x^i) - t^i)^2$$

Over-fitting



Root-Mean-Square (RMS) Error:

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$$

Polynomial Coefficients

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

Overfitting

Formal definition:

A hypothesis f is said to overfit the training data
if there exists another hypothesis, f’,
such that f has smaller error than f’ on the training data,
but f’ has smaller error on the test data than f.

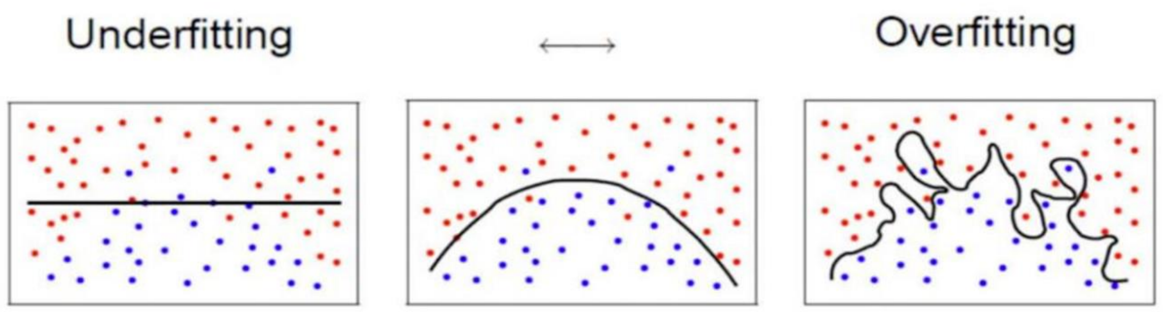
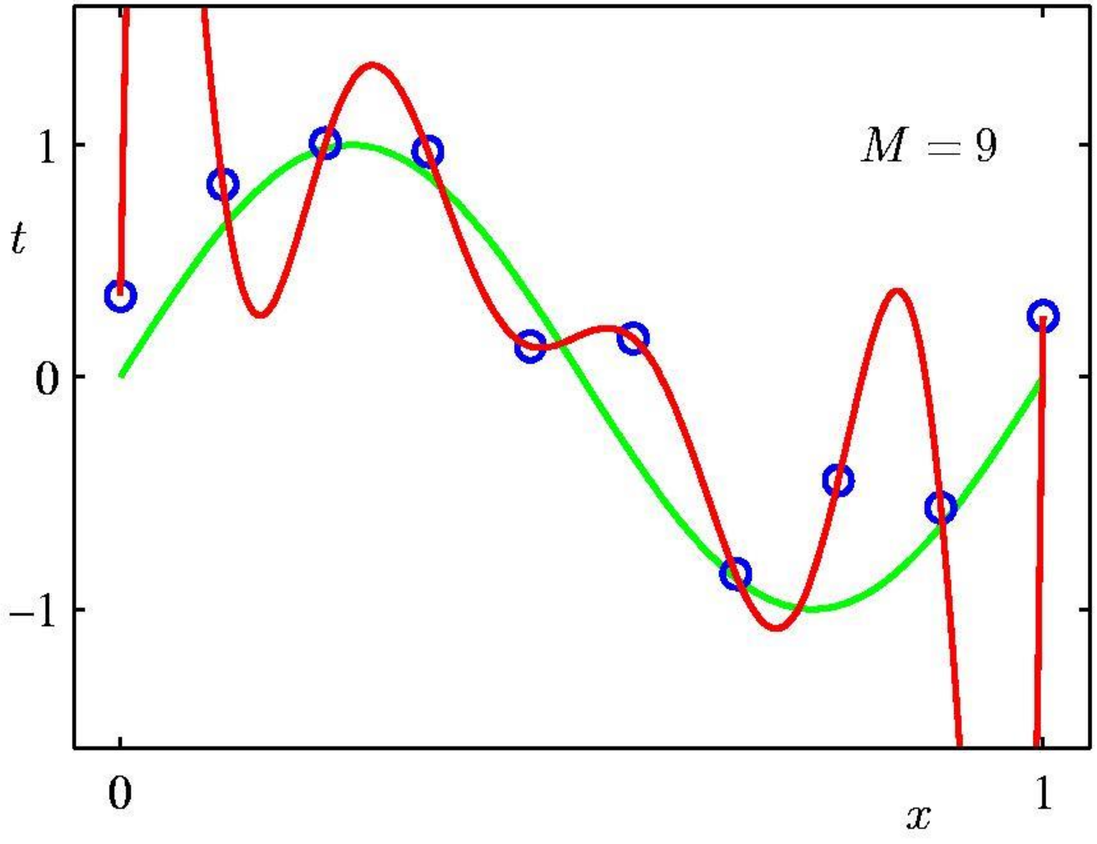


Image Credit: https://tomrobertshaw.net/2015/12/introduction-to-machine-learning-with-naive-bayes

Effect of Data

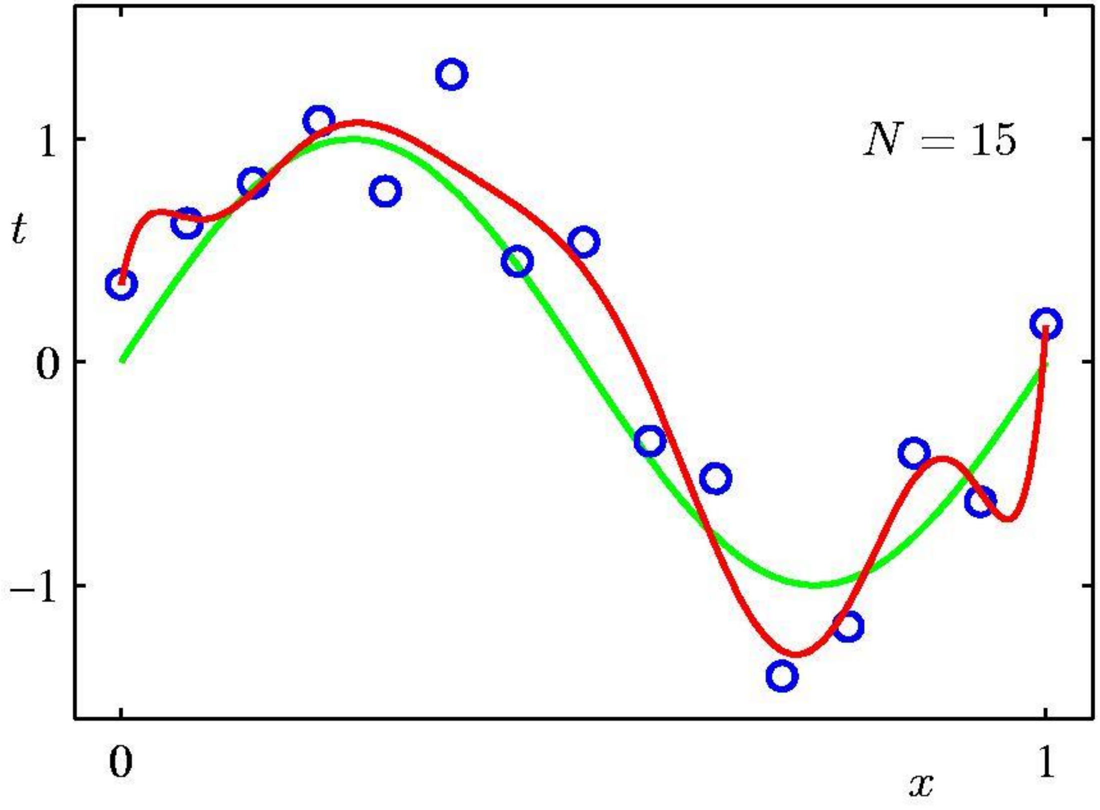
N=10 samples

9th Order Polynomial



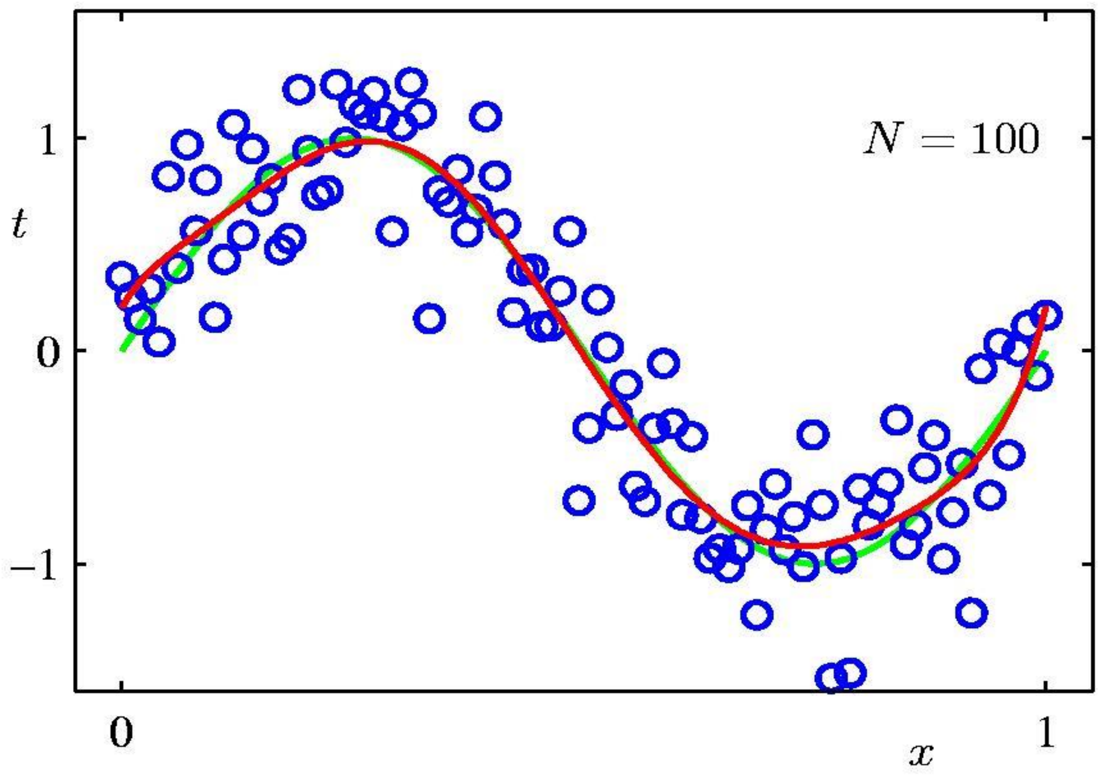
N=15 samples

9th Order Polynomial



N=100 samples

9th Order Polynomial



As data increases, the polynomial is further bound, reducing wild changes.

Regularization

Regularization

- Use complex models, but penalize large coefficient values:

$$Min_{w,b}\,(MSE + penalty) = Min\left[\underbrace{\frac{1}{N}\sum_{i=1}^{N}\left(y_i - f_{w,b}(X_i)\right)^2}_{\textbf{Fit data}} + \underbrace{penalty(w)}_{\textbf{Regularize}}\right]$$

Lasso (L1 Regularization):

$$\text{minimize}\left(\frac{1}{N}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i;\mathbf{w}))^2 + \lambda\sum_{j=1}^{p}|w_j|\right)$$

Ridge (L2 Regularization):

$$\text{minimize}\left(\frac{1}{N}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i;\mathbf{w}))^2 + \lambda\sum_{j=1}^{p}w_j^2\right)$$

Regularization

- Use complex models, but penalize large coefficient values:

$$Min_{w,b}\ (MSE + penalty) = Min\left[\underbrace{\frac{1}{N}\sum_{i=1}^{N}\left(y_i - f_{w,b}(X_i)\right)^2}_{\text{Fit data}} + \underbrace{penalty(w)}_{\text{Regularize}}\right]$$

Lasso (L1 Regularization):

$$\text{minimize}\left(\frac{1}{N}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i;\mathbf{w}))^2 + \lambda\sum_{j=1}^{p}|w_j|\right)$$

Ridge Regression (L2 Regularization)

$$\text{minimize}\left(\frac{1}{N}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i;\mathbf{w}))^2 + \lambda\sum_{j=1}^{p}w_j^2\right)$$

The tuning parameter λ serves to control the relative impact of the penalty term on the regression coefficient estimates.

Lasso Regression

Lasso (L1 Regularization):

$$\text{minimize}\left(\frac{1}{N}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i;\mathbf{w}))^2 + \lambda\sum_{j=1}^{p}|w_j|\right)$$

- LASSO stands for "Least Absolute Shrinkage and Selection Operator"
- LASSO eliminates the least important features from the model, it automatically performs a type of **feature selection**.
- It is best to apply regularization after variable standardization (weights should be of comparable scale, as the L1 or L2 norm is minimized as a whole).
- λ should be chosen using cross-validation

Ridge Regression

Ridge Regression (L2 regularization)

$$\text{minimize}\left(\frac{1}{N}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i;\mathbf{w}))^2 + \lambda\sum_{j=1}^{p} w_j^2\right)$$

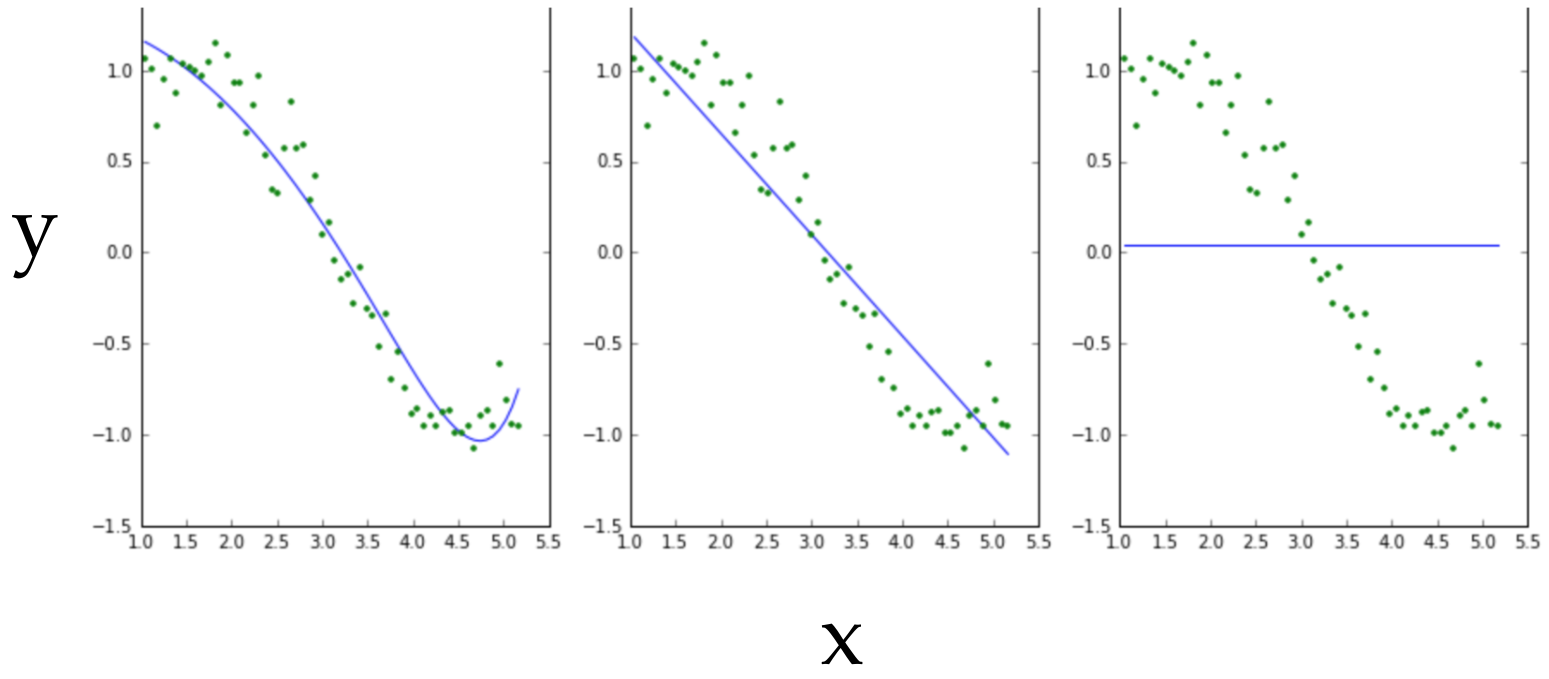
- The shrinkage penalty has the effect of shrinking the estimates of ***w*** towards zero (not selecting one in case of multiple colinear features)

- It is best to apply regularization after variable standardization (weights should be of comparable scale, as the L1 or L2 norm is minimized as a whole).

- λ should be chosen using cross-validation

Effect of Lambda

- Blue line is the predicted function.
- Green dots are the training data
- As lambda increases the model becomes simpler.



- When there are multiple colinear features, **L1 regularization (Lasso)** will often set the weight of one to zero. So, it is often used for feature selection and model interpretability (you may ignore features with zero weights).

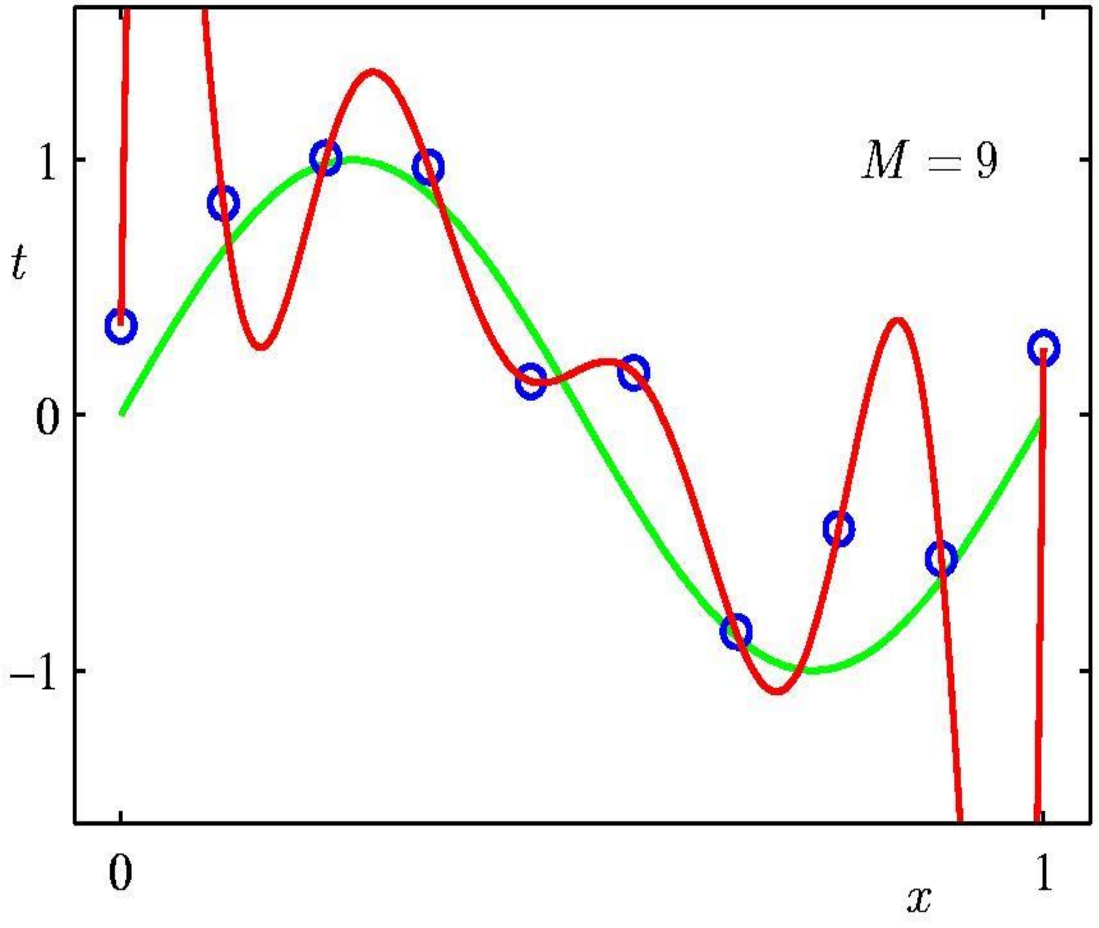
- **L2 regularization (Ridge)** will decrease the weights of two colinear features, w/o setting one to zero, so not used for feature selection or simplicity, but may be better in terms of generalization.

Regularization on 9th Order Polynomial

Example from the Bishop book.

$\ln \lambda$ = -inf

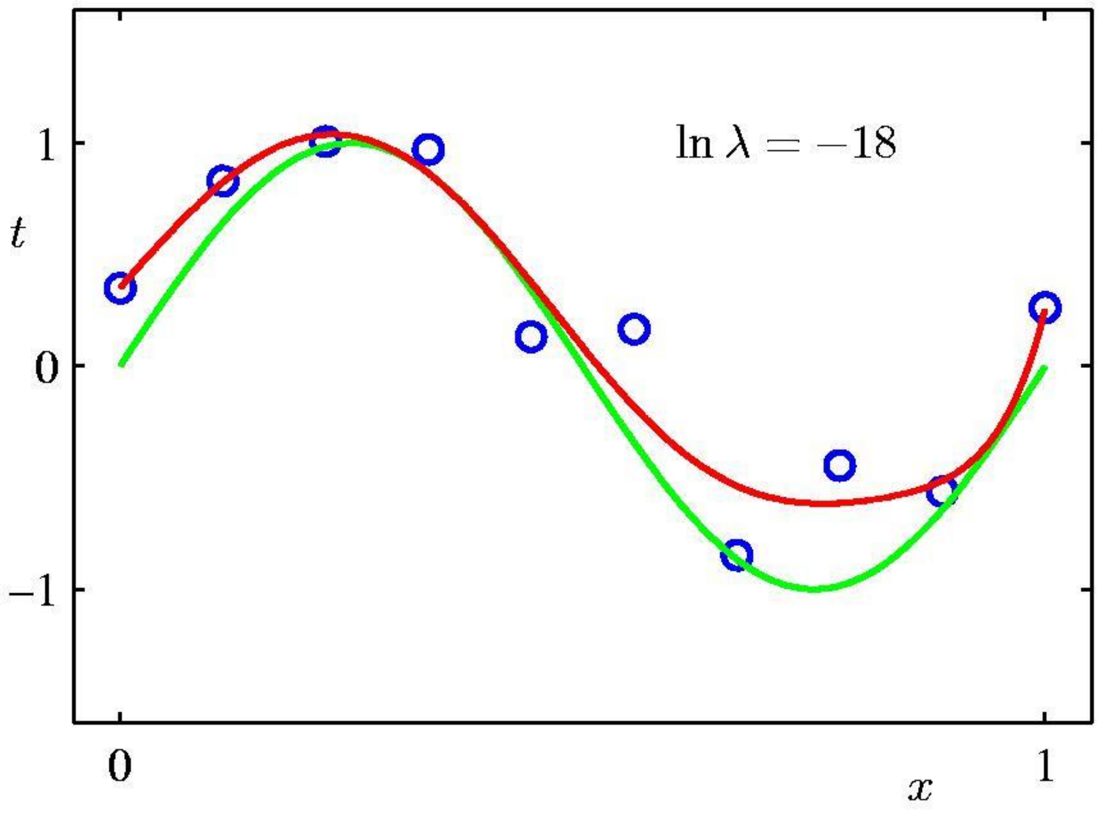
Too small λ – no regularization effect



Regularization on 9th degree polynomial:

$\ln \lambda = -18$

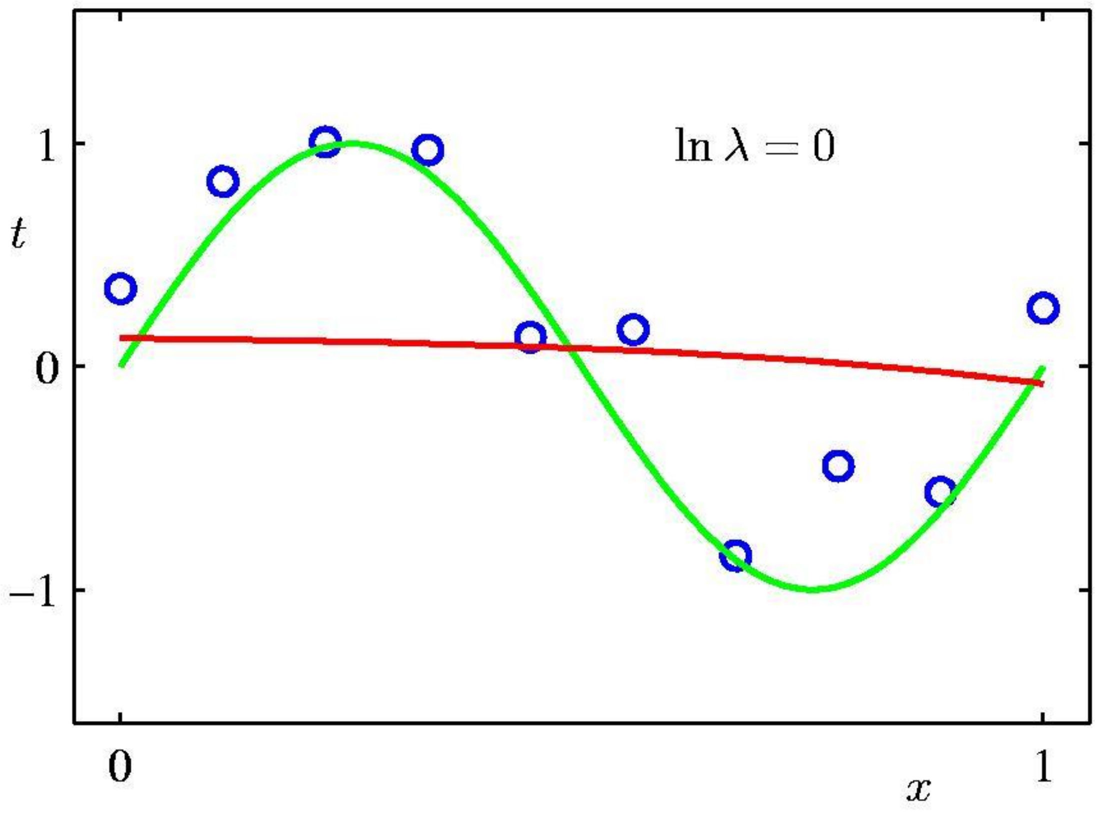
Right λ –good fit



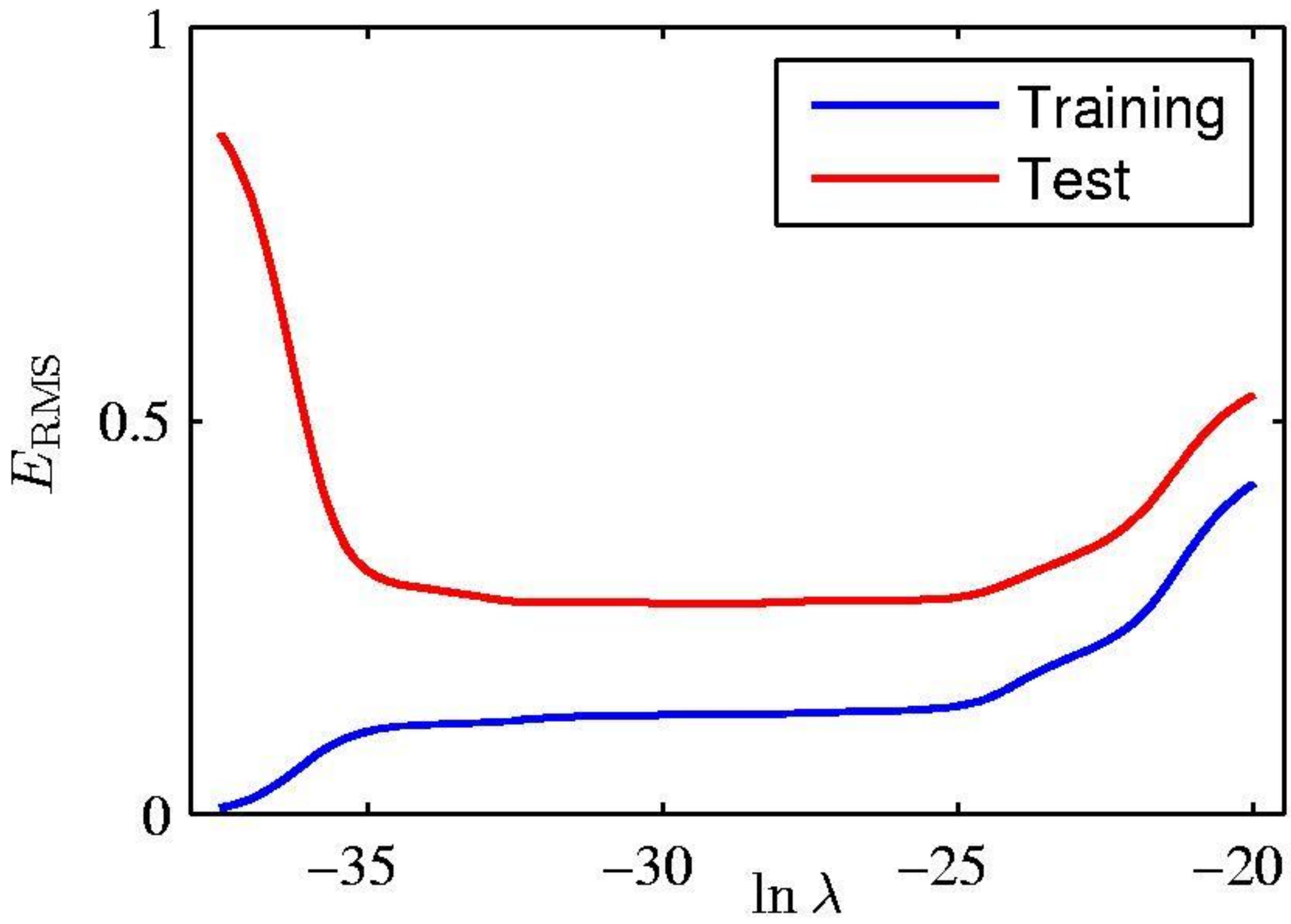
Regularization:

$\ln \lambda = 0$

Large λ –regularization dominates



Regularization: E_{RMS} *vs.* $\ln \lambda$



Polynomial Coefficients

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

Very little

Too much

Regularization

How to select the polynomial degree and regularization coefficient?

Grid search:

```
for degree d in [1, 2, 3]:
    //inner loop – compute val error for each $\lambda$
    for $\lambda$ in [0, 0.1, 1, 10, ...]:
        train model(degree, $\lambda$) on training set
        val_error(d, $\lambda$ ) = validation error of the model
    //select best $\lambda$ for degree
    min_val_error(d) = min $_{\lambda}$ val_error(d, $\lambda$ )

//Select best model
min_val_error = min $_d$ val_error(d)
```

What You Should Know

- Different regression types (linear, polynomial, multiple linear)
- Least Squares and Gradient Descent solutions to regression
- When you may choose one or the other or how to choose the right complexity
  - You may choose a linear model if you have some prior expectation for the linearity
  - If you have large amounts of data, high complexity models may not pose a problem
- Regularization
  - You should use regularization to control your model complexity
- Final model selection should be done via grid search over hyperparameters (degree and lambda)