7.1 Overfitting

Machine Learning 1: Foundations

Marius Kloft *(TUK)*

Contents of this Class

Overfitting and Regularization

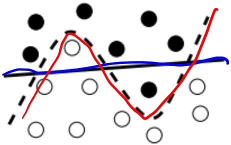
Overfitting Dilemna

Given too few data we do not know which classifier works better: a simple or complex one.

Overfitting Dilemna

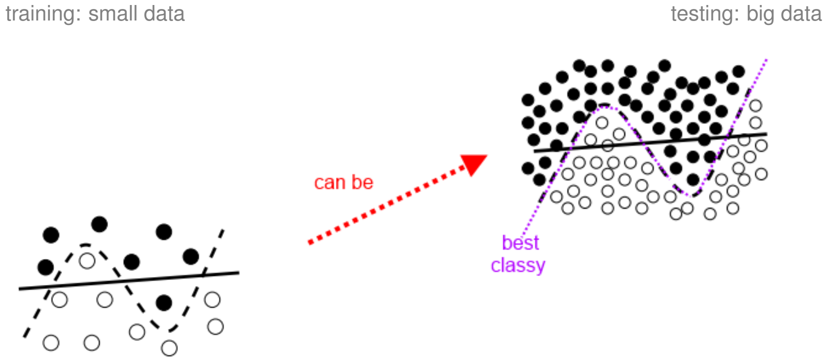
Given too few data we do not know which classifier works better: a simple or complex one.

training: small data



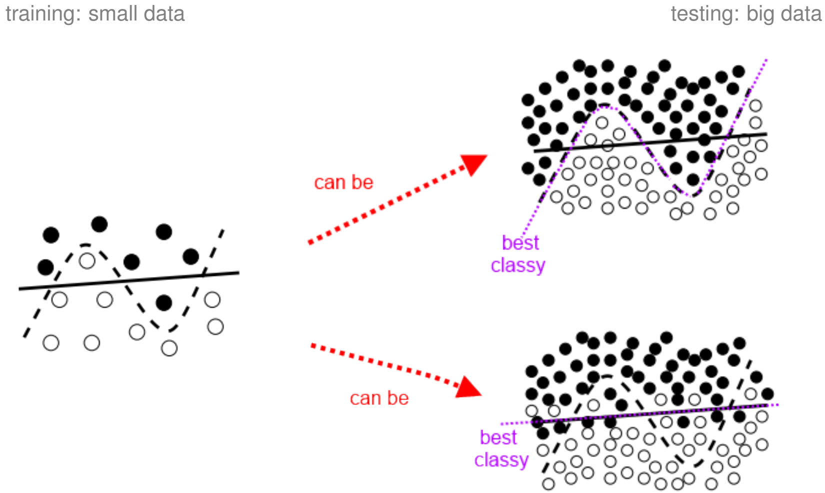
Overfitting Dilemna

Given too few data we do not know which classifier works better: a simple or complex one.

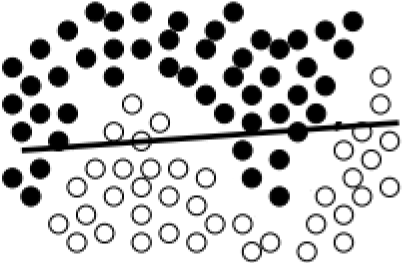


Overfitting Dilemna

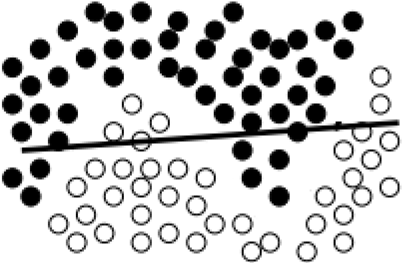
Given too few data we do not know which classifier works better: a simple or complex one.



Two things can go wrong...



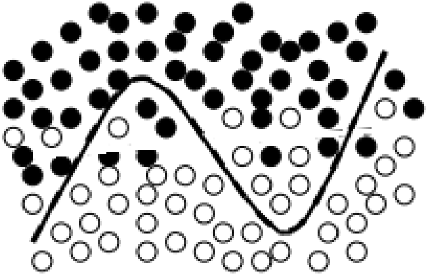
Two things can go wrong...



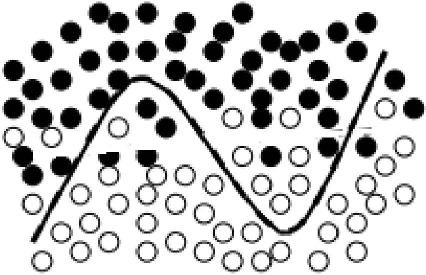
Underfitting

Learning a too simple classifier that is not complex enough to describe the training data well.

Two things can go wrong...



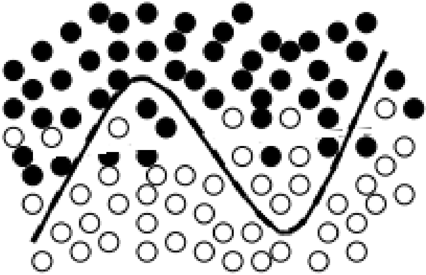
Two things can go wrong...



Overfitting

Learning a too complex classifier that fits the training data “too well”

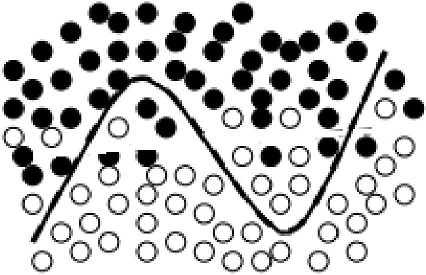
Two things can go wrong...



Overfitting

Learning a too complex classifier that fits the training data “too well” (does not “generalize” to new data)

Two things can go wrong...



Overfitting

Learning a too complex classifier that fits the training data “too well” (does not “generalize” to new data)

Generalization

Predict accurately for new (previously unseen) examples.

Example: SVM with **polynomial** kernel

Polynomial kernel of degree $m \in \mathbb{N}$

Defined as:

$$k(\mathbf{x}, \tilde{\mathbf{x}}) := (\langle \mathbf{x}, \tilde{\mathbf{x}} \rangle + b)^m, \quad b \in \mathbb{R}_+$$

Example: SVM with **polynomial** kernel

Polynomial kernel of degree $m \in \mathbb{N}$

Defined as:

$$k(\mathbf{x}, \tilde{\mathbf{x}}) := (\langle \mathbf{x}, \tilde{\mathbf{x}} \rangle + b)^m, \quad b \in \mathbb{R}_+$$

Prediction functions are degree-m polynomials:

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle = \sum_{i=(i_1,\ldots,i_d)\in\mathbb{N}_0^d : \sum_{j=1}^d i_j \leq m} w_i c_i x_1^{i_1} \cdots x_d^{i_d}$$

Example: SVM with **polynomial** kernel

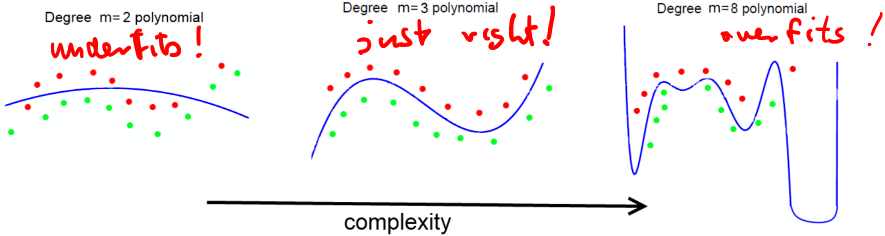
Polynomial kernel of degree $m \in \mathbb{N}$

Defined as:

$$k(\mathbf{x}, \tilde{\mathbf{x}}) := (\langle \mathbf{x}, \tilde{\mathbf{x}} \rangle + b)^m, \quad b \in \mathbb{R}_+$$

Prediction functions are degree-m polynomials:

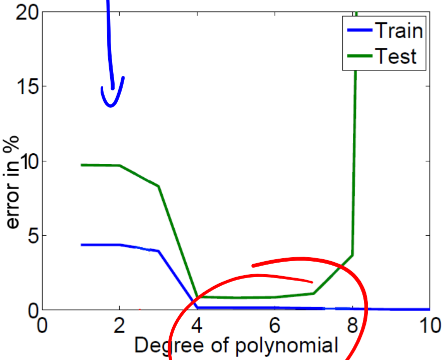
$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle = \sum_{i=(i_1,\ldots,i_d) \in \mathbb{N}_0^d : \sum_{j=1}^d i_j \leq m} w_i c_i x_1^{i_1} \cdots x_d^{i_d}$$



Typical result

under-fitting

overfitting



just right

Example: SVM with **Gaussian** kernel

Gaussian kernel with bandwidth $\sigma > 0$

$$k(\mathbf{x}, \tilde{\mathbf{x}}) := \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \tilde{\mathbf{x}}\|^2\right)$$

Demo:

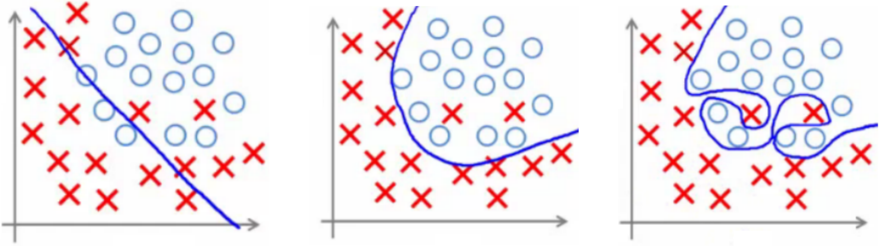
`https://cs.stanford.edu/~karpathy/svmjs/demo/`

Example: SVM with **Gaussian** kernel

Gaussian kernel with bandwidth $\sigma > 0$

$$k(\mathbf{x}, \tilde{\mathbf{x}}) := \exp\left(-\frac{1}{2\sigma^2}\left\|\mathbf{x} - \tilde{\mathbf{x}}\right\|^2\right)$$

Demo:
`https://cs.stanford.edu/~karpathy/svmjs/demo/`



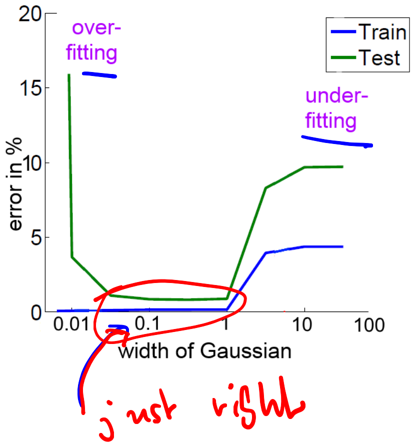
large width — under-fitting

moderate width — just right

small width — over-fitting

complexity

Typical result



Similar Behavior of Various Other Learning Machines

k-nearest neighbor algorithm:

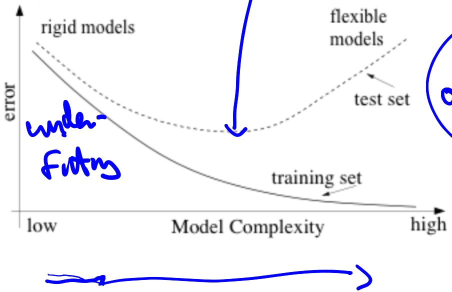
- too small *k*: overfits
- too large *k*: underfits

Deep learning

- Architecture that is too deep
  & has too many neurons and connections: overfits
- Architecture that is too shallow
  & has too few neurons and connections: underfits

Big Picture

just right



overfitting

The Problem

Even the simplest classifiers can overfit, e.g.:

The Problem

Even the simplest classifiers can overfit, e.g.:

- linear classifiers

The Problem

Even the simplest classifiers can overfit, e.g.:

- linear classifiers
- with very large number of dimensions

The Problem

Even the simplest classifiers can overfit, e.g.:

- linear classifiers
- with very large number of dimensions

Especially in very large danger of overfitting are:

- deep neural networks

(because they are very complex)

The Problem

Even the simplest classifiers can overfit, e.g.:

- linear classifiers
- with very large number of dimensions

Especially in very large danger of overfitting are:

- deep neural networks

(because they are very complex)

How to avoid under- and overfitting?

The Problem

Even the simplest classifiers can overfit, e.g.:

- linear classifiers
- with very large number of dimensions

Especially in very large danger of overfitting are:

- deep neural networks

(because they are very complex)

How to avoid under- and overfitting?

Solution

Choose an (overly) complex model (thus avoiding **under**fitting), but use a technique called **regularization** to avoid **over**fitting