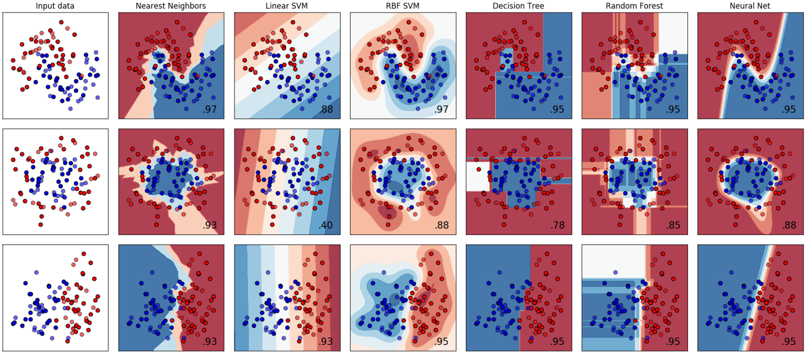
Machine Learning

Supervised learning

Maxime Gasse



http://scikit-learn.org

Introduction

Given an observation $\mathbf{x}$, make a prediction $\hat{\mathbf{y}}$.

Example

- customer bank records $\rightarrow$ solvency
- patient symptoms $\rightarrow$ disease
- apartment address, surface, year $\rightarrow$ price
- mushroom picture $\rightarrow$ variety

We have a data set $\mathcal{D}$ of previously observed $(\mathbf{x}, \mathbf{y})$ pairs.

Decision theory

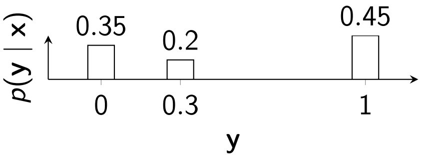
Prediction

Find a mapping $\mathbf{h}$ from an input space $\mathcal{X}$ to an output space $\mathcal{Y}$,

$$\mathbf{h} : \mathcal{X} \to \mathcal{Y}.$$

Example (regression)

$\mathcal{Y} = \mathbb{R}^1$. For a given $\mathbf{x}$, we know $p(\mathbf{y} \mid \mathbf{x})$:



You are asked to predict a value for $\mathbf{y}$. What is your answer?

Bayes-optimal prediction

Cost of $\hat{\mathbf{y}}$ instead of $\mathbf{y}$? Loss function:

$$L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}.$$

Risk of $\mathbf{h}(\mathbf{x}) = \hat{\mathbf{y}}$? Expected loss:

$$\begin{aligned}\mathbb{E}_{\mathbf{y}|\mathbf{x}}[L(\hat{\mathbf{y}}, \mathbf{y})] &= \sum_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x}) \times L(\hat{\mathbf{y}}, \mathbf{y}) \quad \text{(classification)} \\ &= \int_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x}) \times L(\hat{\mathbf{y}}, \mathbf{y}) d\mathbf{y} \quad \text{(regression)}.\end{aligned}$$

Bayes-optimal prediction $\iff$ risk-minimization

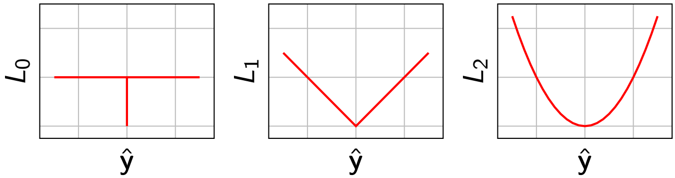
$$\mathbf{h}_L^\star(\mathbf{x}) = \arg\min_{\hat{\mathbf{y}}} \mathbb{E}_{\mathbf{y}|\mathbf{x}} L(\hat{\mathbf{y}}, \mathbf{y}).$$

Example: univariate regression

Popular loss functions for regression:

- squared error: $L_2(\hat{\mathbf{y}}, \mathbf{y}) = \sum_i (\hat{y}_i - y_i)^2$;
- absolute error: $L_1(\hat{\mathbf{y}}, \mathbf{y}) = \sum_i |\hat{y}_i - y_i|$;
- zero-one error: $L_0(\hat{\mathbf{y}}, \mathbf{y}) = 1$ for every $\hat{\mathbf{y}} \neq \mathbf{y}$.

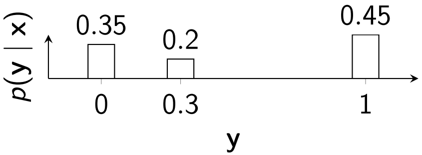
Illustration in $\mathbb{R}^1$ ($\mathbf{y}$ fixed):



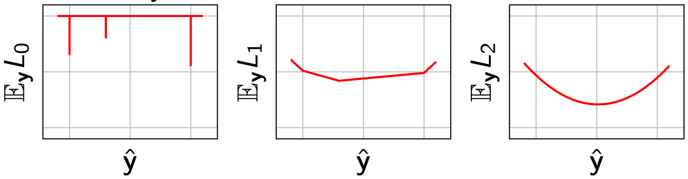
How could this affect the risk-minimizer $\mathbf{h}^\star$?
Because of uncertainty in $p(\mathbf{y} \mid \mathbf{x})$...

Example: univariate regression

For a given $\mathbf{x}$:



Expected loss: $\sum_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x}) L(\hat{\mathbf{y}}, \mathbf{y})$



$\mathbf{h}^\star_{L_0}(\mathbf{x}) = 1$ (mode), $\mathbf{h}^\star_{L_1}(\mathbf{x}) = 0.3$ (median), $\mathbf{h}^\star_{L_2}(\mathbf{x}) = 0.51$ (mean).

Plug-in risk minimization

$\mathcal{D}$ a set of i.i.d. data samples $\{(\mathbf{x}, \mathbf{y})^{(i)}\}_{i=1}^{N}$ drawn from $p(\mathbf{x}, \mathbf{y})$,
$\mathcal{Q}$ a restricted probabilistic model (e.g. parametric distribution).

Learn $q^\star$ via maximum log-likelihood:

$$\arg\max_{q \in \mathcal{Q}} \sum_{i=1}^{N} \log q(\mathbf{y}^{(i)} | \mathbf{x}^{(i)}).$$

Risk-minimizing prediction: $\mathbf{h}^\star(\mathbf{x}) = \arg\min_{\hat{\mathbf{y}}} \sum_{\mathbf{y}} q^\star(\mathbf{y}|\mathbf{x}) L(\hat{\mathbf{y}}, \mathbf{y})$.

Bayes-optimal under two conditions:

- statistical sufficiency $\mathcal{D} \approx p$ (e.g. $N \to \infty$);
- model sufficiency $p \in \mathcal{Q}$ (e.g. $\mathcal{Q}$ unrestricted).

Example: multinomial regression, naive Bayes...

Example: binary classification

Loss function:

L		$\hat{\mathbf{y}} = 0$	$\hat{\mathbf{y}} = 1$
$\mathbf{y}$	0	0	α
	1	β	0

Expected loss:

$$\mathbb{E}_{\mathbf{y}|\mathbf{x}}[L(1, \mathbf{y})] = p(\mathbf{y} = 0|\mathbf{x}) \times \alpha$$
$$\mathbb{E}_{\mathbf{y}|\mathbf{x}}[L(0, \mathbf{y})] = p(\mathbf{y} = 1|\mathbf{x}) \times \beta$$

Bayes-optimal prediction:

$$\mathbf{h}^\star(\mathbf{x}) = \left[p(\mathbf{y} = 1|\mathbf{x}) > \frac{\alpha}{\alpha + \beta} \right].$$

Example: binary classification

Loss function:

L		$\hat{\mathbf{y}} = 0$	$\hat{\mathbf{y}} = 1$
$\mathbf{y}$	0	0	2
	1	5	0

Expected loss:

$$\mathbb{E}_{\mathbf{y}|\mathbf{x}}[L(1, \mathbf{y})] = p(\mathbf{y} = 0|\mathbf{x}) \times 2$$
$$\mathbb{E}_{\mathbf{y}|\mathbf{x}}[L(0, \mathbf{y})] = p(\mathbf{y} = 1|\mathbf{x}) \times 5$$

Bayes-optimal prediction:

$$\mathbf{h}^\star(\mathbf{x}) = [p(\mathbf{y} = 1|\mathbf{x}) > 28.6\%] \,.$$

Direct risk minimization

$\mathcal{D}$ a set of i.i.d. data samples $\{(\mathbf{x}, \mathbf{y})^{(i)}\}_{i=1}^{N}$ drawn from $p(\mathbf{x}, \mathbf{y})$,
$\mathcal{H}$ a restricted hypothesis space (e.g. parametric model).

Learn $\mathbf{h}^\star$ directly via empirical risk minimization:

$$\arg\min_{\mathbf{h} \in \mathcal{H}} \sum_{i=1}^{N} L(\mathbf{h}(\mathbf{x}^{(i)}), \mathbf{y}^{(i)}).$$

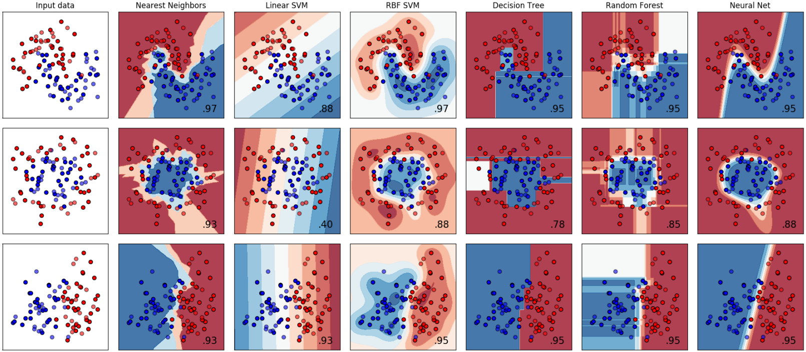
Bayes-optimal under two conditions:

- statistical sufficiency $\mathcal{D} \approx p$ (e.g. $N \to \infty$);
- model sufficiency $\mathbf{h}^\star \in \mathcal{H}$ (e.g. $\mathcal{H}$ unrestricted).

Example: linear regression, decision tree, SVM...

Decision boundaries

$$\mathbf{y} \in \{\text{blue}, \text{red}\}, \text{ zero-one error } L_{0/1} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$



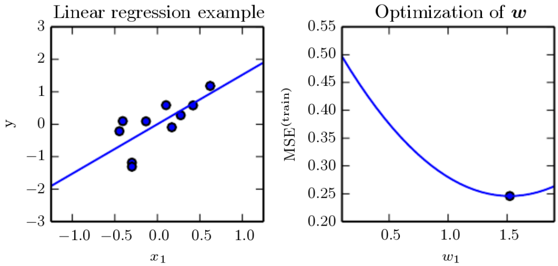
http://scikit-learn.org/

Statistical learning

Linear regression

Linear hypotheses: $\mathcal{H} = \{\mathbf{h} \mid \mathbf{h}(\mathbf{x}) = \mathbf{A}^\intercal \mathbf{x} + \mathbf{b}\}$.
Reformulation: $\mathbf{x} = (1, x_1, \ldots, x_d)$, so that $\mathbf{h} = \mathbf{W}^\intercal \mathbf{x}$.

$$\mathbf{h}^\star = \arg\min_{\mathbf{h}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[L_2(\mathbf{h}(\mathbf{x}), \mathbf{y})],$$
$$\text{eq. } \mathbf{W}^\star = \arg\min_{\mathbf{W}} \sum_i |\mathbf{W}^\intercal \mathbf{x}^{(i)} - \mathbf{y}^{(i)}|_2^2.$$



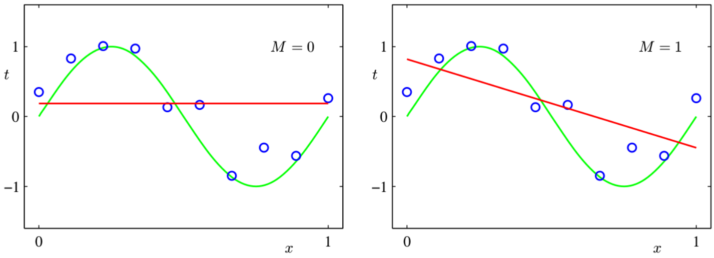
$\mathbb{E}_{L_2}$ convex w.r.t. $\mathbf{W} \implies$ closed-form solution ($\nabla \mathbb{E}_{L_2} = 0$):
$$\mathbf{W}^\star = (\mathbf{X}^\intercal \mathbf{X})^{-1} \mathbf{X}^\intercal \mathbf{Y}.$$

Capacity, overfitting, underfitting

The higher the capacity of $\mathcal{H}$, the more powerful the model.

Example (polynomial regression with degree M)

$$\mathbf{x} = (x_0, x_1^1, \ldots, x_d^1, \ldots, x_1^M, \ldots, x_d^M).$$



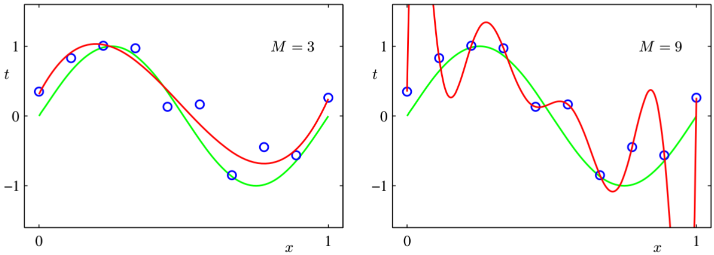
M increases
$N = 10$ (fixed)

Capacity, overfitting, underfitting

The higher the capacity of $\mathcal{H}$, the more powerful the model.

Example (polynomial regression with degree M)

$$\mathbf{x} = (x_0, x_1^1, \ldots, x_d^1, \ldots, x_1^M, \ldots, x_d^M).$$



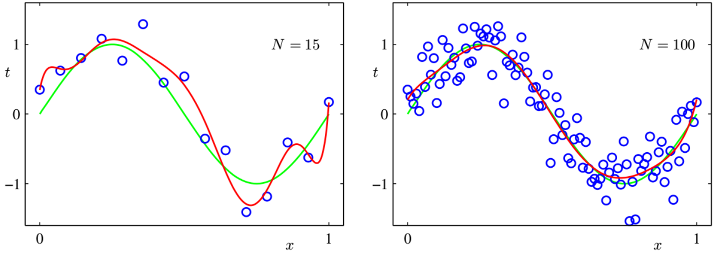
M increases
$N = 10$ (fixed)

Capacity, overfitting, underfitting

The higher the capacity of $\mathcal{H}$, the more powerful the model.

Example (polynomial regression with degree M)

$$\mathbf{x} = (x_0, x_1^1, \ldots, x_d^1, \ldots, x_1^M, \ldots, x_d^M).$$

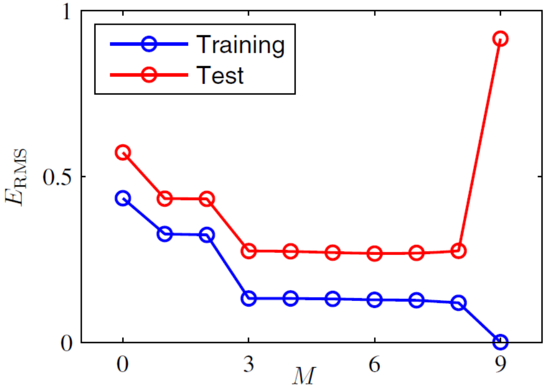


$M = 9$ (fixed)
N increases

Measuring under/over-fitting

Split $\mathcal{D}$ into a training set $\mathcal{D}_{train}$ and a test set $\mathcal{D}_{test}$.

Test error increases $\implies$ overfitting...



Always evaluate your model on a separate test set !

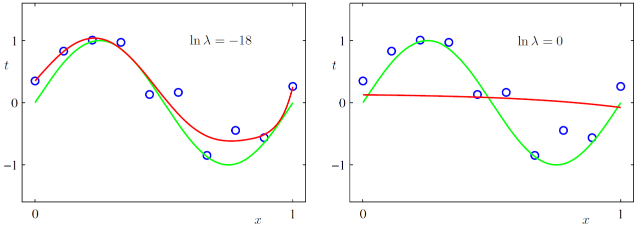
Regularization

Restrict $\mathcal{H}$ to consider only simple hypotheses.

Example: penalize high coefficients in $\mathbf{W}$.

$$\mathbf{W}^\star = \arg\min_{\mathbf{W}} \sum_i L(\mathbf{y}^{(i)}, \mathbf{w}^\intercal \mathbf{x}^{(i)}) + \lambda \|\mathbf{w}\|_2^2,$$

with λ a hyper-parameter.



Regularization / prior equivalence

Let $\mathcal{D} = \{\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \dots\}$ sampled from p with parameters θ.

Maximum a-posteriori (MAP) estimate:

$$
\begin{aligned}
\theta^\star &= \arg\max_\theta p(\theta|\mathcal{D}) && \text{posterior} \\
&= \arg\max_\theta p(\theta|\mathcal{D}) \times p(\mathcal{D}) && \\
&= \arg\max_\theta p(\theta, \mathcal{D}) && \text{joint} \\
&= \arg\max_\theta p(\mathcal{D}|\theta) \times p(\theta) && \text{likelihood} \times \text{prior} \\
&= \arg\max_\theta \prod_i p(\mathbf{v}^{(i)}|\theta) \times p(\theta) && \text{(i.i.d. hypothesis)} \\
&= \arg\max_\theta \sum_i \log p(\mathbf{v}^{(i)}|\theta) + \log p(\theta) && \text{log-lik.} + \text{reg}
\end{aligned}
$$

Regularization / prior equivalence

Let $\mathcal{D} = \{(x, y)^{(1)}, (x, y)^{(2)}, \dots\}$.

Assume a Gaussian model and a Gaussian prior:
$p(y|x, \theta) = \mathcal{N}(y|\beta x, \sigma^2)$ and $p(\beta) = \mathcal{N}(\beta|0, \lambda^{-1})$

$$\begin{aligned}\beta^\star &= \arg\max_\beta \prod_i \mathcal{N}(y^{(i)}|\beta x^{(i)}, \sigma^2) \times \mathcal{N}(\beta|0, \lambda^{-1}) \\ &= \arg\min_\beta \sum_i \frac{1}{\sigma^2}(y^{(i)} - \beta x^{(i)})^2 + \lambda\beta^2 \qquad \text{(neg. log)}\end{aligned}$$

In short:

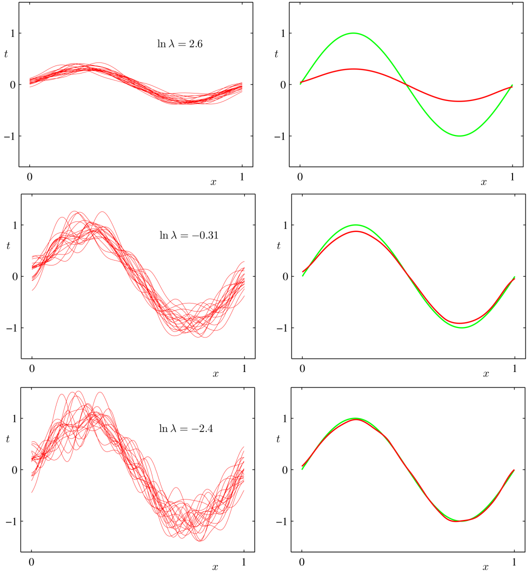
- L_2 regularization $\approx$ Gaussian prior;
- L_1 regularization $\approx$ Laplacian prior.

Measuring bias/variance

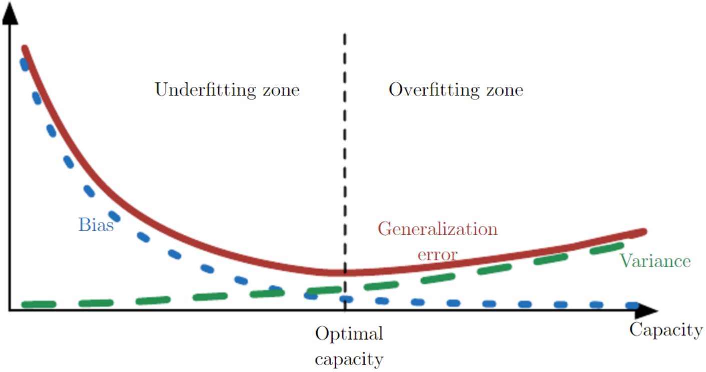
$M = 9$, λ varies.

Repeat with different training sets of same size ($N = 25$).

L: 100 reps (variance).
R: average (bias).



Optimal capacity



Every model has hyper-parameters: k-NN (k, metric), SVM (kernel, C, γ), decision tree (depth), neural net (weight decay, learning rate). Choosing the model is also a hyper-parameter.

Again, using $\mathcal{D}_{test}$ too much for tuning leads to overfitting...

Hyper-parameter tuning

Let θ be our parameters and λ our hyper-parameters.

Split $\mathcal{D}$ into $\mathcal{D}_{train}$ / $\mathcal{D}_{valid}$ / $\mathcal{D}_{test}$.

Grid search:

1. for each $\lambda_i \in \{\lambda_1, \lambda_2, \dots\}$, learn $\theta_i^\star$ on $\mathcal{D}_{train}$;
2. keep parameters $\theta^\star$ that perform best on $\mathcal{D}_{valid}$;
3. evaluate $\theta^\star$ on $\mathcal{D}_{test}$.

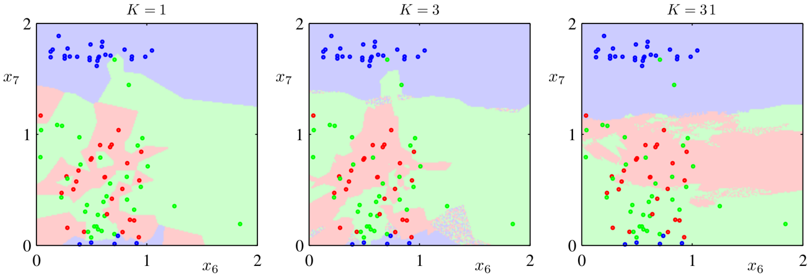
Do not repeat !

K Nearest Neighbours (k-NN)

Main idea

For a given $\mathbf{x}$, let $(1, \ldots, N)$ be an ordering of $\mathcal{D}$ such that $\text{dist}(\mathbf{x}, \mathbf{x}^{(1)}) \leq \cdots \leq \text{dist}(\mathbf{x}, \mathbf{x}^{(N)})$, and $\mathbf{Y}_{\text{nb}(\mathbf{x})} = \{\mathbf{y}^{(i)} | i \leq K\}$. Then,

$$\mathbf{h}^\star(\mathbf{x}) = \arg\min_{\hat{\mathbf{y}}} \sum_{i=1}^{K} L(\hat{\mathbf{y}}, \mathbf{y}^{(i)}).$$



Parameters: number of neighbours (K), distance metric (dist).

In practice

Pros:

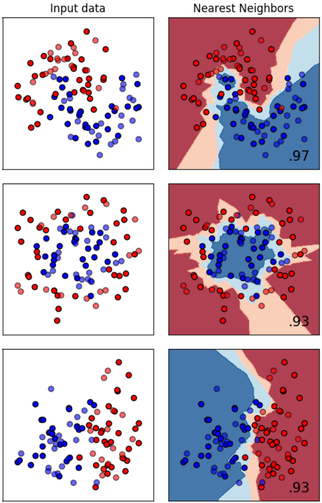
- + infinite capacity
- + no training

Cons:

- – expensive predictions, $O(N)$ in memory and time
- – distance metric in high dimensions ?

Variants: neighbours weighting

$$\sum_{i=1}^{K} (\mathrm{d}_{\max} - \mathrm{dist}(\mathbf{x}, \mathbf{x}^{(i)})) L(\hat{\mathbf{y}}, \mathbf{y}^{(i)}).$$

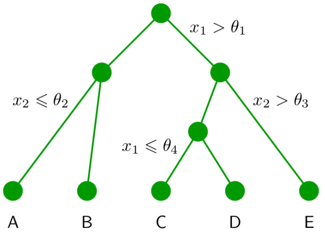


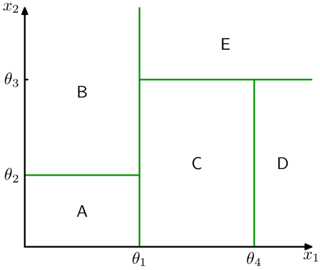
Classification and Regression Tree (CART)

Decision tree

For a given $\mathbf{x}$, let $\mathbf{Y}_{\text{bin}(\mathbf{x})} = \{\mathbf{y}^{(i)} | \text{bin}(\mathbf{x}^{(i)}) = \text{bin}(\mathbf{x})\}$. Then,

$$\mathbf{h}^{\star}(\mathbf{x}) = \arg\min_{\hat{\mathbf{y}}} \sum_{\mathbf{y} \in \mathbf{Y}_{\text{bin}(\mathbf{x})}} L(\hat{\mathbf{y}}, \mathbf{y}).$$





Parameters: construction strategy, max depth, min leaf size.

Build the tree

Learning algorithm: start for a single root node, then repeat

1. pick one expandable node (non-terminal)
2. pick the (x_j, θ) split that maximizes the information gain;
3. create child nodes: $(x_j \leq \theta)$ and $(x_j > \theta)$.

Information gain (IG):

$$IG(\mathcal{D}_{par}) = N_{par} I(\mathcal{D}_{par}) - N_{left} I(\mathcal{D}_{left}) - N_{right} I(\mathcal{D}_{right}).$$

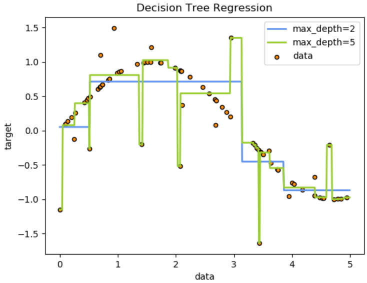
For classification:

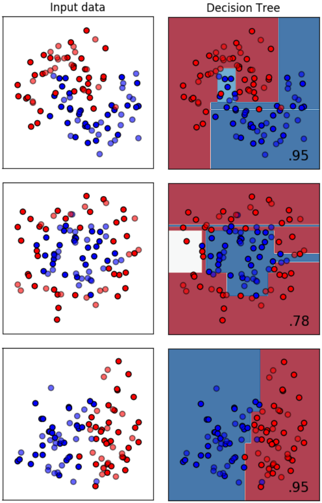
- Gini impurity: $\sum_{\mathbf{y} \in \mathcal{Y}} p(\mathbf{y})(1 - p(\mathbf{y}))$
- Entropy: $-\sum_{\mathbf{y} \in \mathcal{Y}} p(\mathbf{y}) \log p(\mathbf{y})$

For regression:

- Variance reduction: $\sum_{y \in \mathcal{D}} (y - \bar{y})^2$

In practice





Pros:

- + infinite capacity
- + fast learning, parallelizable
- + fast prediction, $O(\log(N))$

Cons:

- — orthogonal cuts
- — high variance or high bias

Random Forest

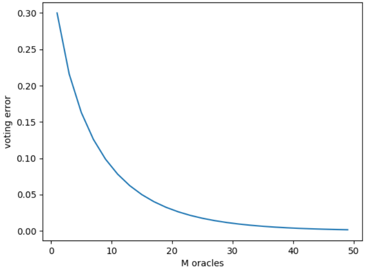
Ensemble learning

Intuitive idea: combine weak learners into a strong one.

Example (independent binary oracles)

Individual oracle error: 30%.
Voting error over m oracles:

$$\sum_{k=m/2+1}^{m} \binom{m}{k} 0.3^k 0.7^{m-k}.$$



Fast training $\implies$ decisions trees.

Models not independent in practice $\implies$ encourage diversity.

- bagging (bootstrap aggregation)
- random feature subsets
- no pruning

In practice

Pros:

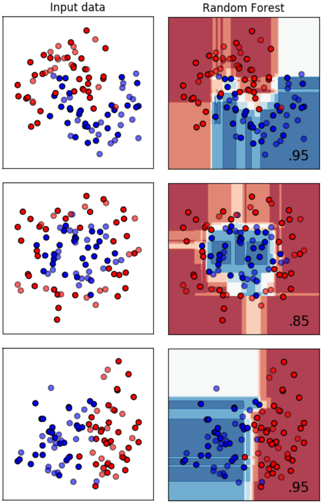
- $+$ good generalization
- $+$ fast learning, parallelizable
- $+$ fast prediction, $O(M \log(N))$

Cons:

- $-$ orthogonal cuts
- $-$ relies on heuristics

Variants:

- Rotation forest (PCA projections)
- Extremely randomized trees
- Gradient boosting



Linear Support Vector Machine (SVM)

Maximum margin separating hyper-plane

Binary classification $y \in \{-1, +1\}$.
Consider $\mathbf{h}(\mathbf{x}) = sign(f(\mathbf{x}))$, and $f(\mathbf{x}) = \mathbf{w}^\intercal \mathbf{x} + b$ (linear model).

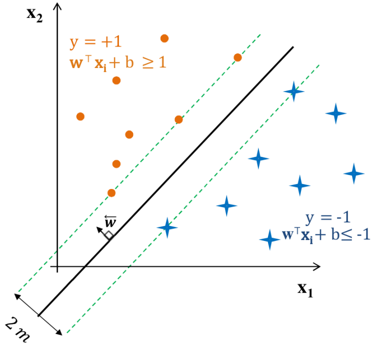
Suppose zero error is possible, we want the separating hyper-plane with maximum margin.

<u>Zero error</u>:

$$y^{(i)}(\mathbf{w}^\intercal \mathbf{x}^{(i)} + b) > 0,\ \forall i.$$

<u>Maximum margin</u>:

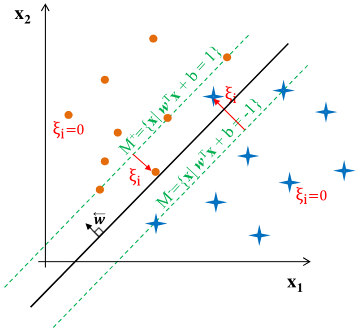
$$\begin{cases} \arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2, \\ \text{s.t.}\ \ y^{(i)}(\mathbf{w}^\intercal \mathbf{x}^{(i)} + b) > 1,\ \forall i. \end{cases}$$



Tolerate classification errors

Introduce slack variables.
$\xi = max(0, 1 - y \times f(\mathbf{x}))$ (hinge loss)

$$\begin{cases} \arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2 + C \sum_i \xi_i, \\ \text{s.t. } \; y^{(i)}(\mathbf{w}^\intercal \mathbf{x}^{(i)} + b) \geq 1 - \xi_i, \; \forall i, \\ \text{and } \; \xi_i \geq 0, \; \forall i. \end{cases}$$



$\Longrightarrow$ convex optimization problem.

In practice

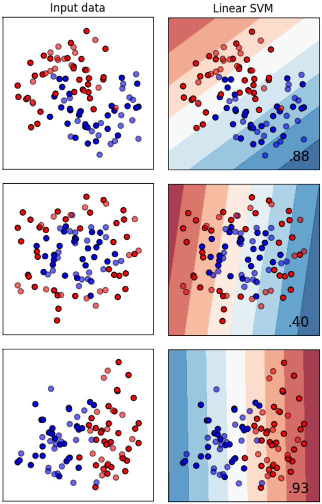
Pros:

- + elegant formulation
- + max-margin separator

Cons:

- – hyper-parameter tuning (C)
- – training complexity $O(ND)$
- – linear decision boundaries

Still missing the main part: kernel trick.



Kernel Support Vector Machine (SVM)

Dual formulation

Karush-Kuhn-Tucker (KKT) multipliers (Lagrange with inequality constraints):

$$\begin{cases} \arg\max_{\boldsymbol{\alpha}} -\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y^{(i)} y^{(y)} \mathbf{x}^{(i)\intercal} \mathbf{x}^{(j)}, \\ \quad \text{s.t.} \;\; 0 \leq \alpha_i \leq C, \; \forall i, \\ \quad \text{and} \;\; \sum_i \alpha_i y^{(i)} = 0. \end{cases}$$

$\implies$ convex optimization problem.

Note:

- support vectors: samples where $\alpha_i > 0$;
- $f(\mathbf{x}) = \sum_i \alpha_i y^{(i)} \mathbf{x}^{(i)\intercal} \mathbf{x}$ (eq. $\mathbf{w} = \sum_i \alpha_i y^{(i)} \mathbf{x}^{(i)}$);

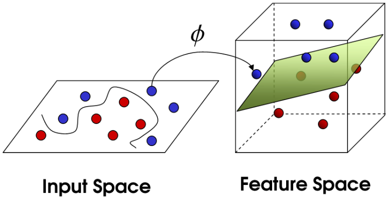
Kernel trick

Feature projection $\implies$ non-linear decision boundary

$\phi : \mathcal{X} \mapsto \mathcal{Z}$

$k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{Z}}$

$f(\mathbf{x}) = \sum_i \alpha_i y^{(i)} k(\mathbf{x}^{(i)}, \mathbf{x})$



Mercer's theorem: no need for an explicit ϕ, a continuous positive semi-definite kernel is enough.

Gaussian radial basis function (RBF) kernel:

- $k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|_2^2)$;
- ϕ exists, with $\mathcal{Z}$ an infinite-dimensional space.

Also: polynomial kernel, Fisher kernel, string kernel...

In practice

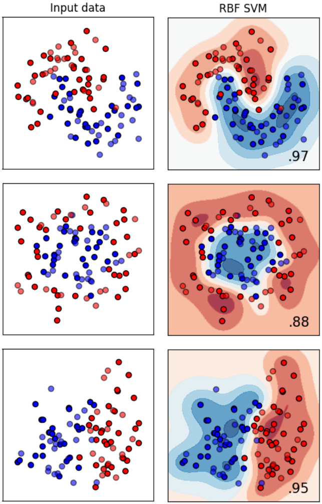
RBF kernel

Pros:

+ elegant formulation
+ max-margin separator
+ non-linear decision boundaries

Cons:

– hyper-parameter tuning (C, λ)
– training complexity $O(N^2 D)$



Artificial Neural Network (ANN)

Covered in next module

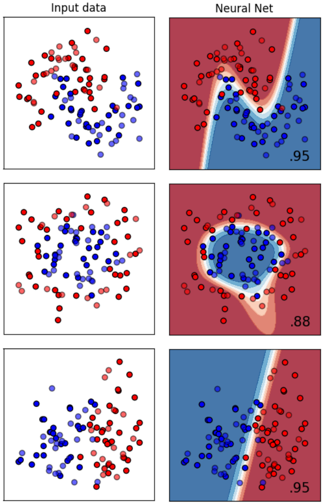
Pros:

- + state-of-the-art on many hard problems
- + scale well to large data sets

Cons:

- – requires large data sets
- – no theoretical guarantee
- – more an art than a science

Try by yourselves:
`http://playground.tensorflow.org`



Course summary

Decision theory

- Bayes-optimal prediction
- plug-in risk minimization
- direct risk minimization

Statistical learning

- linear regression
- the under/over-fitting (or bias/variance) problem
- regularization and hyper-parameters

Supervised learning models

- k-nearest neighbours (k-NN)
- classification and regression tree (CART)
- random forest
- support vector machine (SVM)