Deep Learning for Image Analysis - Lecture 1: Introduction to Machine Learning

Thomas Walter, PhD

Centre for Computational Biology (CBIO)
MINES Paris-Tech, PSL Research University
Institut Curie, PSL Research University
INSERM U900

Overview

Overview

Machine Learning: Basic definitions

- **Machine Learning** is concerned with the technology that enables computer programs to improve their performance at a certain task by experience.
- In particular, we want to infer (learn) some function f from data, capable of predicting the output y from an input (measurement) x:

$$y = f(x)$$

- In **supervised learning**, the training data contains both measurements x_i and the corresponding output variables y_i. Together, they build the training set T:

$$T = \{(x_i, y_i)\}_{i=1,\ldots,N}$$

- In **unsupervised learning**, there are no annotations y_i. We aim at inferring **patterns** from the data (clusters, latent variables).

Different types of learning

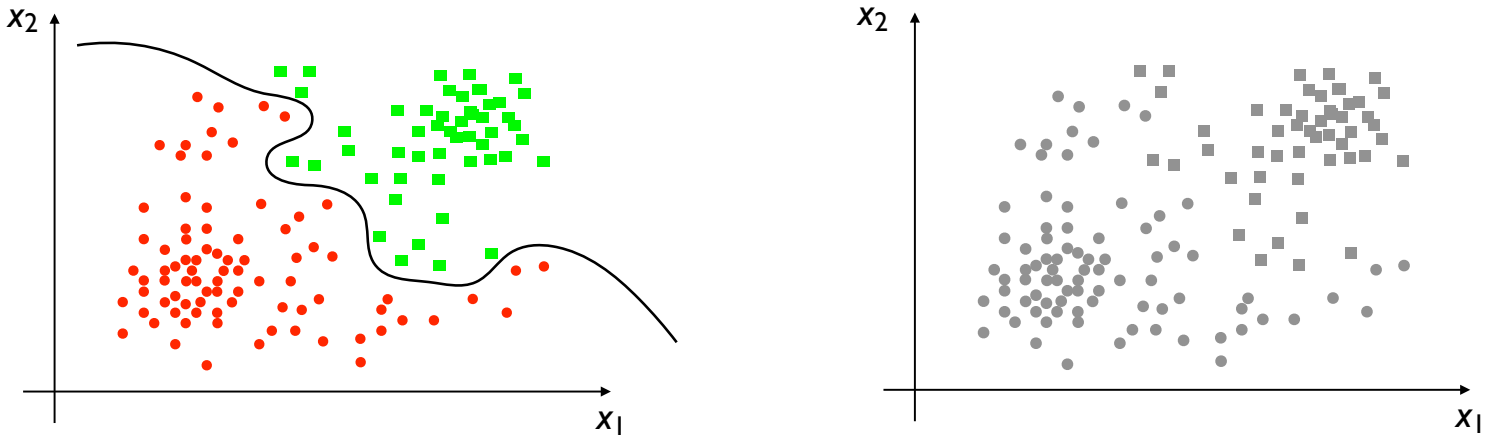


Figure: Supervised and unsupervised learning

Depending on the type of the output variable and on whether we are in a supervised or unsupervised setting, we have different names for machine learning problems:

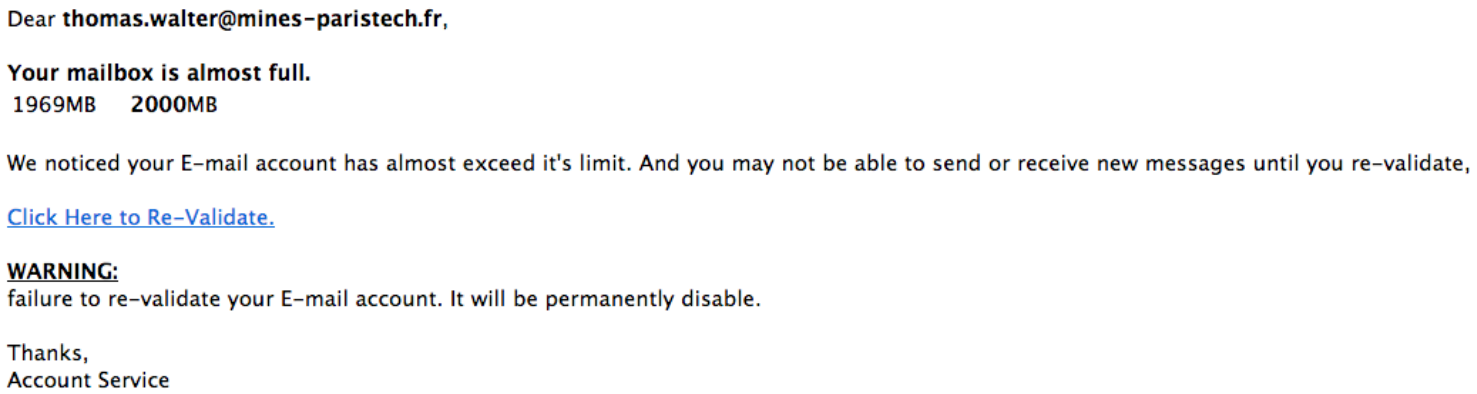
	Supervised	Unsupervised
y discrete	Classification	Clustering
y continuous	Regression	Dimensionality reduction

Objects and features

- Machine learning typically deals with objects outside the mathematical world (emails, images, genomes, cars, ...).
- The first step is therefore to find a suitable representation of the objects.
  - **feature engineering**: finding descriptors according to existing domain knowledge
  - **representation learning**: learning the descriptors together with the classifier
- In many cases the objects can be represented by a P-dimensional vector of features (or descriptors): $\mathrm{x} \in \mathbb{R}^P$.
- It can be convenient to map a feature vector to a higher dimensional space:

$$\begin{aligned} \phi : \mathbb{R}^P &\rightarrow \mathbb{R}^Q && (1) \\ \mathrm{x} &\rightarrow \phi(\mathrm{x}) && (2) \end{aligned}$$

Example: classification of SPAM emails



Dear **thomas.walter@mines-paristech.fr**,

Your mailbox is almost full.
1969MB 2000MB

We noticed your E-mail account has almost exceed it's limit. And you may not be able to send or receive new messages until you re-validate,

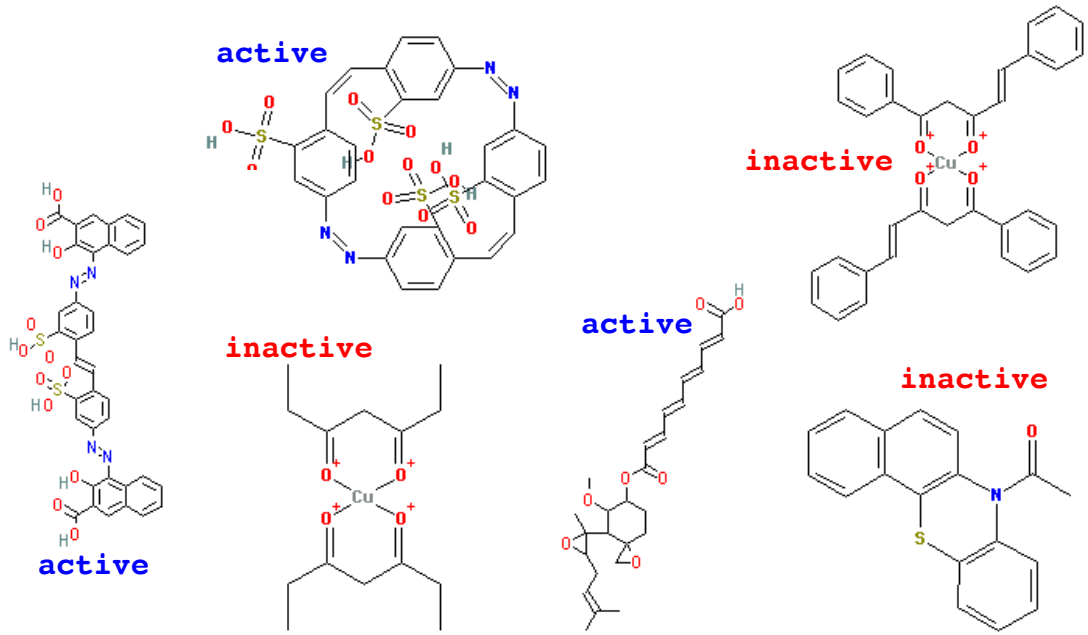
Click Here to Re-Validate.

WARNING:
failure to re-validate your E-mail account. It will be permanently disable.

Thanks,
Account Service

- This is a binary classification problem: $y \in \{0, 1\}$ (0: junk, 1: normal).
- The features can be constructed in the following way: for each email annotated by the user, the words are listed. An email is described as a vector of frequencies of these words.
- The system learns then a function that assigns to each vector of measured word frequencies the label y.

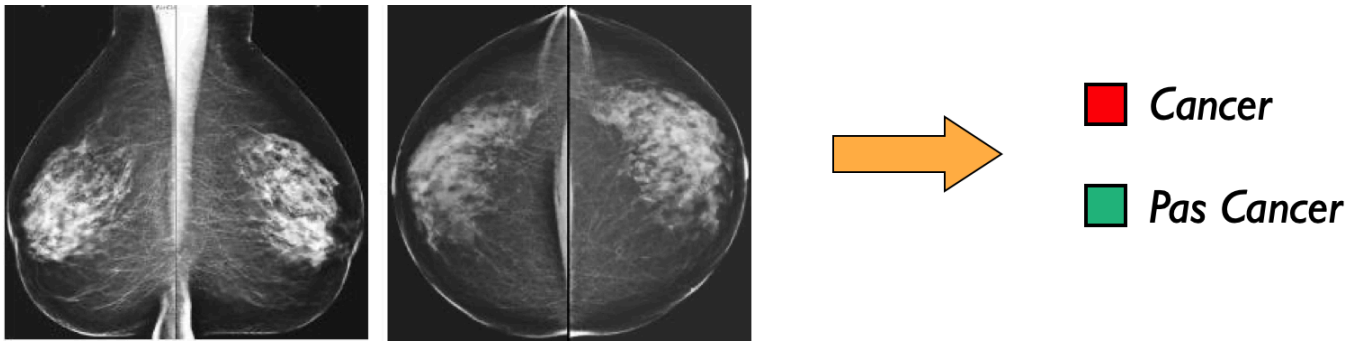
Example: classification of drugs



- Here, we want to classify molecules with respect to their efficiency against a disease (binary classification: a drug is efficient or not).
- An important question here is how to encode a molecule. One option is to define chemoinformatic features and obtain a vectorial representation of the molecule $x \in \mathbb{R}^P$.

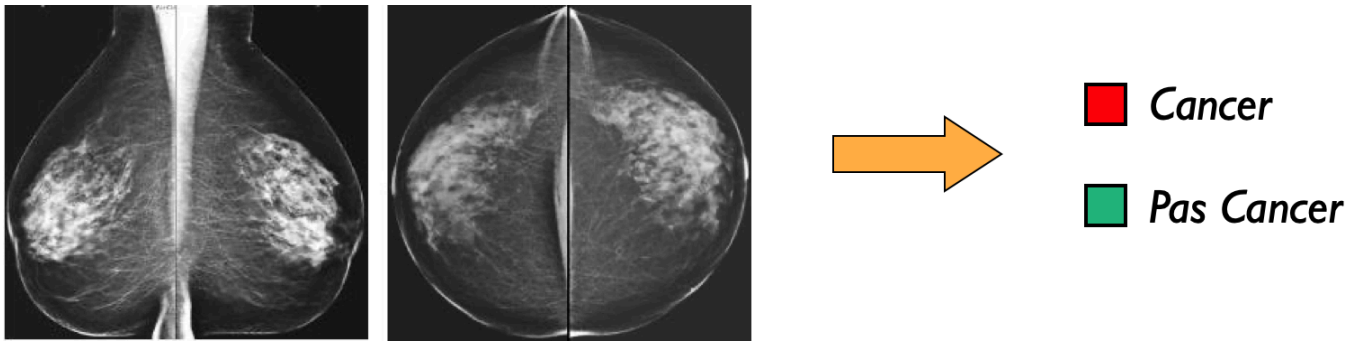
Overview

An example from medical diagnostics: radiology



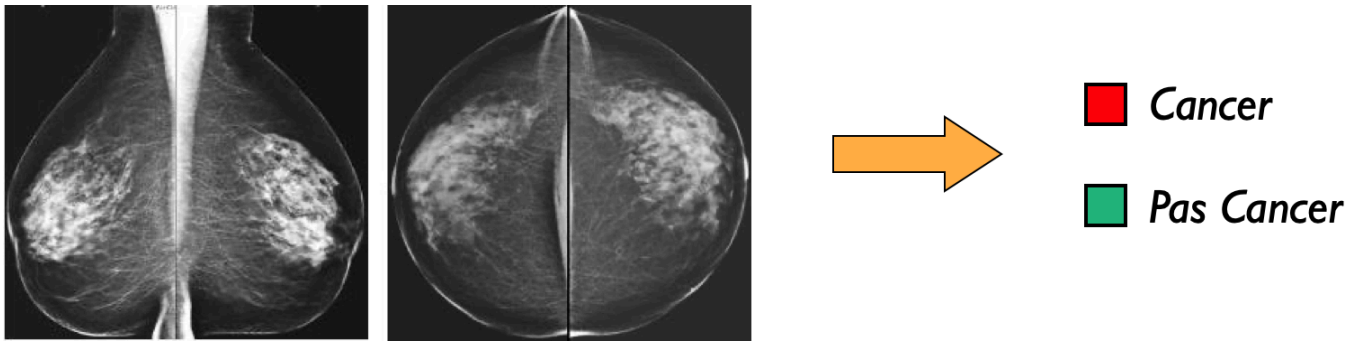
- Problem presented in an international challenge (won by Terapixel).

An example from medical diagnostics: radiology



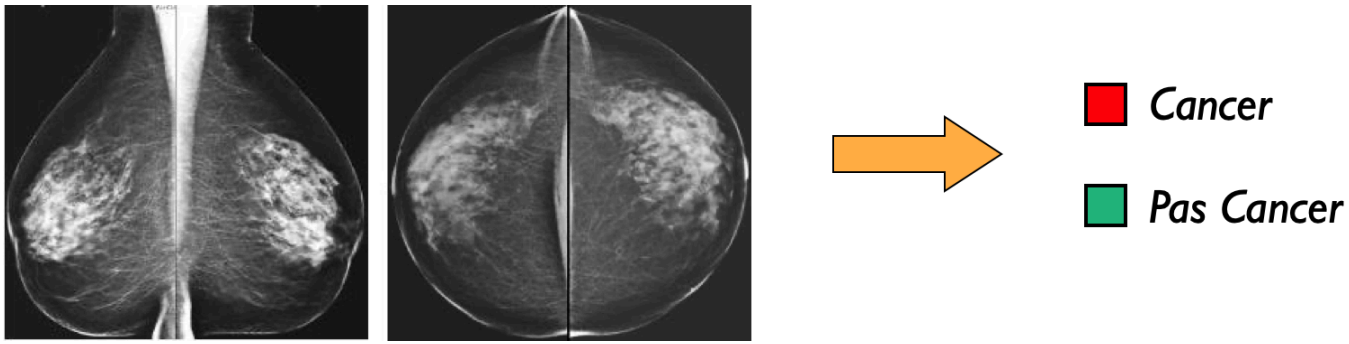
- Problem presented in an international challenge (won by Terapixel).
- A Neural Networks trained on 640.000 images obtained an accuracy of 80%.

An example from medical diagnostics: radiology



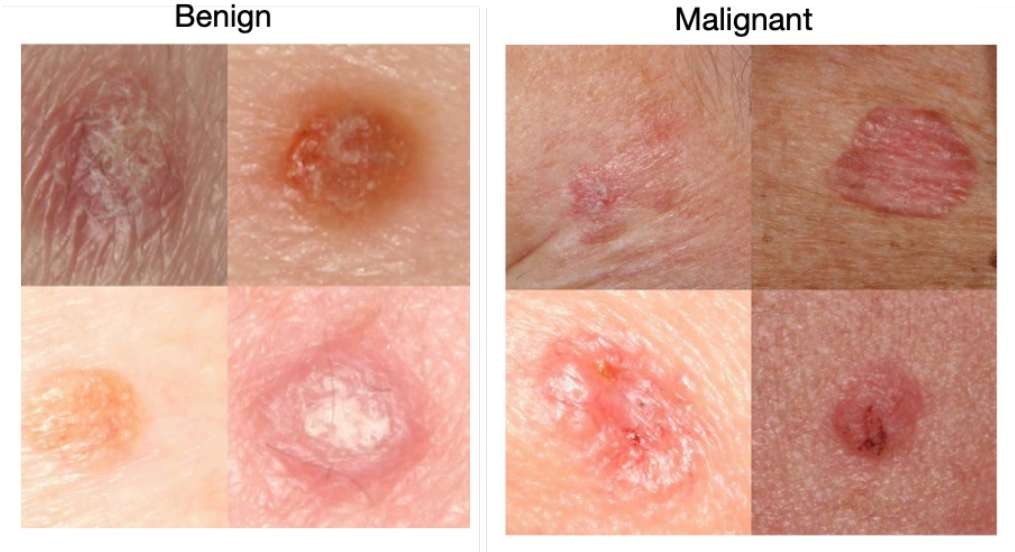
- Problem presented in an international challenge (won by Terapixel).
- A Neural Networks trained on 640.000 images obtained an accuracy of 80%.
- Problem ownership: who owns the trained network?

An example from medical diagnostics: radiology



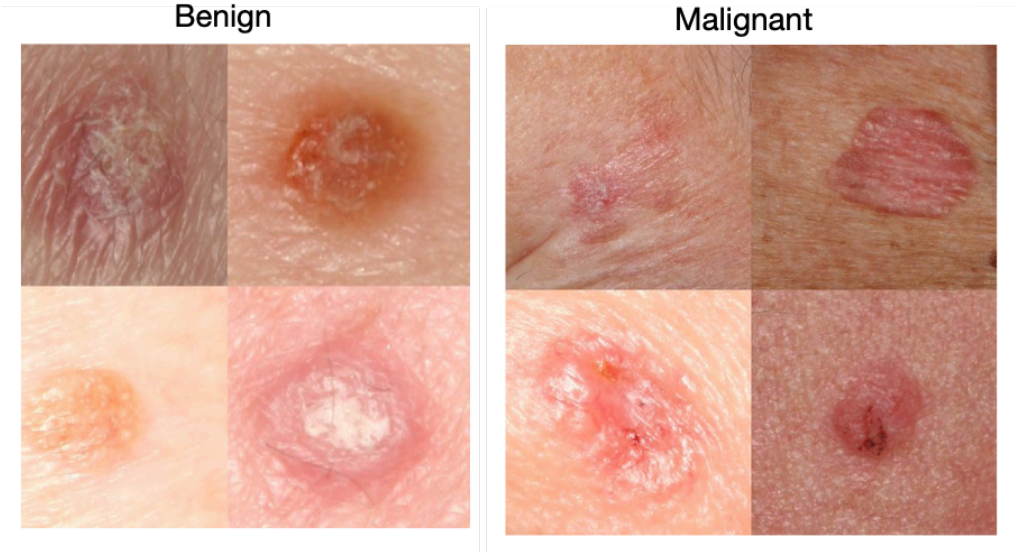
- Problem presented in an international challenge (won by Terapixel).
- A Neural Networks trained on 640.000 images obtained an accuracy of 80%.
- Problem ownership: who owns the trained network?
- Problem responsibility: who is responsible in case of a wrong diagnosis?

Skin cancer detection



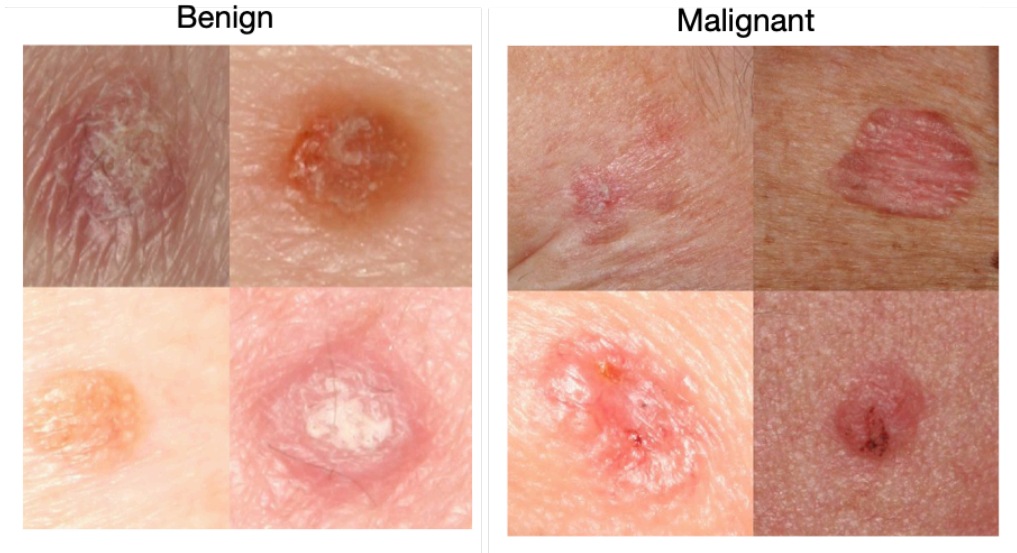
- A Neural Network trained on 127 000 images was shown to outperform 21 dermatologists (72% accuracy vs. 66%) [Esteva et al., 2017]

Skin cancer detection



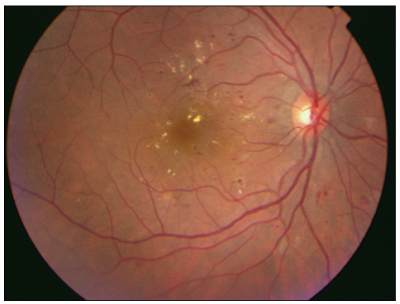
- A Neural Network trained on 127 000 images was shown to outperform 21 dermatologists (72% accuracy vs. 66%) [Esteva et al., 2017]
- Improving healthcare or mass layoff of medical doctors?

Skin cancer detection

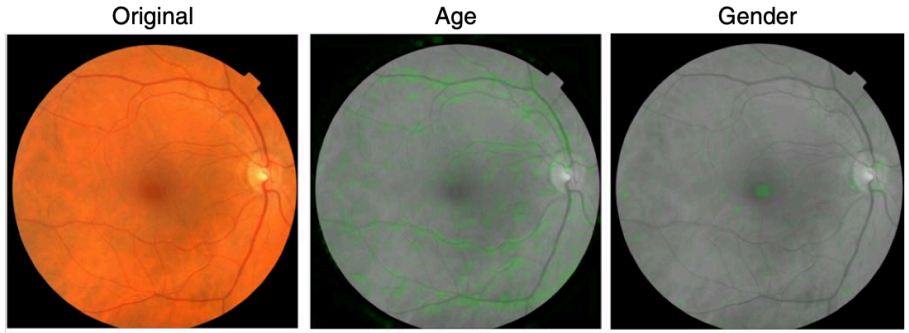


- A Neural Network trained on 127 000 images was shown to outperform 21 dermatologists (72% accuracy vs. 66%) [Esteva et al., 2017]
- Improving healthcare or mass layoff of medical doctors?
- Do the algorithms work equally well for different ethnic groups?

What else can we predict?

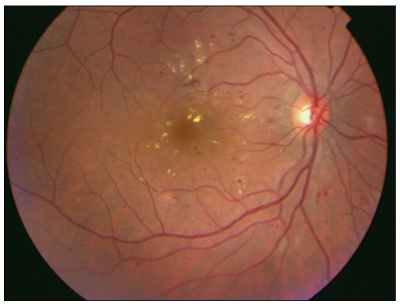


Diabetic Retinopathy

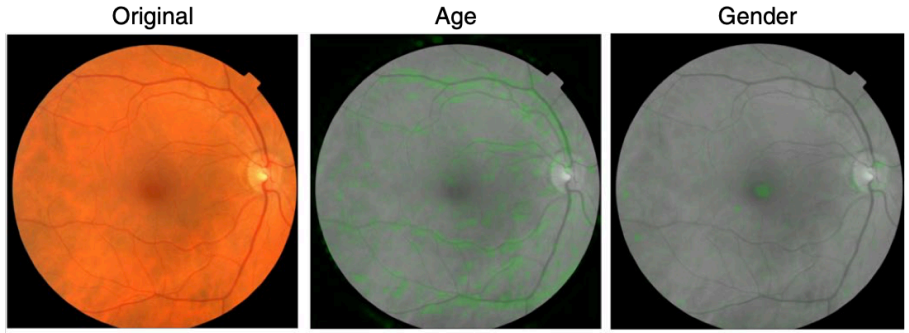


- A NN trained on 128.000 images reaches 87% sensitivity and 91 % specificity and became the first FDA approved autonomous IA medical device [Abràmoff et al., 2018]

What else can we predict?

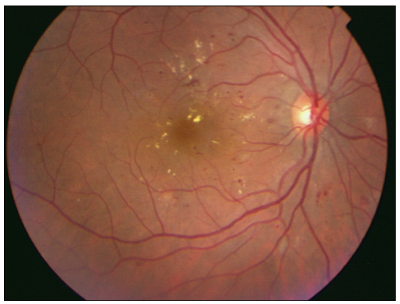


Diabetic Retinopathy

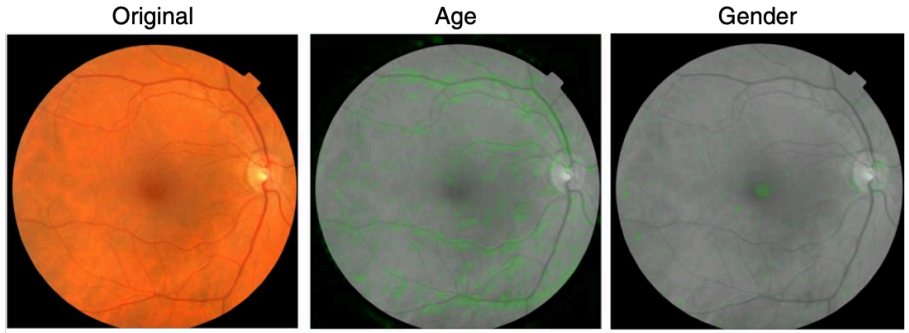


- A NN trained on 128.000 images reaches 87% sensitivity and 91 % specificity and became the first FDA approved autonomous IA medical device [Abràmoff et al., 2018]
- From eye images, we can also predict age, BMI, cardiovascular risks and smoking habits [Poplin et al., 2018]

What else can we predict?

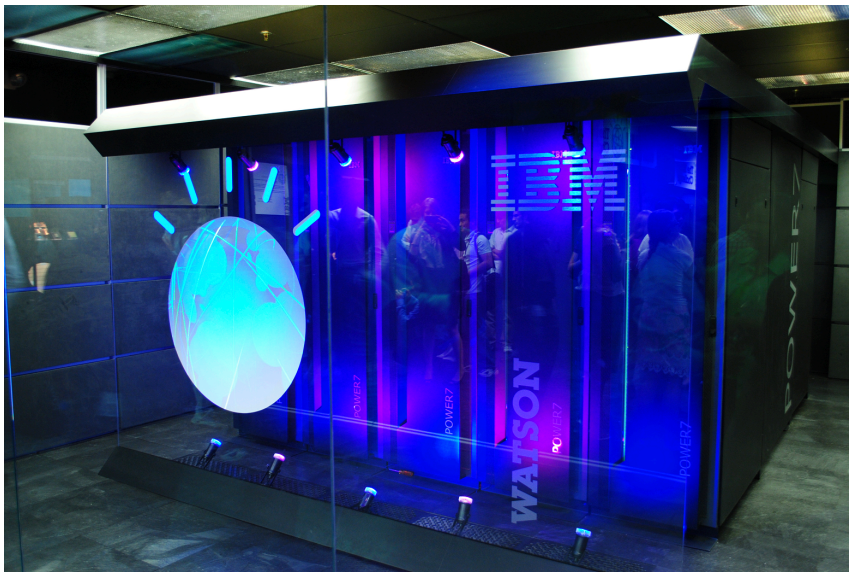


Diabetic Retinopathy



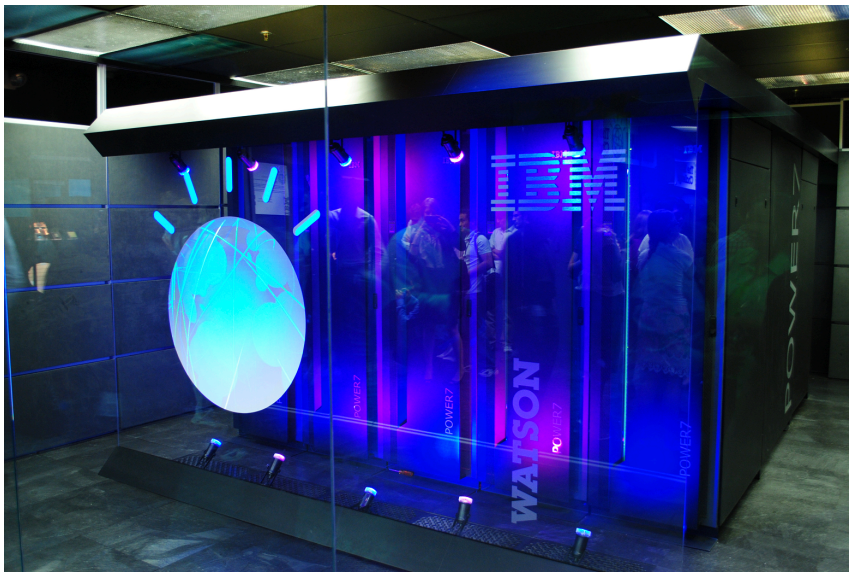
- A NN trained on 128.000 images reaches 87% sensitivity and 91 % specificity and became the first FDA approved autonomous IA medical device [Abràmoff et al., 2018]
- From eye images, we can also predict age, BMI, cardiovascular risks and smoking habits [Poplin et al., 2018]
- What do we allow to predict? Who is using this information?

Watson: should we stop thinking on our own?



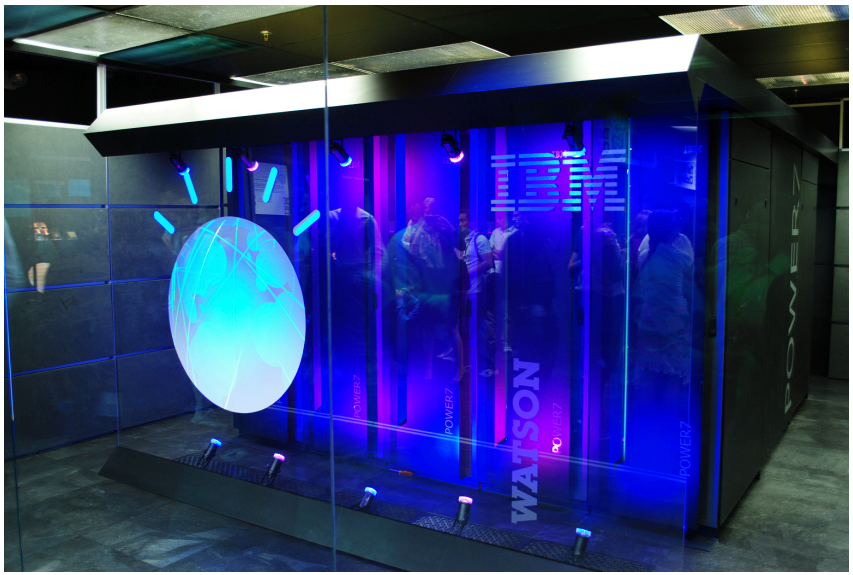
- Watson (developed by IBM) outperformed humans in the game Jeopardy.

Watson: should we stop thinking on our own?



- Watson (developed by IBM) outperformed humans in the game Jeopardy.
- The system is used for clinical decision support (with usually good results).

Watson: should we stop thinking on our own?



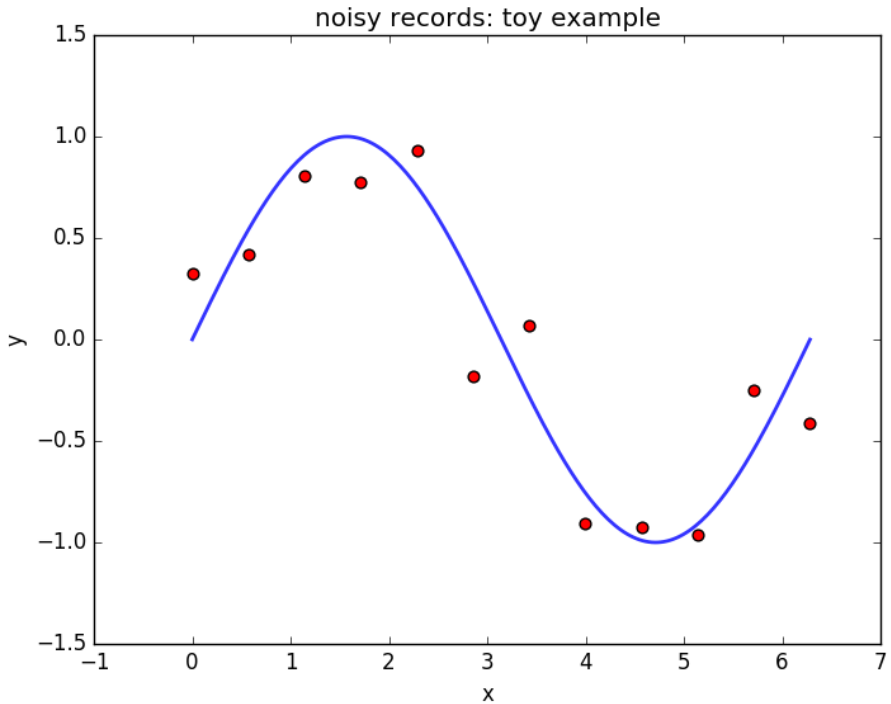
- Watson (developed by IBM) outperformed humans in the game Jeopardy.
- The system is used for clinical decision support (with usually good results).
- Medical personnel usually follow the advice.

Beyond the medical domain ...

- AI for criminal risk assessment: predicting the likelihood from the prisoners profiles. Problem: learning "racist" associations.
- AI in autonomous driving:
  - Individual cases versus overall performance
  - Responsibility in case of an accident
  - Moral hard-coding
- Generative Models for images: new tools for manipulations, mobbing, etc.
- AI replacing humans: mass layouts?
- ...

Overview

A simple example: polynomial curve fitting[1]



- From a set of measured points (x_i, y_i) (red), we would like to build a model to predict the value y for any given x.
- The true function is $g(x) = \sin(x)$ (displayed in blue).
- The measurements y_i are noisy outputs of that function, i.e.

$$y_i = \sin(x_i) + \epsilon \,, \qquad \epsilon \sim \mathcal{N}(0, 0.2) \tag{3}$$

[1] Example adapted from [Bishop, 2006]

A simple example: polynomial curve fitting

- We use the following polynomial model:

$$\begin{aligned} f(x) &= a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m \\ &= \boldsymbol{\theta}^T \boldsymbol{\phi}(x) \end{aligned} \tag{4}$$

- Parameter vector: $\boldsymbol{\theta} = (a_0, a_1, \ldots, a_m)^T$
- Here, the initial measurement x is a scalar. In our model, we map x to a higher dimensional space:

$$\begin{aligned} \boldsymbol{\phi} : \mathbb{R}^P &\rightarrow \mathbb{R}^Q \\ x &\rightarrow \boldsymbol{\phi}(x) = (1, x, x^2, \ldots, x^m)^T \end{aligned} \tag{5}$$

- The model is linear in the parameters θ and linear in $\boldsymbol{\phi}$, but for $m > 1$, the model is not linear in x.

A simple example: polynomial curve fitting

- One classical approach is to minimize the least squared error between measured and predicted values:

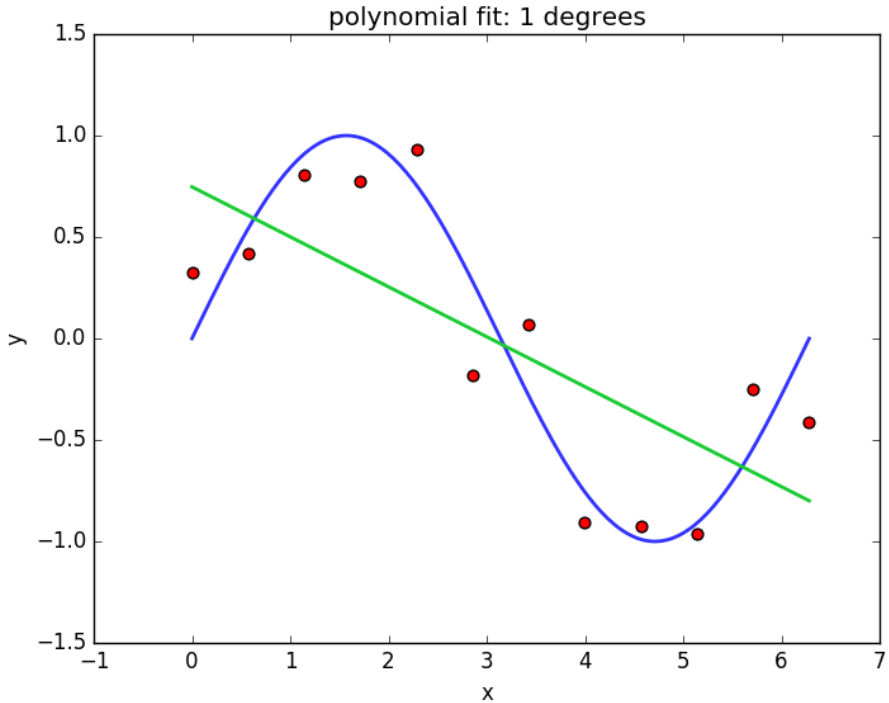
$$\begin{aligned} \min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) &= \min_{\boldsymbol{\theta}} \sum_{i=1}^{N} (y_i - f(x_i))^2 \\ &= \min_{\boldsymbol{\theta}} \sum_{i=1}^{N} (y_i - \boldsymbol{\theta}^T \boldsymbol{\phi}(x_i))^2 \end{aligned} \tag{6}$$

- This can be achieved by setting the gradient with respect to $\boldsymbol{\theta}$ to zero:

$$\nabla_{\boldsymbol{\theta}} L = \left(\frac{\partial L}{\partial a_0}, \frac{\partial L}{\partial a_1}, \ldots, \frac{\partial L}{\partial a_m}\right)^T = 0 \tag{7}$$

- Unlike for most optimization problems in this course, this leads to an analytical solution for $\boldsymbol{\theta}$. This is known as **linear regression**. For more details, we refer to [Hastie et al., 2009].

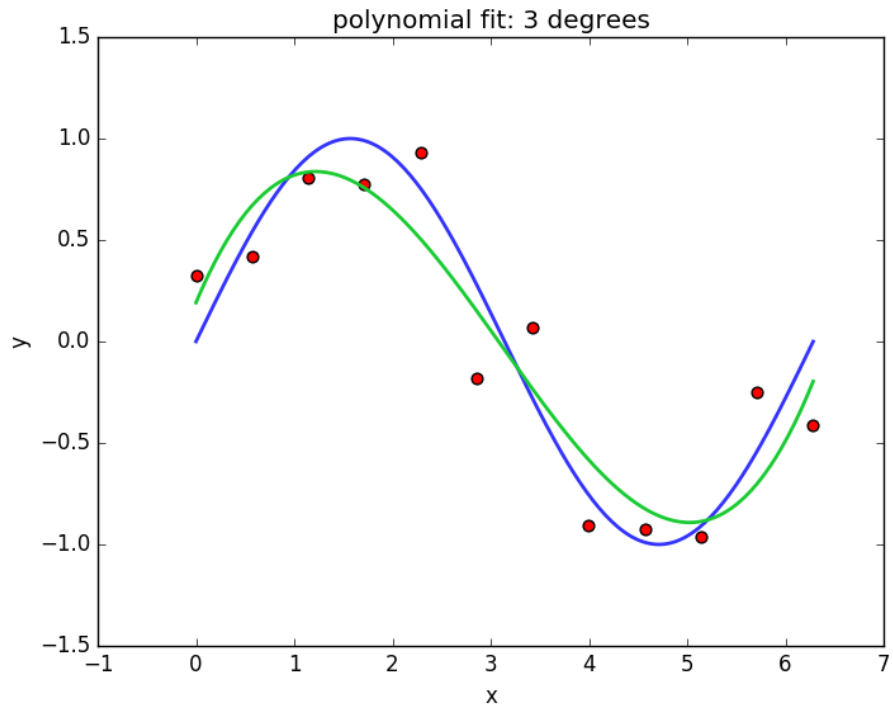
Overfitting and underfitting



$$\|\boldsymbol{\theta}\|^2 = 0.67$$

For $m = 1$, the model is linear in its inputs. The solution is not capable of modeling the measured data points; we get a poor approximation of the original function. The family of functions we have used was not complex enough to model the true data distribution. We also speak of **underfitting**.

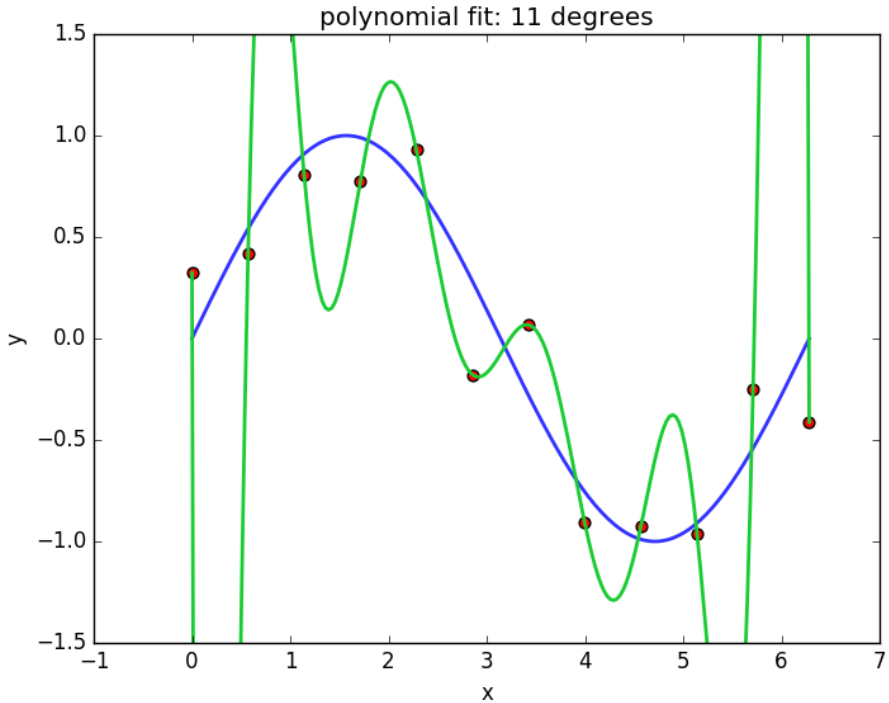
Overfitting and underfitting



$$\|\boldsymbol{\theta}\|^2 = 1.72$$

For $m = 3$, we obtain a solution that seems to be quite right: it is sufficiently complex to model the true data distribution, but not too complex to model the small variations which are due to noise.

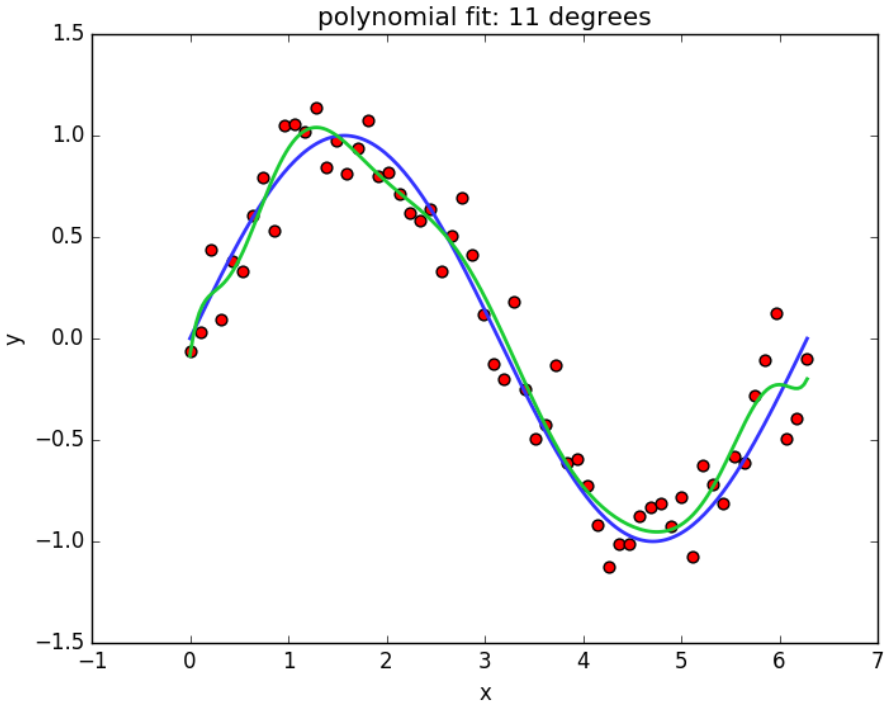
Overfitting and underfitting



$$\|\boldsymbol{\theta}\|^2 \approx 10^7$$

For $m = 11$, we obtain a solution that has zero error (the function passes through every point of the training set). But the coefficients with large absolute values that cancel each other precisely on the training points lead to a highly unstable function. We speak of **overfitting** and **poor generalization**.

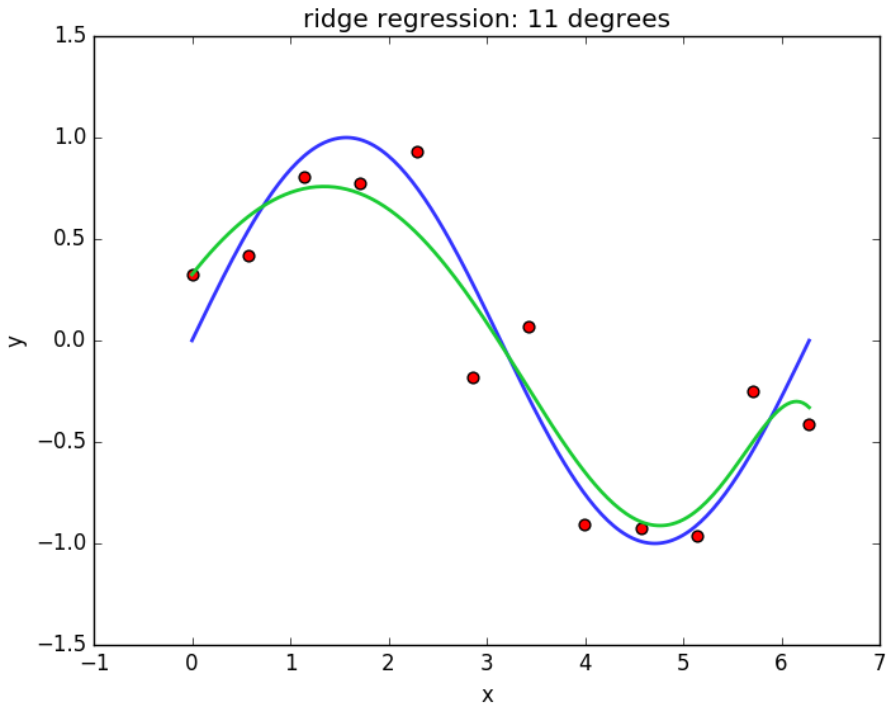
Overfitting and underfitting



$$\|\boldsymbol{\theta}\|^2 = 5647$$

One way of reducing overfitting is to increase the number of samples. Even if the function is complex, it cannot be "too wild", as it has to find a compromise between many training samples. This however implies the annotation (or measurement) of more samples.

Overfitting and underfitting

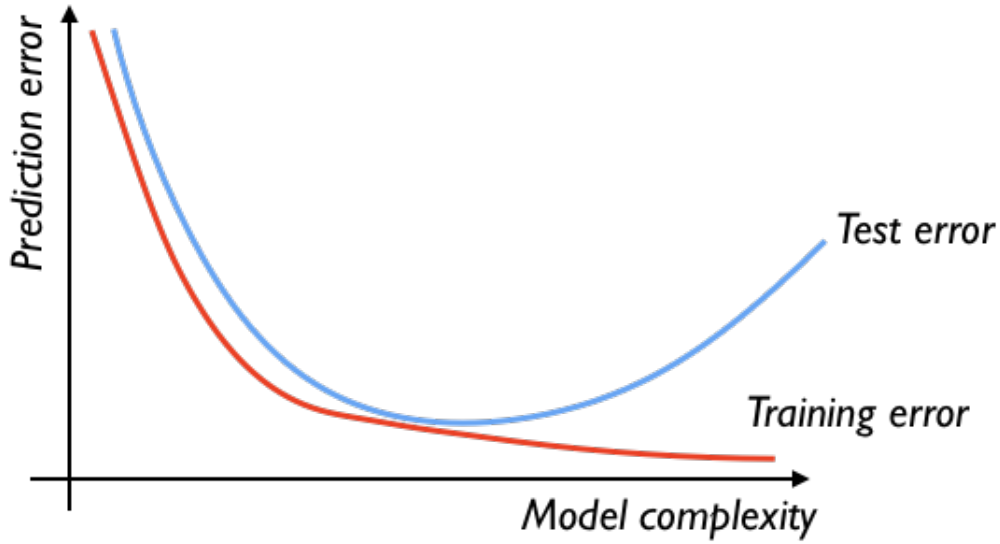


$$\|\boldsymbol{\theta}\|^2 = 0.41$$

Another way of preventing overfitting without increasing the number of samples, is to add a penalization term in the optimization procedure. This is also known as **regularization**:

$$L = \sum_{i=1}^{N} (y_i - \boldsymbol{\theta}^T \boldsymbol{\phi}(x_i))^2 + \lambda \|\boldsymbol{\theta}\|^2 \tag{8}$$

Generalization: training and test error



- Supervised Learning aims at finding a function f that predicts an output value y from a measurement x for unseen data, i.e. for data that has not been used to find f.
- Machine Learning is much concerned with avoiding f to **memorize** the training set, i.e. to perform well on a training set but poorly a test set.
- An important paradigm is that we must never evaluate the performance of our machine learning method on the data that has been used to train it.

Generalization: strategies

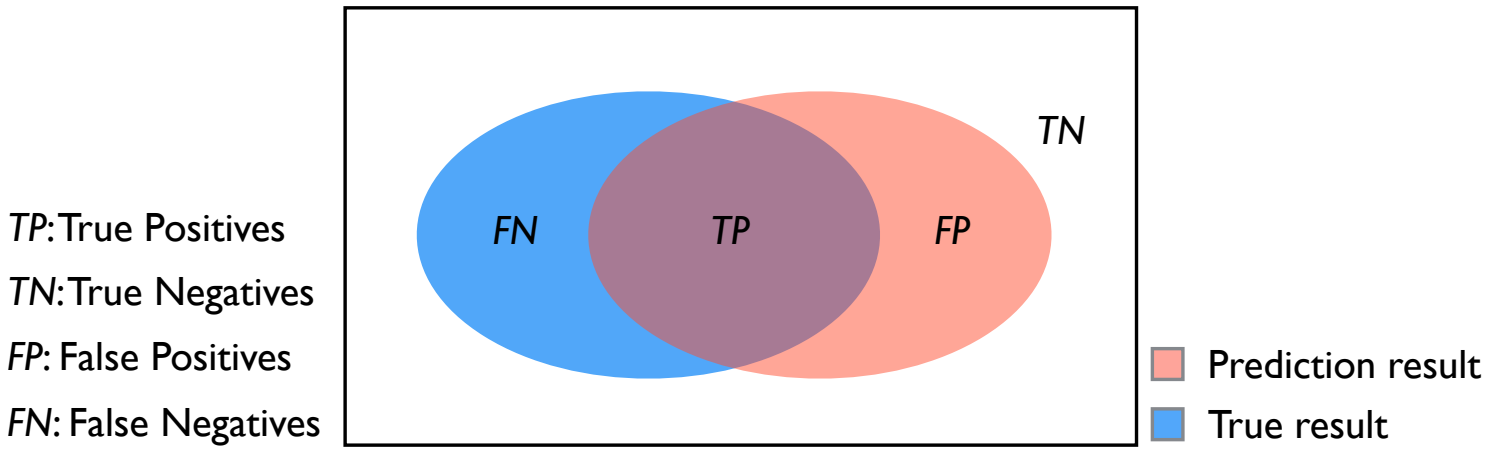
- Many ML algorithms can be written as an optimization problem:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) + \lambda \mathcal{R}(\boldsymbol{\theta})$$

  Minimizing the loss $L(\boldsymbol{\theta})$ aims at finding the rule to reproduce the annotations in the training set, minimizing the regularization term $\mathcal{R}(\boldsymbol{\theta})$ aims at avoiding the model to adapt too much to the training data, leading to simpler models. We have seen the L_2 norm, but there are many other options for $\mathcal{R}$.
- Other regularization strategies include:
  - Model averaging (ensemble methods)
  - Artificial or actual increase of training data

Overview

Model evaluation - Accuracy

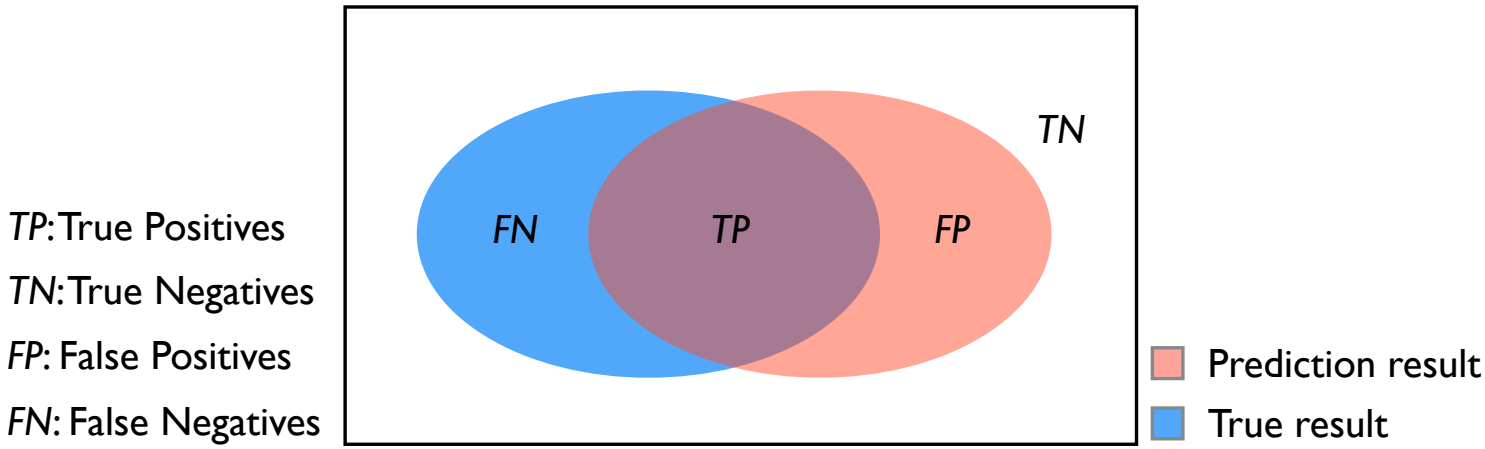


- Accuracy: the percentage of correctly classified samples.

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

- The accuracy is the most widely used metric, but it does not tell us which kind of errors are made.

Model evaluation - Sensitivity / Recall

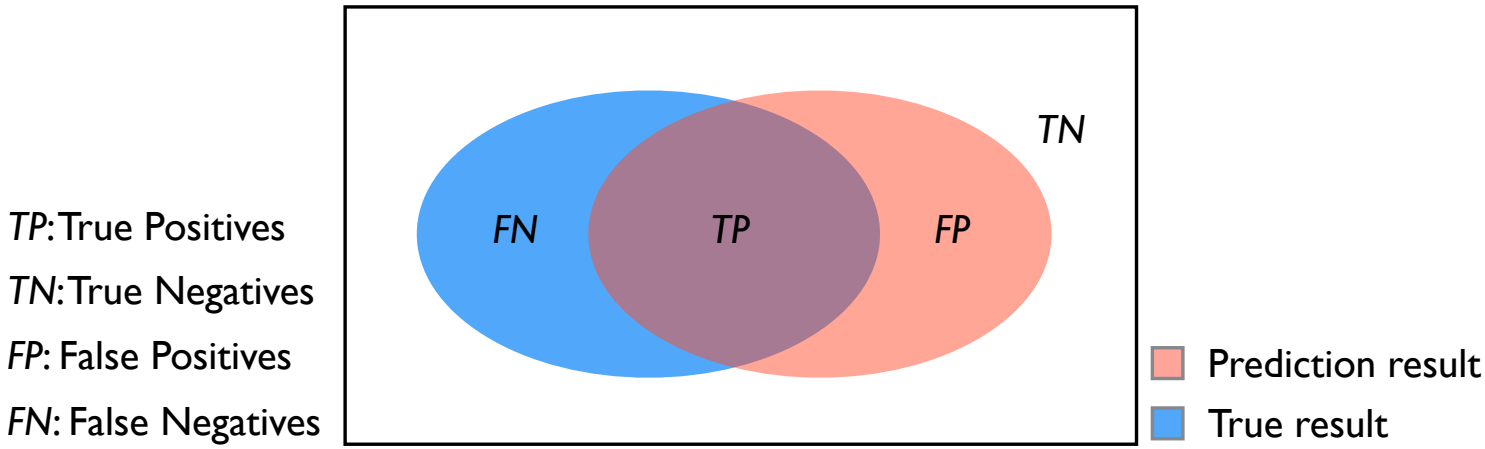


- Recall (aka sensitivity) is the percentage of positive samples that have been correctly classified as positive.

$$\text{recall} = \text{sensitivity} = \frac{TP}{TP + FN} \tag{10}$$

- Example: percentage of sick patients that are recognized as such by the diagnostic test.
- Sensitivity alone is useless, as if all all samples are classified positive, it will be 100%.

Model evaluation - Specificity

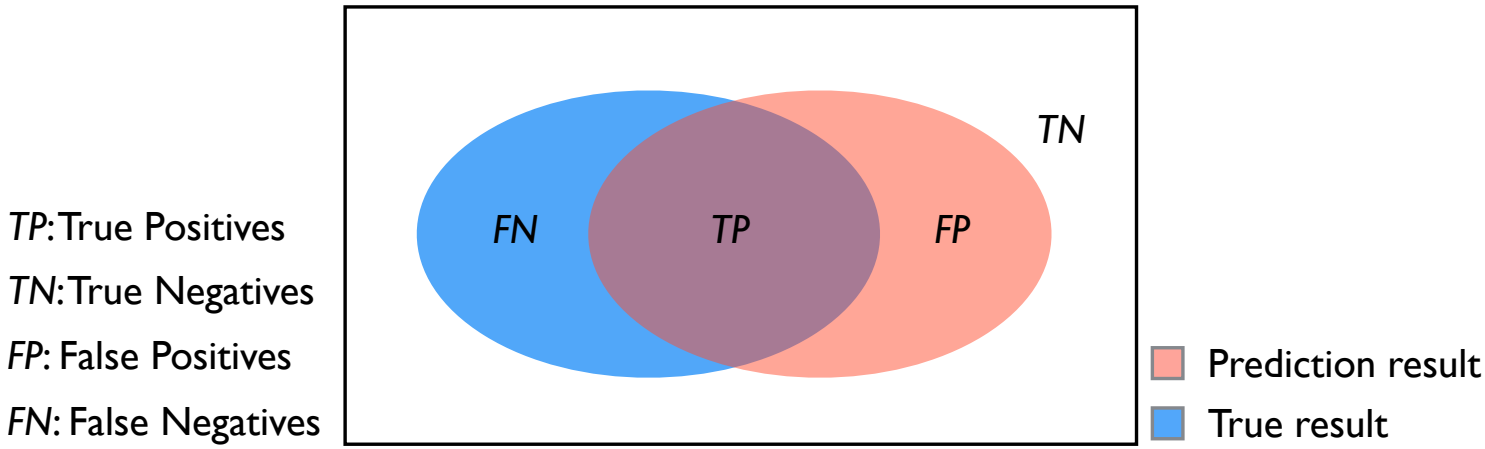


- Specificity is the percentage of negative samples that have been correctly classified as negatives

$$\text{specificity} = \frac{TN}{TN + FP} \tag{11}$$

- We usually seek a compromise between sensitivity and specificity.
- Specificity is not very informative if the dataset is very imbalanced.

Model evaluation - Precision / positive predictive value

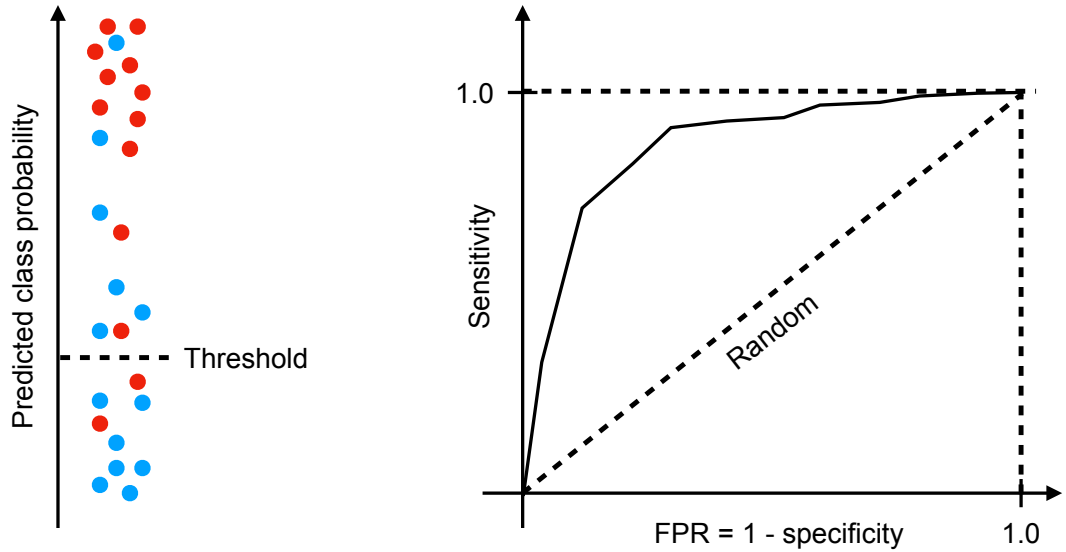


- Precision (aka positive predictive value) is the probability that a sample is positive if it was classified positive.

$$\text{precision} = \frac{TP}{TP + FP} \tag{12}$$

- Example: precision gives you the probability of being sick, given that the diagnostic test was positive.
- Precision is often preferred to specificity as it is also a suitable metric if the dataset is imbalanced.

Model evaluation - ROC and F1



- To measure a compromise between precision and recall, one can also use the F_1-score:

$$F_1 = 2\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{13}$$

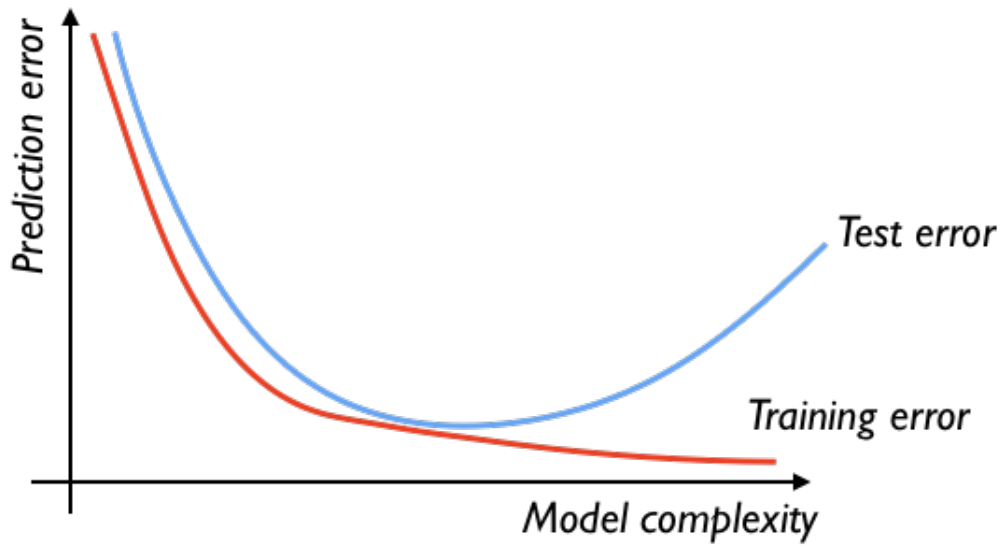
- Another popular metric is related to *ROC*-curves or precision-recall curves (AUC: area under the curve).
- The AUC is identical to the concordance index: the probability that $(+, -)$ pairs of samples are correctly ordered.

Performance Assessment for a trained classifier

- First idea: we train a classifier on the training set T and calculate the accuracy on the same training set T (empirical risk, training error).

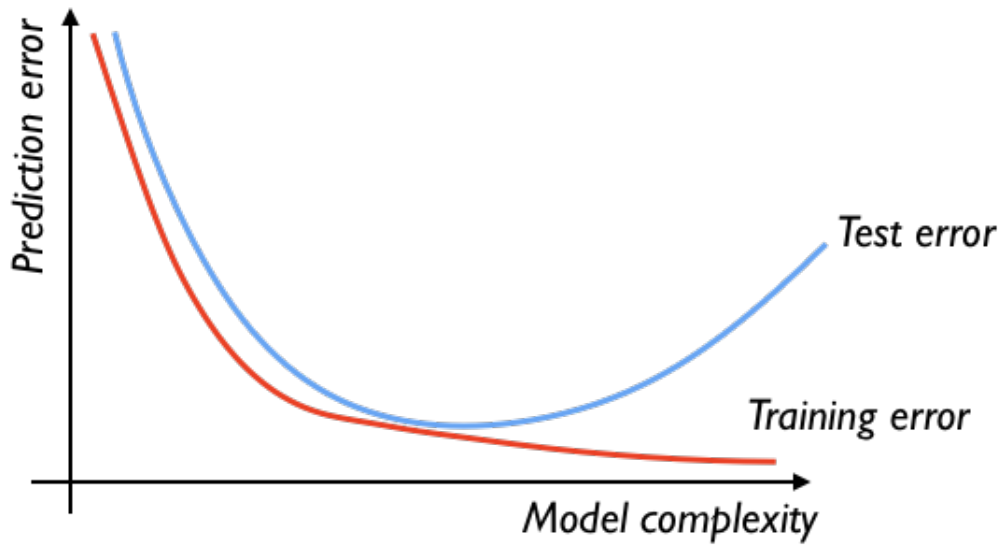
Performance Assessment for a trained classifier

- First idea: we train a classifier on the training set T and calculate the accuracy on the same training set T (empirical risk, training error).
- Problem: Training error is a poor approximation of the test error: a classifier might have good performance on the training set, but poor performance on the test set.



Performance Assessment for a trained classifier

- First idea: we train a classifier on the training set T and calculate the accuracy on the same training set T (empirical risk, training error).
- Problem: Training error is a poor approximation of the test error: a classifier might have good performance on the training set, but poor performance on the test set.



- We need to estimate the test error (error on unseen samples)!

Performance Assessment for a trained classifier

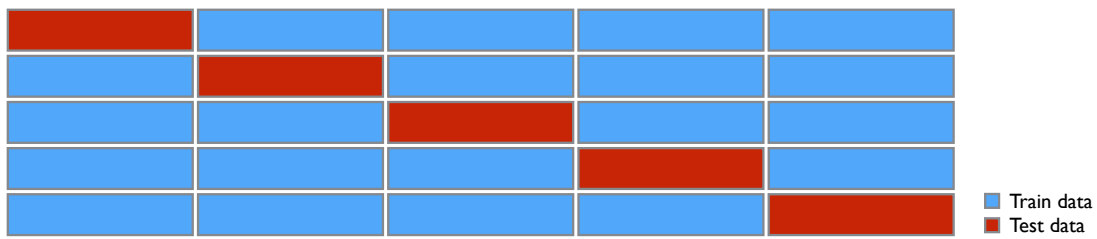
- Second idea: to split the training set in two subsets T_1, T_2. We train on T_1, we test on T_2.

Performance Assessment for a trained classifier

- Second idea: to split the training set in two subsets T_1, T_2. We train on T_1, we test on T_2.
- Problem: annotated data is often expensive, and we would like to have the largest possible sets for training and testing.

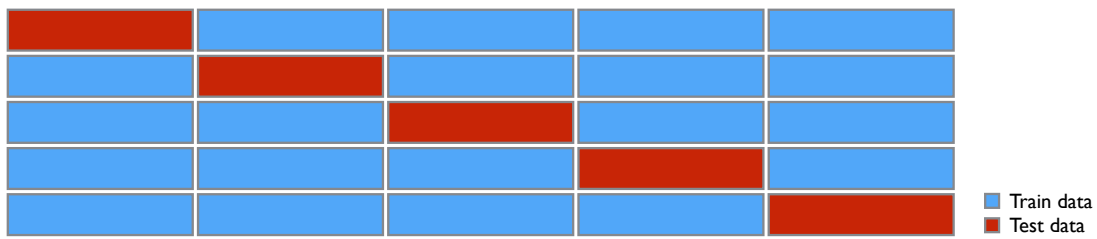
Performance Assessment for a trained classifier

- Second idea: to split the training set in two subsets T_1, T_2. We train on T_1, we test on T_2.
- Problem: annotated data is often expensive, and we would like to have the largest possible sets for training and testing.
- Third idea: Cross Validation. We split the data into K folds. We train on $K-1$ folds, and we test on the remaining fold. We iterate until each fold has been used once for testing.



Performance Assessment for a trained classifier

- Second idea: to split the training set in two subsets T_1, T_2. We train on T_1, we test on T_2.
- Problem: annotated data is often expensive, and we would like to have the largest possible sets for training and testing.
- Third idea: Cross Validation. We split the data into K folds. We train on $K-1$ folds, and we test on the remaining fold. We iterate until each fold has been used once for testing.



- This provides us with an estimation of the accuracy for unseen data (test error).

How to set hyperparameters ...

- Training a classifier: finding automatically a large number of feature weights or other parameters from annotated data (in our example: coefficients of the polynomial).
- Hyperparameters: a small number of parameters that are set to control the training process, such as the regularization parameter λ.
- The question is: how can we find good hyperparameters?
- Strategy: grid search. We choose the set of hyperparameters that perform best (i.e. that lead to the classifier with the best performance).

Performance evaluation with optimized hyperparameters

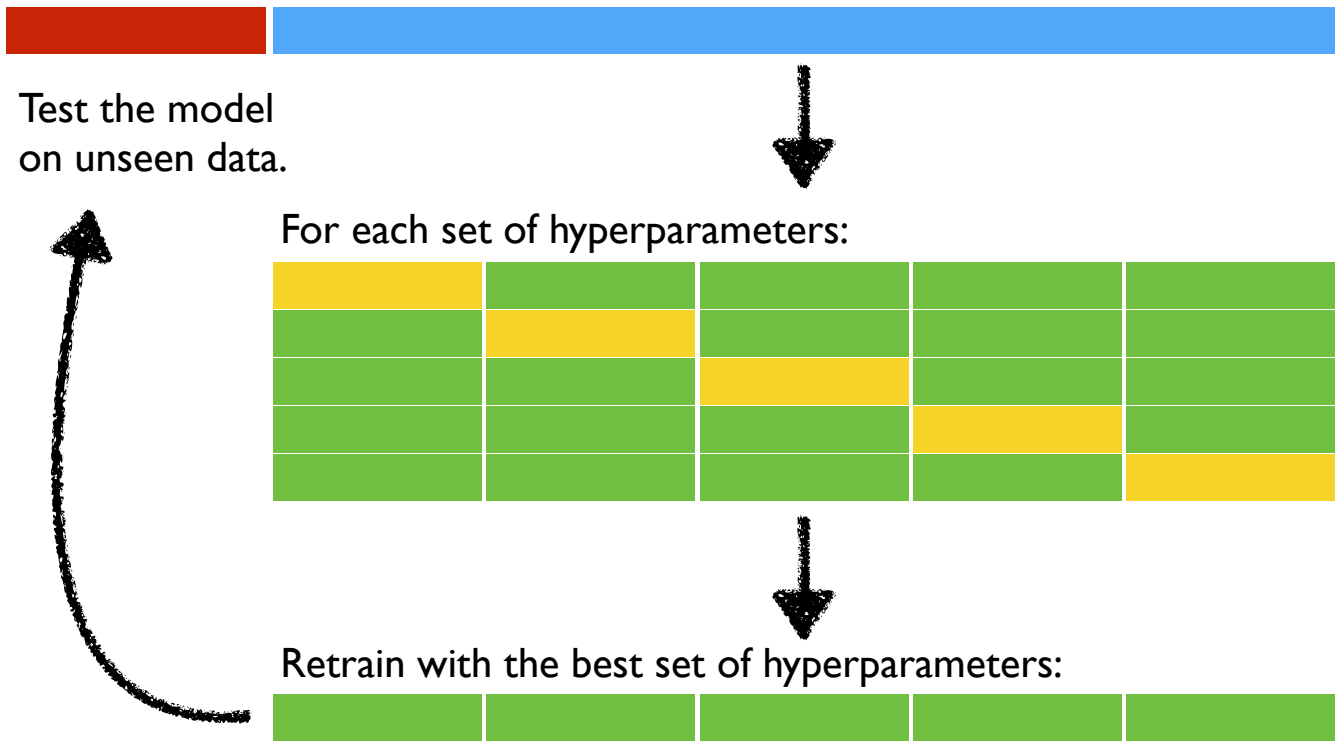
- If we optimize the hyperparameters in this way, the final performance might be over-optimistic (we have chosen the hyperparameters that give best performance for the test set, i.e. we have used the test set to set the parameters).

Performance evaluation with optimized hyperparameters

- If we optimize the hyperparameters in this way, the final performance might be over-optimistic (we have chosen the hyperparameters that give best performance for the test set, i.e. we have used the test set to set the parameters).
- Solution: We split the training set in 3:
  - **Training set**: used to obtain the classifier for a given set of hyperparameters.
  - **Validation set**: used to find good hyperparameters.
  - **Test set**: used to evaluate performance.

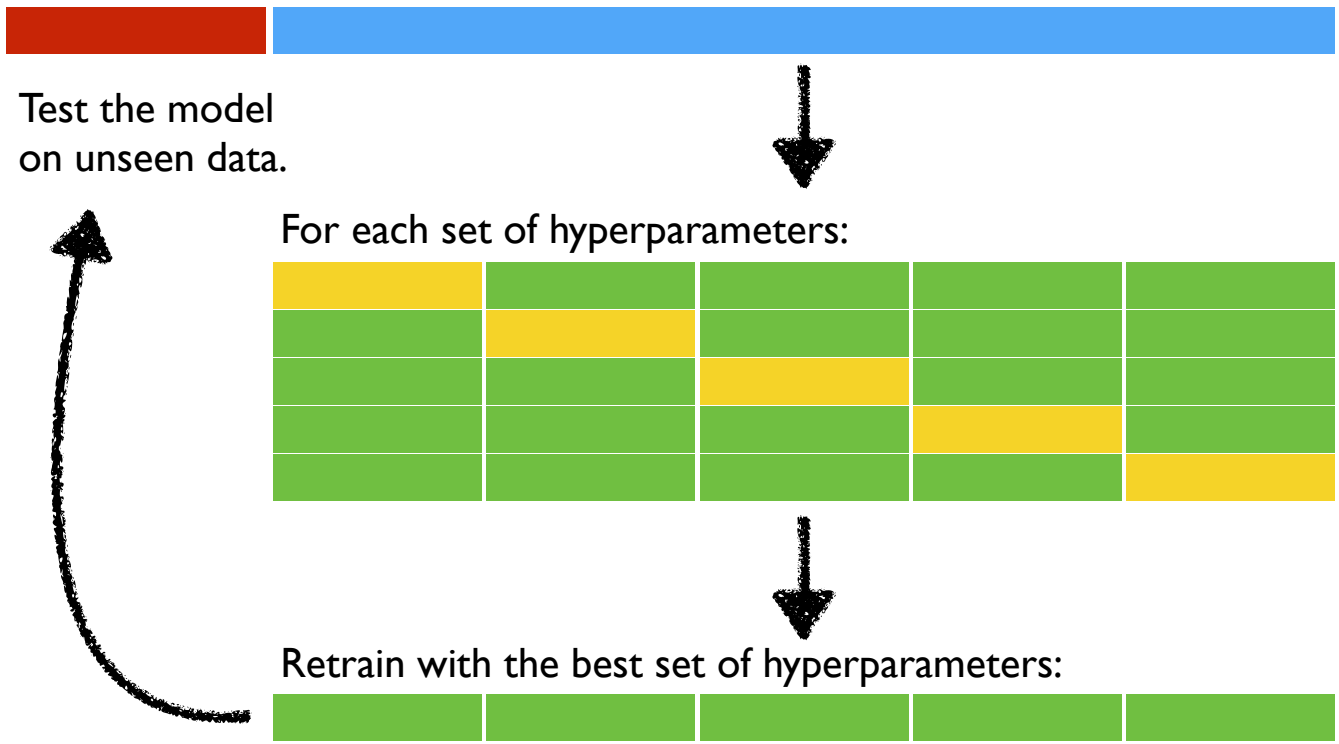
Performance evaluation in deep learning

- If we want to use make of the entire training set, we will perform nested cross validation:



Performance evaluation in deep learning

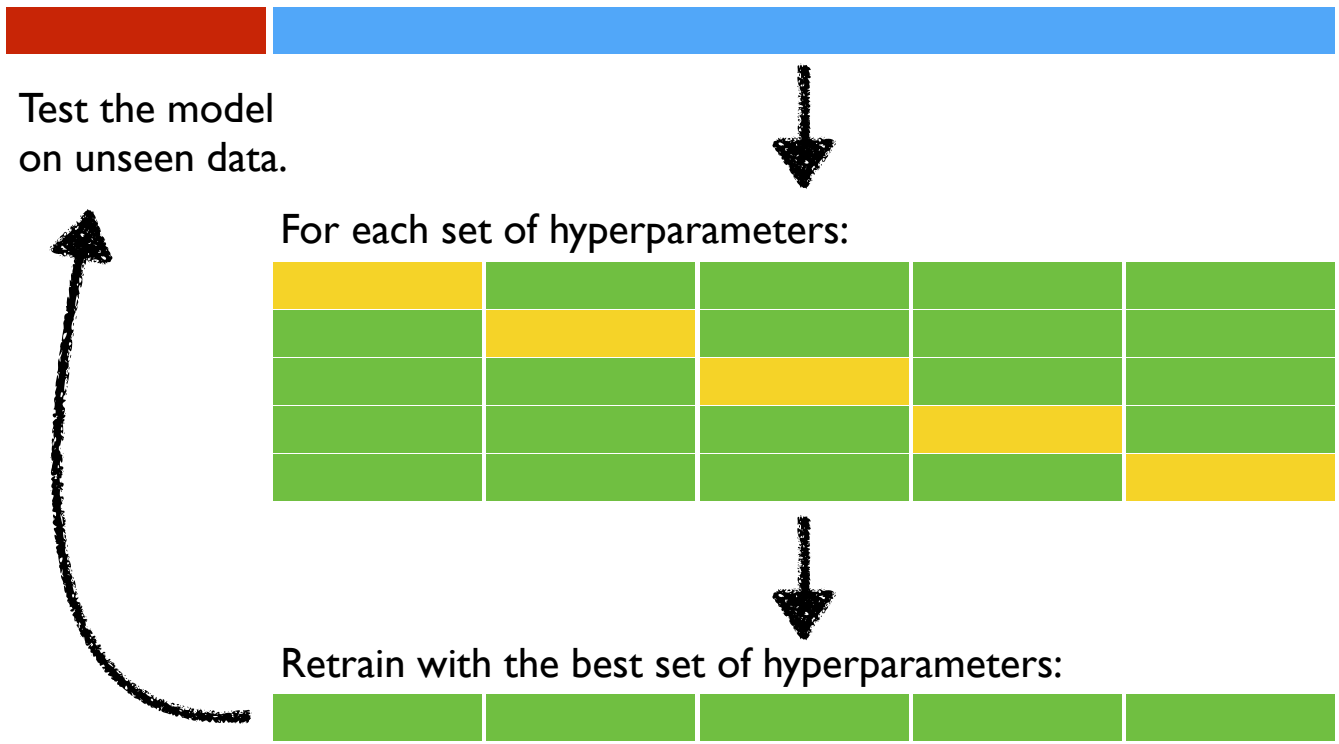
- If we want to use make of the entire training set, we will perform nested cross validation:



- Nested cross validation is extremely time consuming.

Performance evaluation in deep learning

- If we want to use make of the entire training set, we will perform nested cross validation:

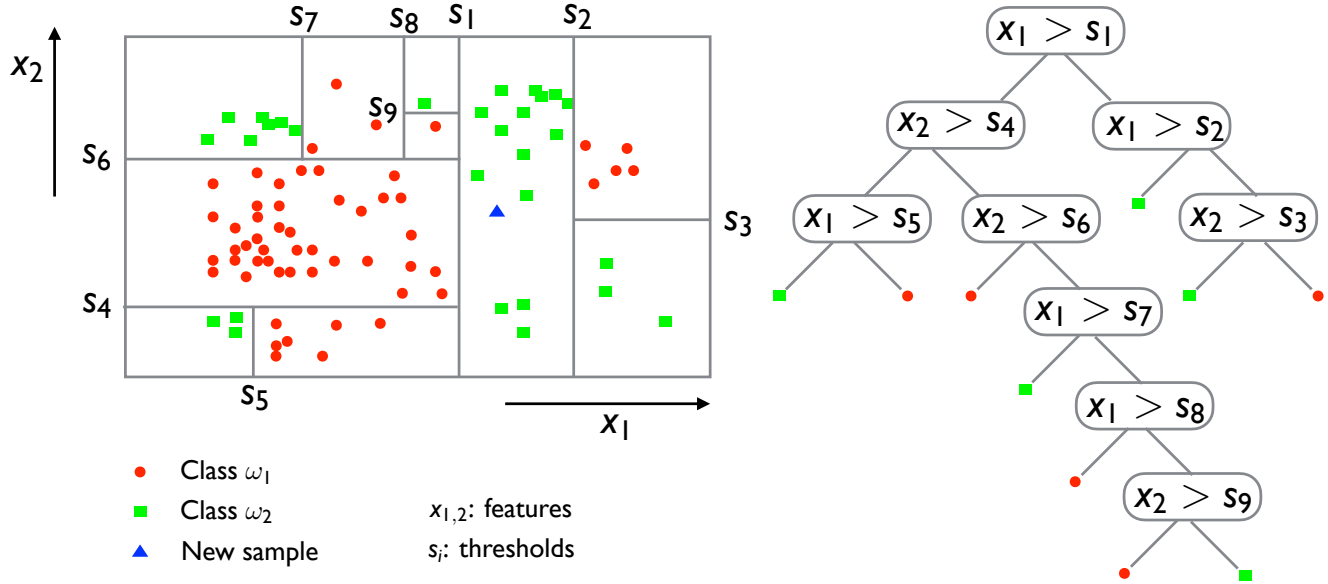


- Nested cross validation is extremely time consuming.
- In the context of deep learning with long training times, nested cross validation is rarely used.

Overview

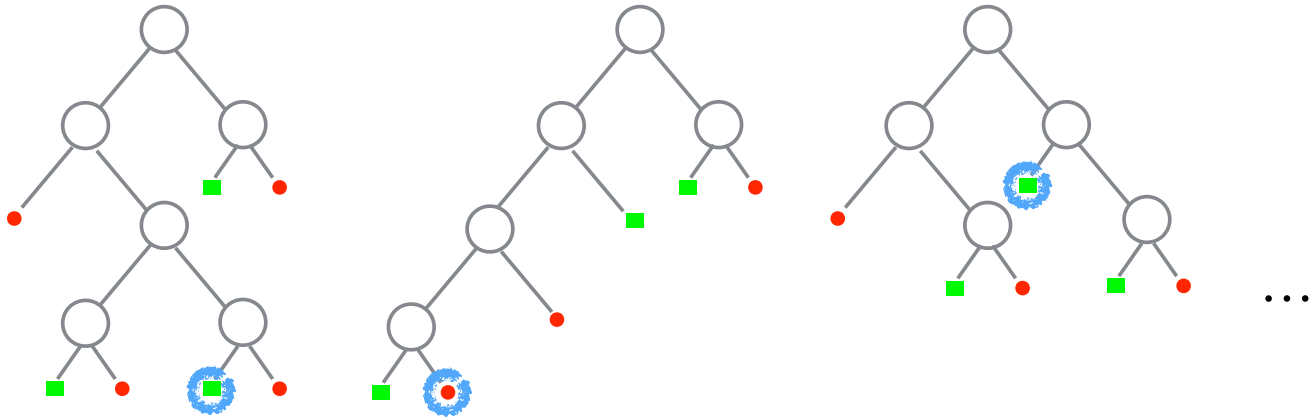
Random Forests

Random Forest: Decision trees



- Decision trees correspond to a series of binary decisions that partition the feature space.
- Classification of a new object: application of the binary decision rules and assignment of the leaf label.
- The decision boundaries can be very complex and adapt very tightly to the training set.

Random Forests



- While decision trees can approximate very complicated decision boundaries, they tend to fit too much to the training data (overfitting).
- Random forests: set of decision trees, each learned on a different (randomly drawn) portion of the data and with different (randomly selected) features.
- Each tree gives a classification result.
- The final result is obtained by a majority vote.

Random Forests

- Because each tree is slightly different from the others, each tree learns a slightly different aspect of the training data.
- A classification method that exists in averaging the results of several classifiers (often weak classifiers), is called **ensemble method**.
- The strategy of averaging is a form of regularization.
- In practice, model averaging is very popular in the deep learning field. Often, one averages simply the output of several networks to obtain a more robust classifier.

Linear Discriminant Analysis

The Bayes rule of classification 1/2

Bayes rule of classification

Bayes Theorem: Let $x \in \mathbb{R}^P$ with a probability density $p(x)$ and $y \in \{1 \ldots, K\}$ a discrete set of class labels. The posterior probability $P(y = k \,|\, x)$ can be written as:

$$P(y = k \,|\, x) = \frac{p(x \,|\, y = k) P(y = k)}{p(x)} \tag{14}$$

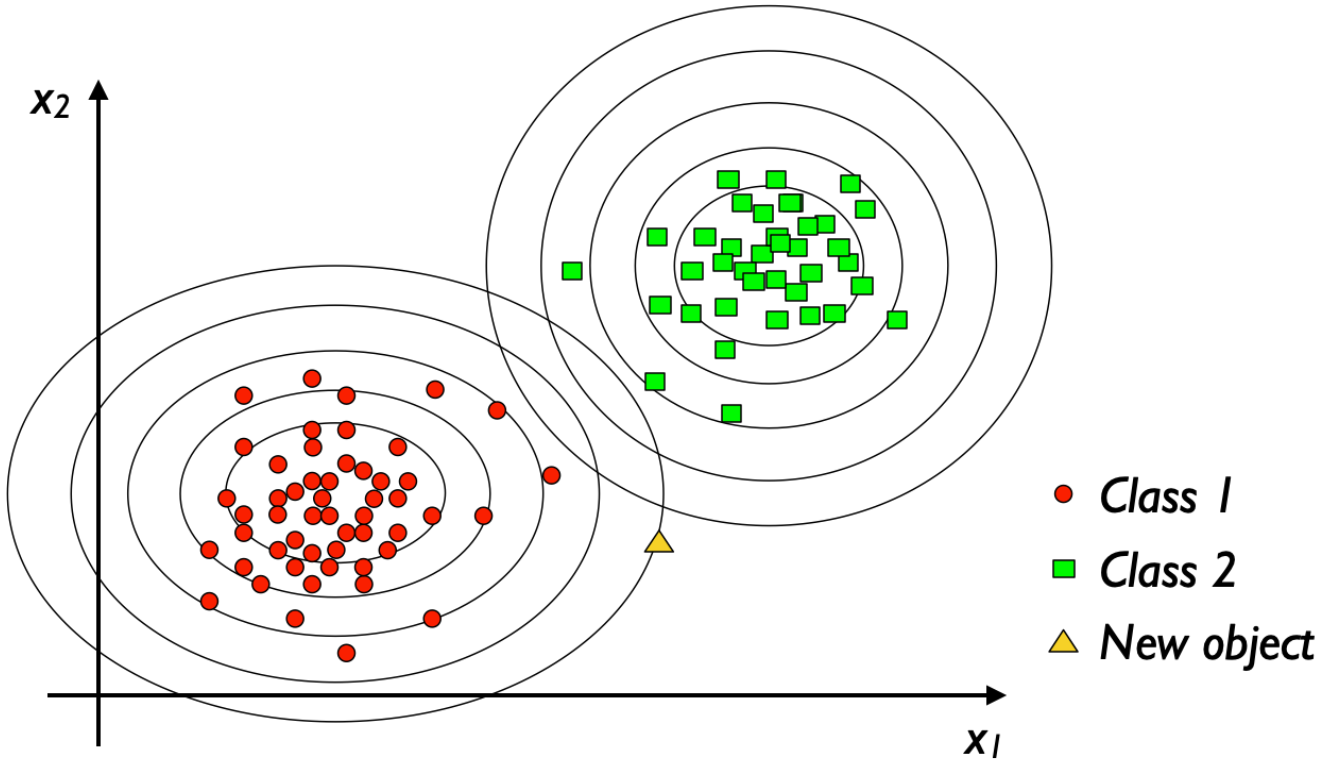
The Bayes rule of classification: *choose the class that maximizes the posterior probability* $P(y = k|x)$:

$$\hat{y}(x) = \arg\max_k P(y = k \,|\, x) = \arg\max_k p(x \,|\, y = k) P(y = k) \tag{15}$$

The Bayes rule of classification 2/2

- The Bayes rule of classification simply states that the best class to choose is the one with highest probability given the observation.
- Importantly, we can calculate this posterior probability from the class dependent feature distributions and the class probabilities.
- This rule is important because it gives the best possible classifier, given that the class dependent feature distributions $p(x \mid y = k)$ and the class probabilities $P(y = k)$ are known.
- However, this is normally not the case: these distributions need to be estimated from the training data.

Normality assumption



We assume that features are normally distributed for each of the classes:

$$p(x|y=k) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)} \qquad (16)$$

Linear Discriminant Analysis (LDA)

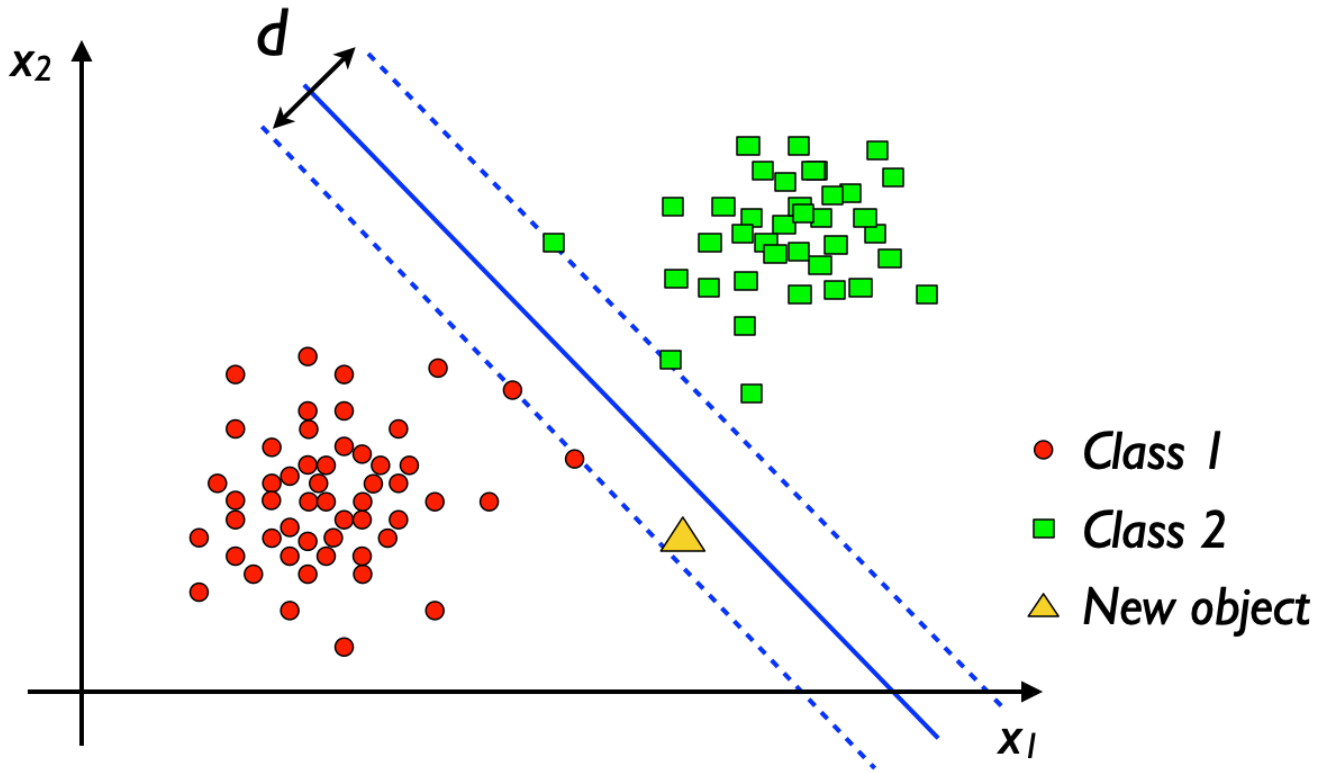
Plugging equation (16) into (15) and the additional assumption that the covariance matrices of the classes are equal, i.e. $\Sigma_1 = \Sigma_2 = \Sigma$, we obtain **Linear Discriminant Analysis, LDA**:

$$\log \frac{P(y=1)}{P(y=2)} + x^T \Sigma^{-1}(\mu_1 - \mu_2) - \frac{1}{2}(\mu_1 - \mu_2)^T \Sigma^{-1}(\mu_1 + \mu_2) > 0$$

- We see that this is a linear classifier: x is multiplied with $\Sigma^{-1}(\mu_1 - \mu_2)$.
- *LDA* is an old technique, but it is still used in practice.
- With neural networks, we typically aim at predicting posterior probabilities without parametric modeling of class dependent densities.

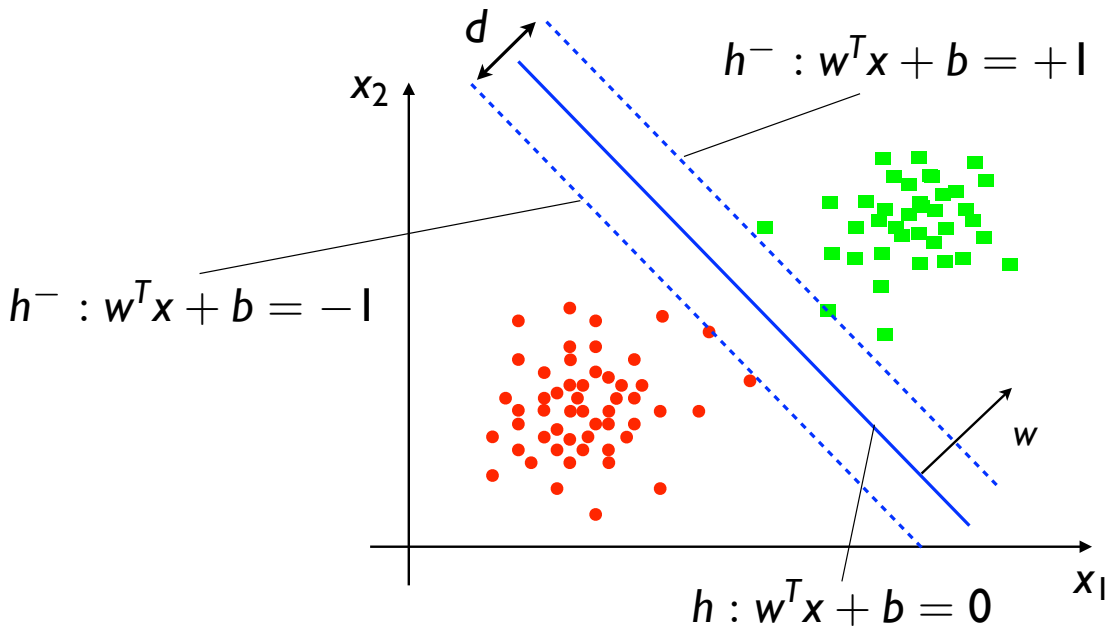
Support Vector Machines

Support Vector Machines: principle



- Optimal placement of a linear decision boundary.
- Intuitive approach: place a "ribbon" instead of a single line, i.e. two parallel lines separated by a distance d.
- Maximization of the width d in order to push the separating hyperplane away from the two classes.

Support Vector Machines: linearly separable training set

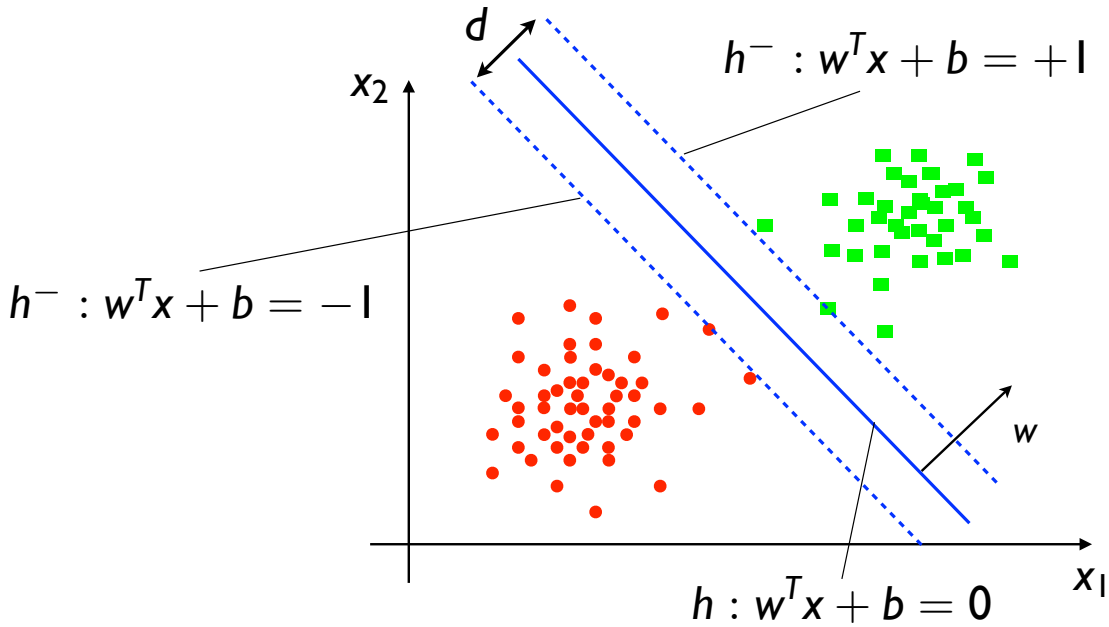


- The ribbon is defined by two parallel lines, for which $w^T x + b = 1$
- The distance between the two parallel lines $w^T x + b = \pm 1$ is given by

$$d = \frac{2}{\|w\|} \tag{17}$$

- Maximizing of d is equivalent to minimizing $\|w\|^2$.

Support Vector Machines: linearly separable training set

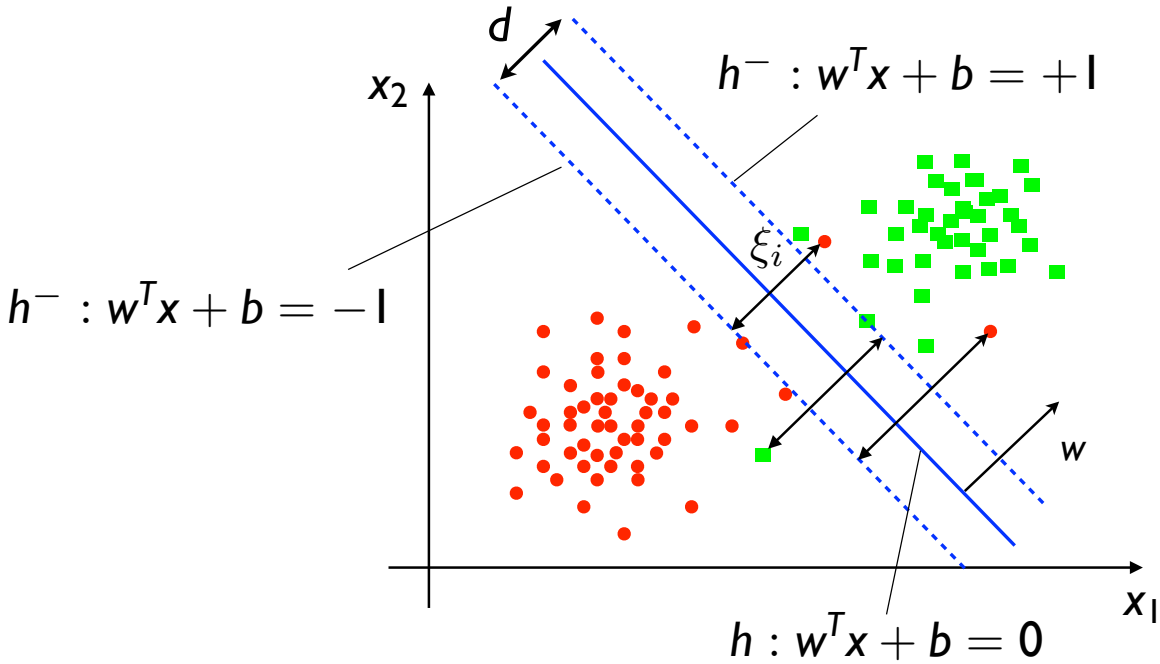


- With this, the problem can be written as a **convex optimization problem under constraints**:

$$\begin{aligned} \text{minimize} \quad & \|w\|^2 \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1 \quad i = 1, \ldots, N \end{aligned}$$

- Convexity implies that there is no local minimum besides the global minimum.

SVM for not linearly separable data

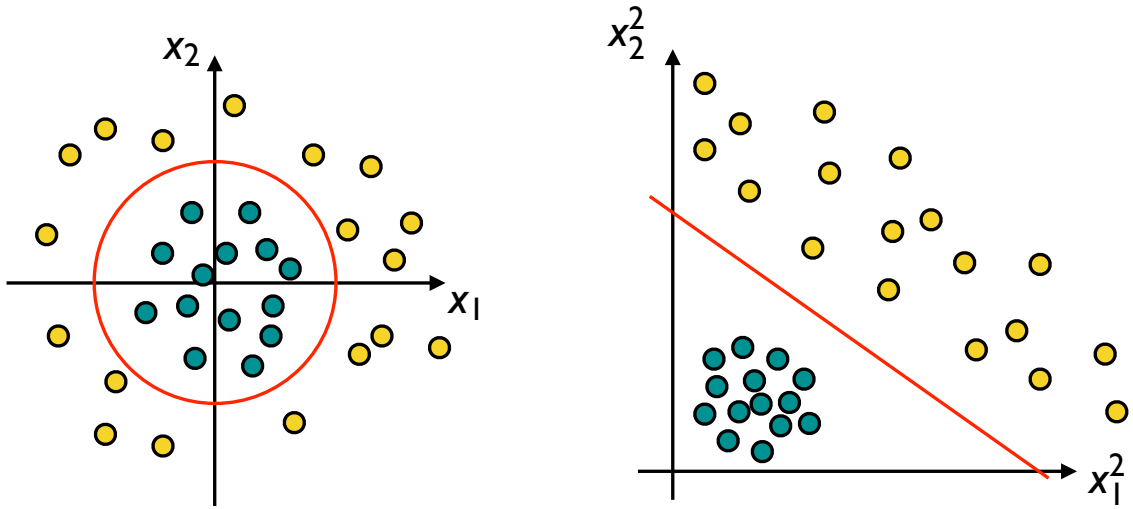


- In reality, complex data is rarely linearly separable.
- In this case, we need to find a compromise between error term $\sum_{i=1}^{N} \xi_i$ and regularization $\|w\|^2$ (we omit the constraints):

$$\min_{w,\xi} \|w\|^2 + C \sum_{i=1}^{N} \xi_i$$

- We note that this corresponds to the general formulation we have seen before.

Support Vector Machines: Kernel trick



- Limitation: linear decision boundaries
- Solution: transform the features in a higher dimensional space.
- SVM: in-built mechanism to do such transformations efficiently.
- Finding good representations is at the very heart of Neural Networks.

Conclusion

- Supervised Machine Learning is concerned with inferring a function f from a set of annotated data, allowing to predict some output variable y from inputs x.
- Different views:
  - Probabilistic view: we maximize the posterior probability.
  - Discriminant view: we optimize the separation of classes.
- Compromise: we want to minimize the error and regularize f.
- Forms of regularization we have seen so far:
  - Regularization by model averaging (e.g. RF)
  - Minimizing a global loss function containing an error and a regularization term:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) + \lambda \mathcal{R}(\boldsymbol{\theta})$$

- All the methods, we have seen so far work on a fixed representation of the objects (often a vector $x \in \mathbb{R}^P$).

References I

[Abràmoff et al., 2018] Abràmoff, M. D., Lavin, P. T., Birch, M., Shah, N., and Folk, J. C. (2018). Pivotal trial of an autonomous AI-based diagnostic system for detection of diabetic retinopathy in primary care offices. *npj Digital Medicine*, 1(1):39.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

[Esteva et al., 2017] Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115–118.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.

References II

[Poplin et al., 2018] Poplin, R., Varadarajan, A. V., Blumer, K., Liu, Y., McConnell, M. V., Corrado, G. S., Peng, L., and Webster, D. R. (2018). Prediction of cardiovascular risk factors from retinal fundus photographs via deep learning. *Nature Biomedical Engineering*, 2(3):158–164.