BABD

INTERNATIONAL MASTER IN BUSINESS ANALYTICS AND BIG DATA

Classification

POLITECNICO DI MILANO
GRADUATE SCHOOL
OF BUSINESS

Cefriel
POLITECNICO DI MILANO

IBM

FT Executive Education Ranking 2019

FT European Business Schools Ranking 2018

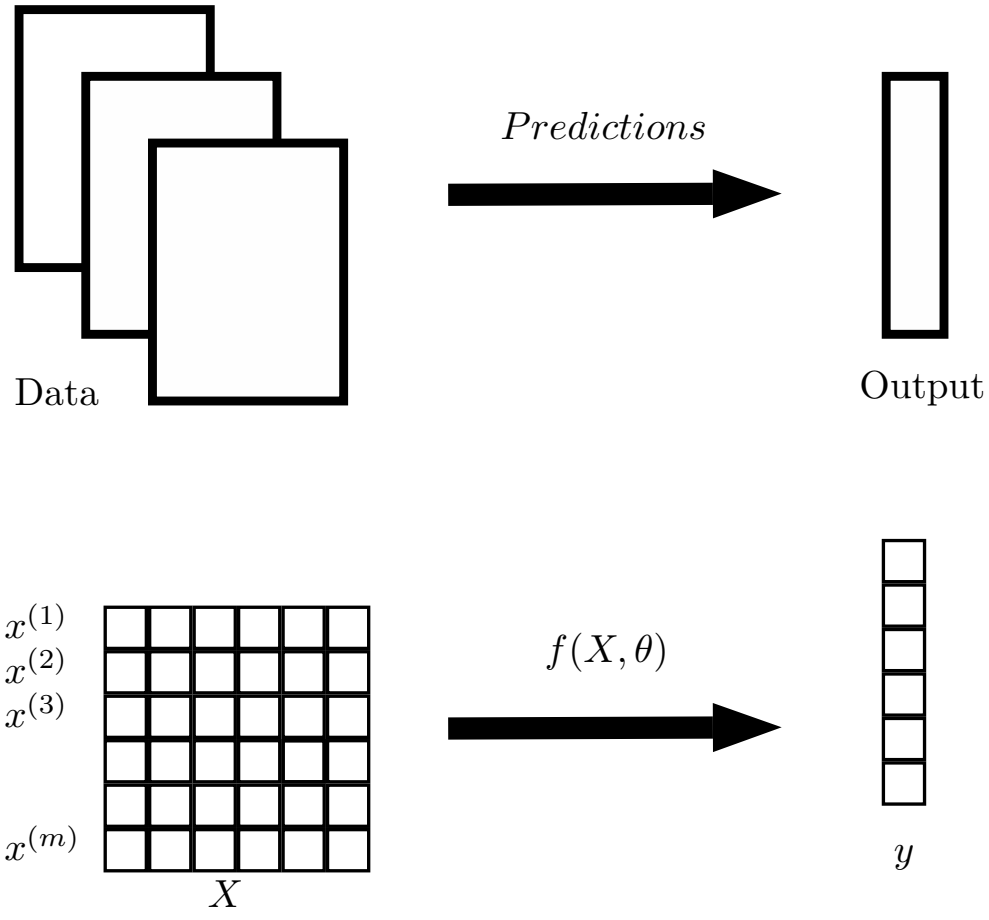
Bloomberg Businessweek 2018 BEST B-SCHOOLS

EFMD EQUIS ACCREDITED

ASSOCIATION AMBA ACCREDITED

EFMD GLOBAL NETWORK EOCCS CERTIFIED

Supervised Learning



The problem: Bank telemarketing[1]

Attribute		Type	Description/Values
Personal	age	num	Age of the potential client
	job	cat	admin., blue- collar, entrepreneur, housemaid,... ,unknown
	marital_status	cat	divorced, married, single, unknown
	education	cat	basic.4y, basic.6y, basic.9y, high.school,.. unknown
Bank	default	cat	The client has credit in default: no,yes,unknown
	housing	cat	The client has a housing loan contract: no,yes,unknown
	loan	cat	The client has a personal loan: no,yes,unknown
Campain	contact	cat	Communication type: cellular,telephone
	month	cat	Last month contacted: jan, feb ,..., dec
	day_of_week	cat	Last contact day : mon, tue,..., fri
	duration	num	Last contact duration (in seconds)
	campain	num	Number of contacts performed during this campaign
	pdays	num	Number of days that passed by after last contact
	previous	num	Number of contacts performed before this campaign
	poutcome	cat	Outcome of the previous marketing campaign: failure,nonexistent,success
Economical	emp.var.rate	num	Employment variation rate in the last quarter
	cons.price.idx	num	Consumer price index in the last month
	cons.conf.idx	num	Monthly consumer confidence index
	euribor3m	num	Dayly Euro Interbank Offered Rate
	nr.employed	num	Number of employed citizens in the last quarter (thousands)
Target	**success**	**target**	**0: no, 1: yes**

[1] A data-driven approach to predict the success of bank telemarketing. S. Moroa, P. Cortez, P. Rita.Decision Support Systems, 62:22-31, 2014.

Classification problem

attributes

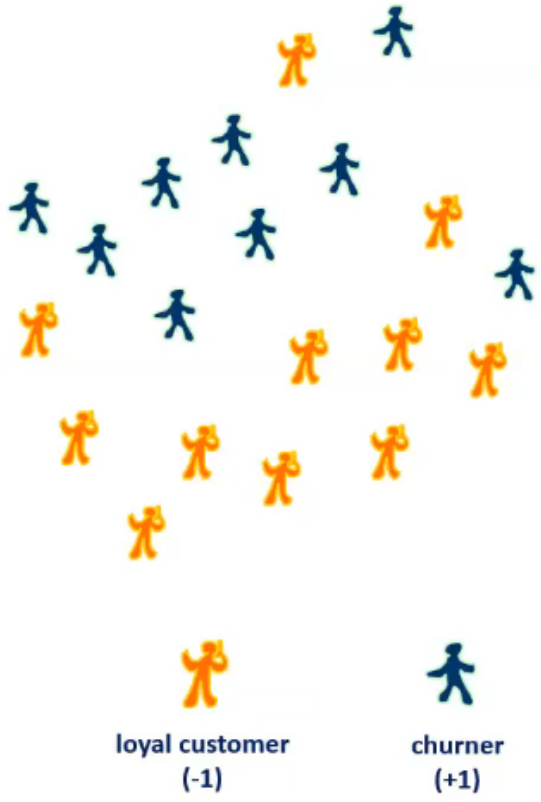
customers

Area	Pothers	Pmob	...	NumSMS	Class
2	0.14	0.59	...	18	1
3	0.26	0.35	...	9	-1
1	0.37	0.23	...	1	1
⋮	⋮	⋮	⋮	⋮	⋮
4	0.41	0.27	...	64	-1

← past data →

Area	Pothers	Pmob	...	NumSMS	Class
1	0.27	0.67	...	36	?
4	0.44	0.22	...	50	?
4	0.31	0.47	...	14	?
⋮	⋮	⋮	⋮	⋮	⋮
2	0.31	0.14	...	49	?

← future data →

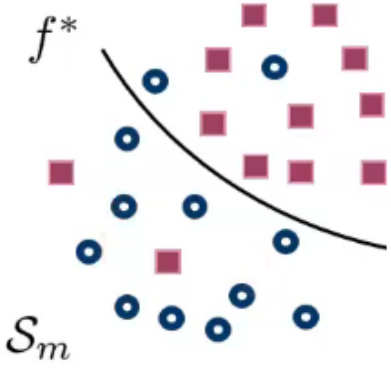


Classification formulation

$\mathcal{S}_m = \{(\mathbf{x}_i, y_i),\ i \in \mathcal{M}\}$: **training set, where** $\mathbf{x}_i \in \Re^n$ **and** $y_i \in \mathcal{D}$

$\mathcal{H}$ **denotes a set of functions** $f(\mathbf{x}) : \Re^n \mapsto \mathcal{D}$

Classification problem: define a hypotheses space $\mathcal{H}$ **and a function** $f^* \in \mathcal{H}$ **which optimally describes the relationship between** $\mathbf{x}_i$ **and** y_i



new patient

$\Rightarrow$ $f^* \in \mathcal{H}$ $\Rightarrow$ $\overline{\mathbf{x}}$ $\Rightarrow$ $f(\overline{\mathbf{x}}) = -1$

healthy !

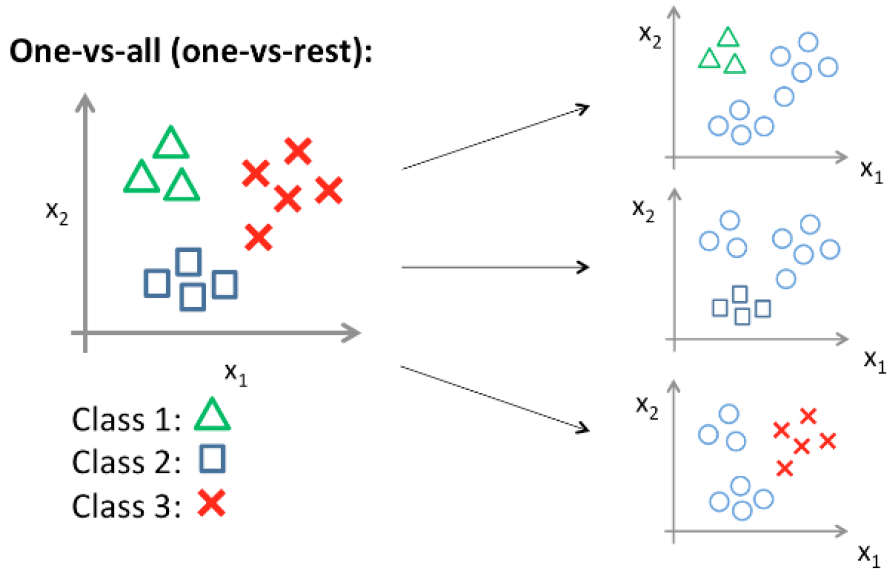
learning prediction

healthy patient (-1)

ill patient (+1)

Multi-class classification

1. **One-vs-Rest** We perform $|H|$ different binary classifications: one for every class.



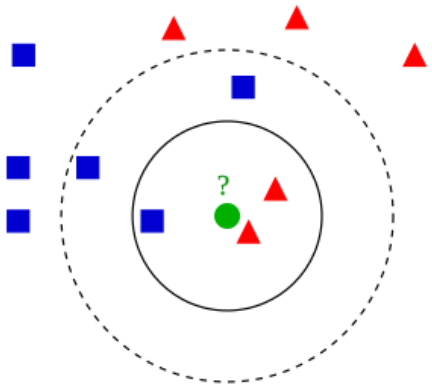
[1]

   We decide based on a majority vote.

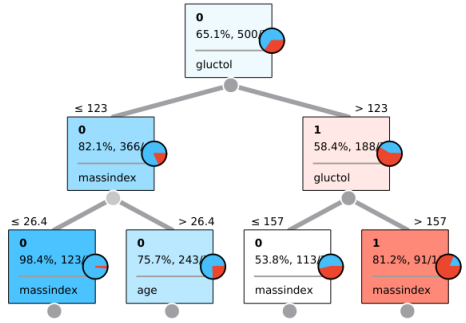
2. **One-vs-One** We perform $|H|(|H-1|)/2$ binary classifications: one for every pair of classes. We decide based on a majority vote.

[1] Image via www.cc.gatech.edu

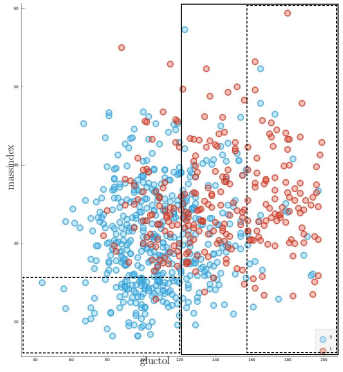
Classification Models

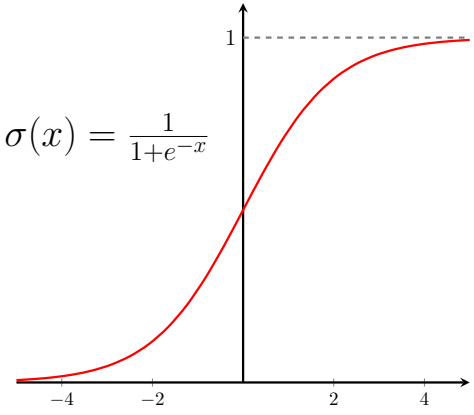


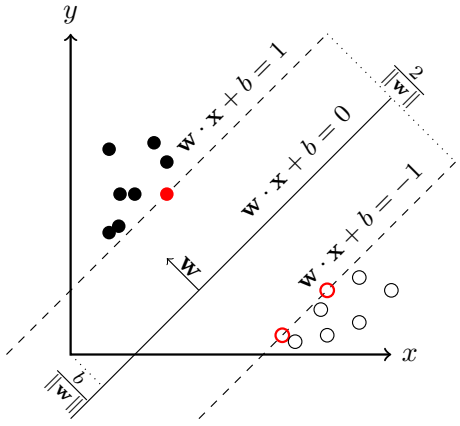
KNN

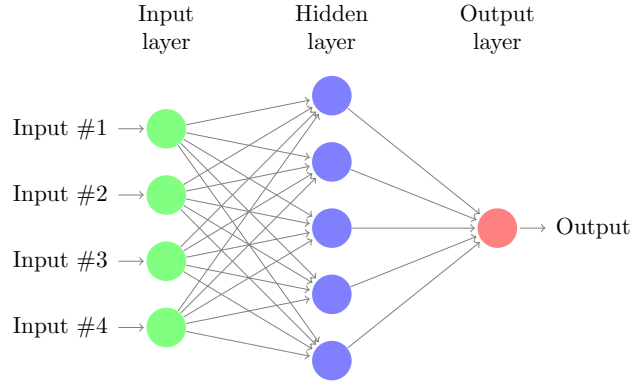


Classification Tree









$$\min_w \underbrace{\frac{1}{2}||w||^2}_{\text{regularization}} + C \underbrace{L(y,x)}_{\text{likelihood}}$$

Logistic Regression

SVM

Neural Network

Classification Models

- Heuristics Methods
  - Nearest Neighbours
  - Classification Trees
- Probabilistic Methods
  - Bayesian Methods
- Regression Methods
  - Logistic regression
- Separation Methods
  - Support vector machine
  - Perceptron
  - Neural Networks

Evaluation Dimensions

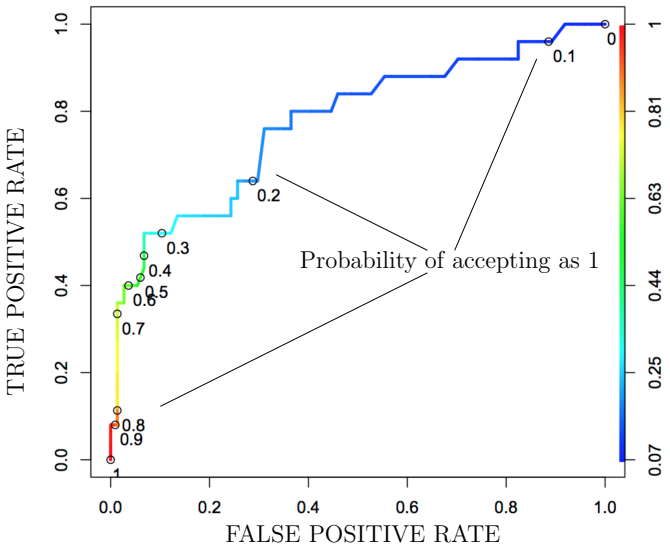
- Prediction accuracy
- Speed
- Robustness
- Scalability
- Intepretability
- Rules effectiveness

Classification - Quality measures - Confusion Matrix

Actual value \ Prediction outcome	0	1
0	True Negative	False Positive
1	False Negative	True Positive

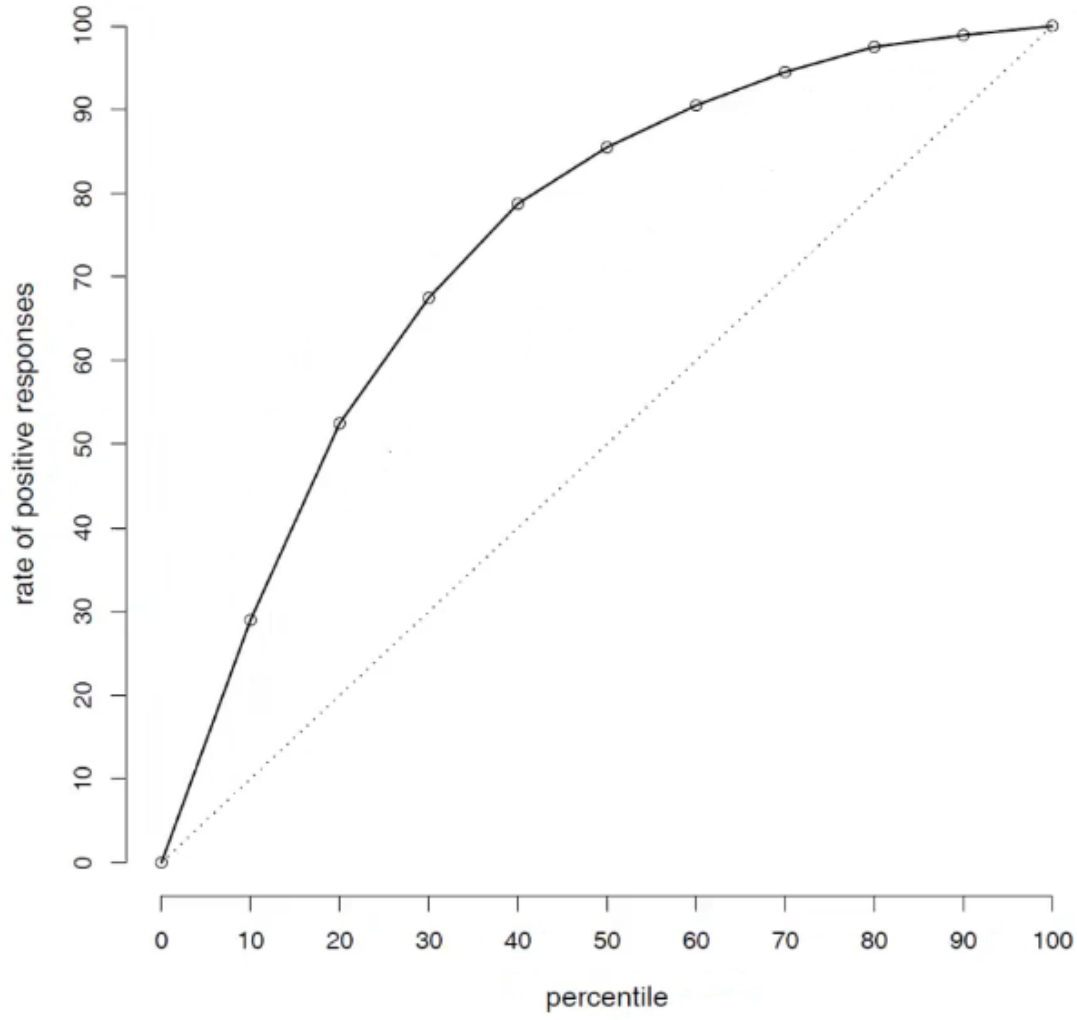
- Precision $= \frac{TP}{TP+FP}$
  "proportion of true positives among **positive predictions**"
- False Positive rate$=\frac{FP}{FP+TN}$
  "proportion of false positives among **actual negatives**"
- Recall (True Positive rate)$=\frac{TP}{FN+TP}$
  "proportion of true positives among **actual positive**"
- Geom. mean$=\sqrt{\text{Precision} \times \text{Recall}}$
- F-score$=\frac{(\beta^2+1)}{\beta^2} \frac{1}{\frac{1}{\text{Precision}}+\frac{1}{\text{Recall}}}$

Classification - Quality measures - ROC curve & AUC

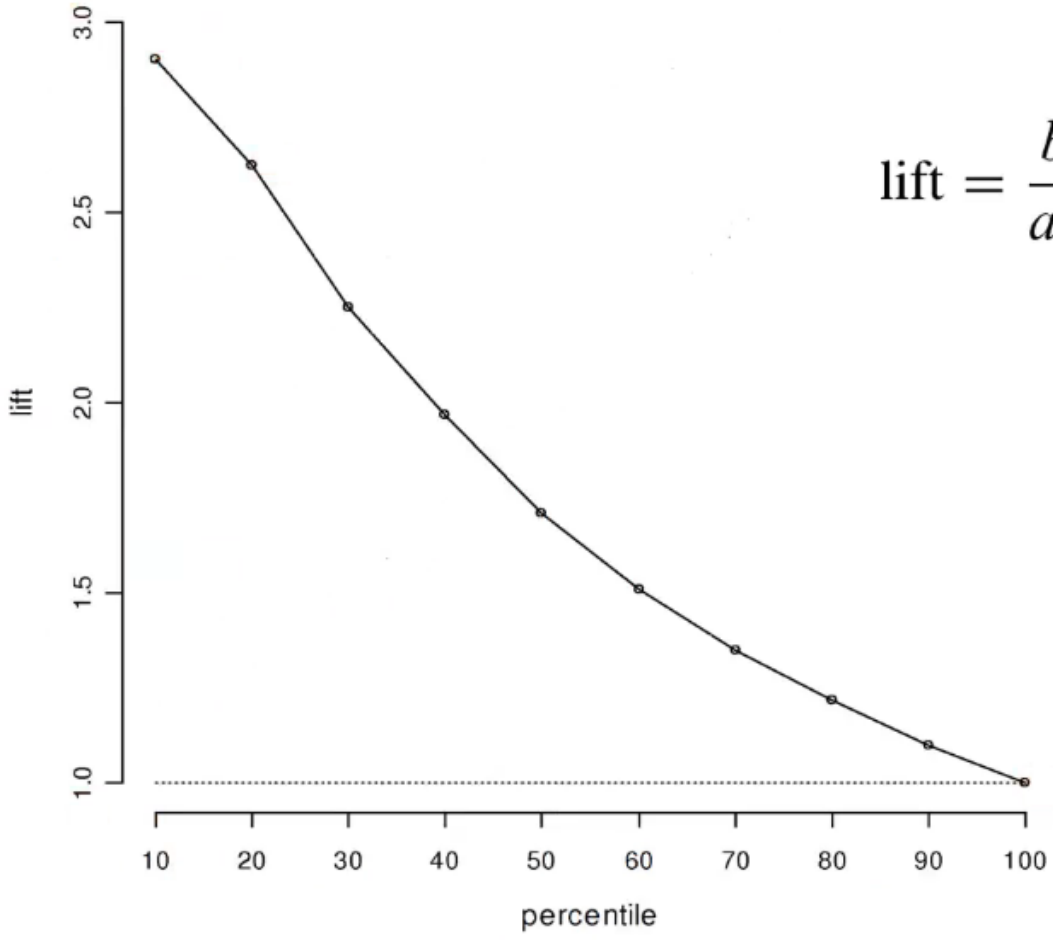


- If we accepting even with small probability then $TPR = FPR = 1$
- If we accepting just with high probability then $TPR = FPR = 0$
- The perfect classificator is the the point $(0, 1)$
- $AUC \in [0.5, 1]$ area under the curve is a quality measure of our algorithm.

Classification - Quality measures - Cumulative (Lift) Gain

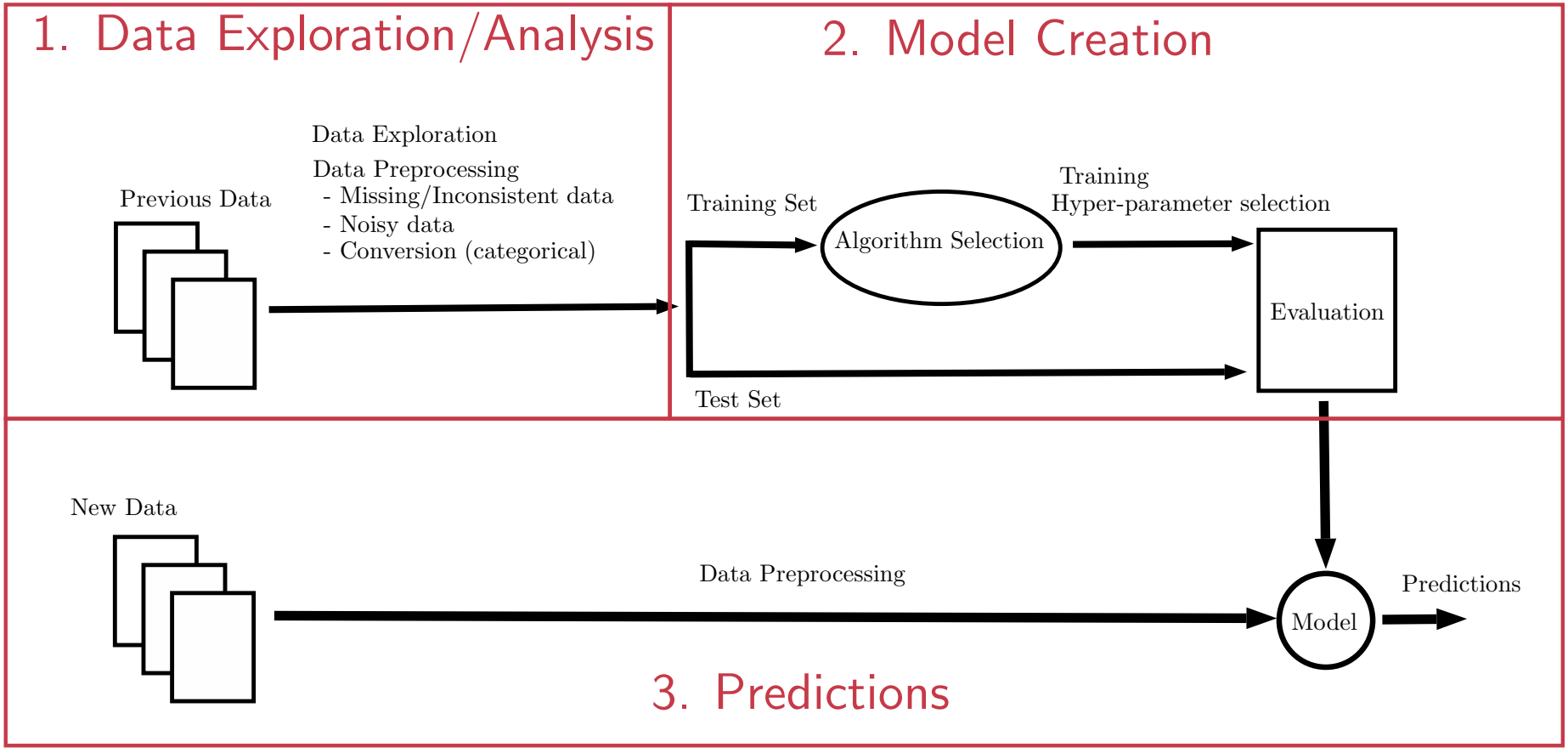


Classification - Quality measures - Lift Chart

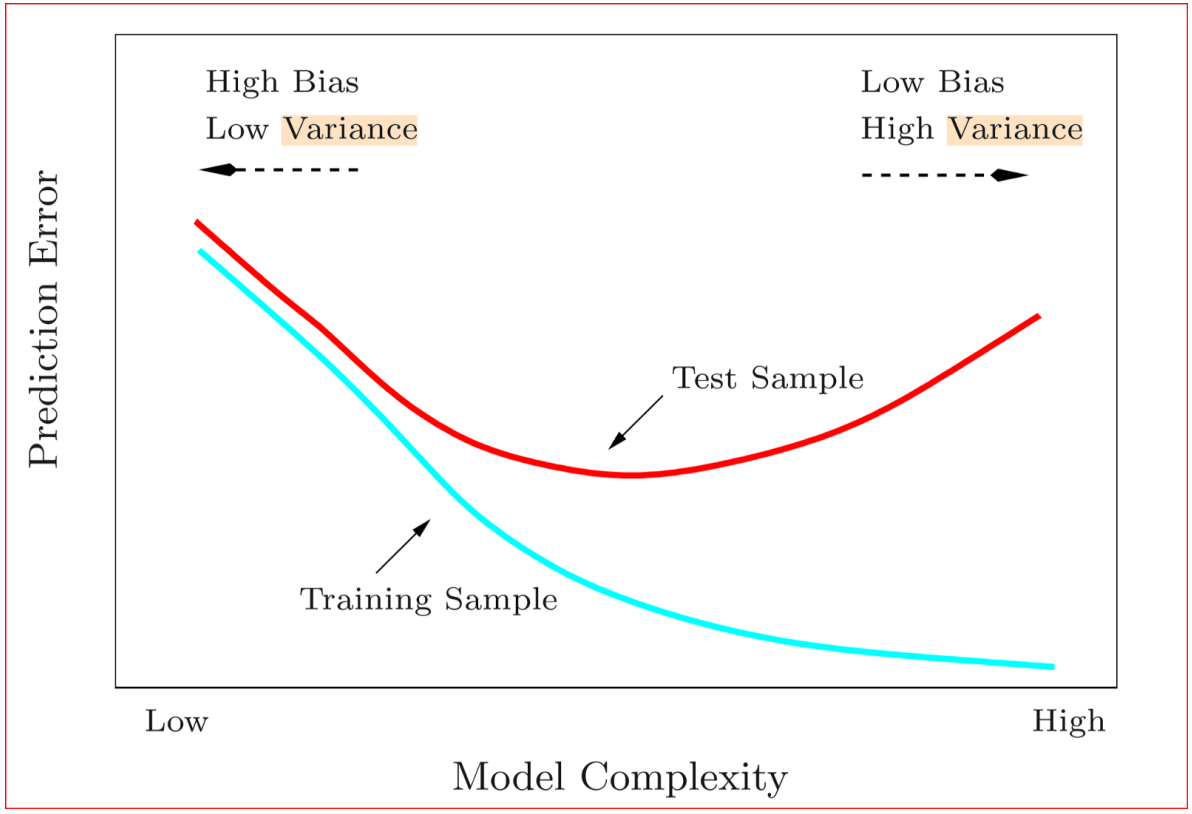


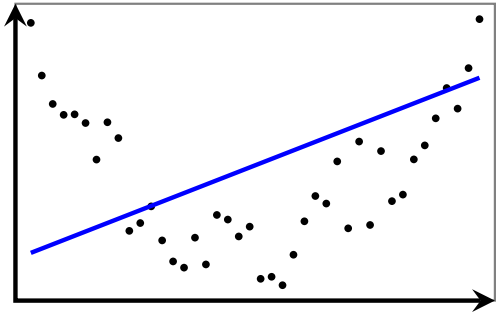
$$\text{lift} = \frac{b/s}{a/m}$$

Workflow

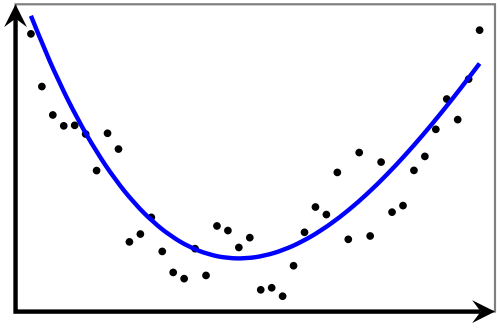

1. Data Exploration/Analysis
2. Model Creation
Data Exploration
Data Preprocessing
- Missing/Inconsistent data
- Noisy data
- Conversion (categorical)
Previous Data
Training Set
Algorithm Selection
Training
Hyper-parameter selection
Evaluation
Test Set
New Data
Data Preprocessing
Model
Predictions
3. Predictions

Under/Over-fitting

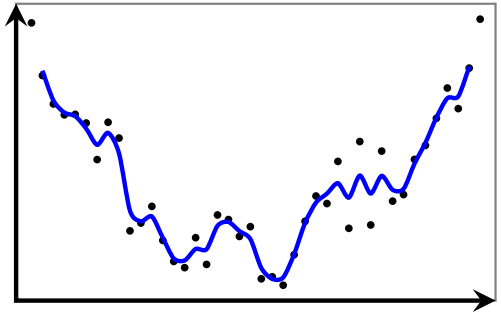




Underfitting

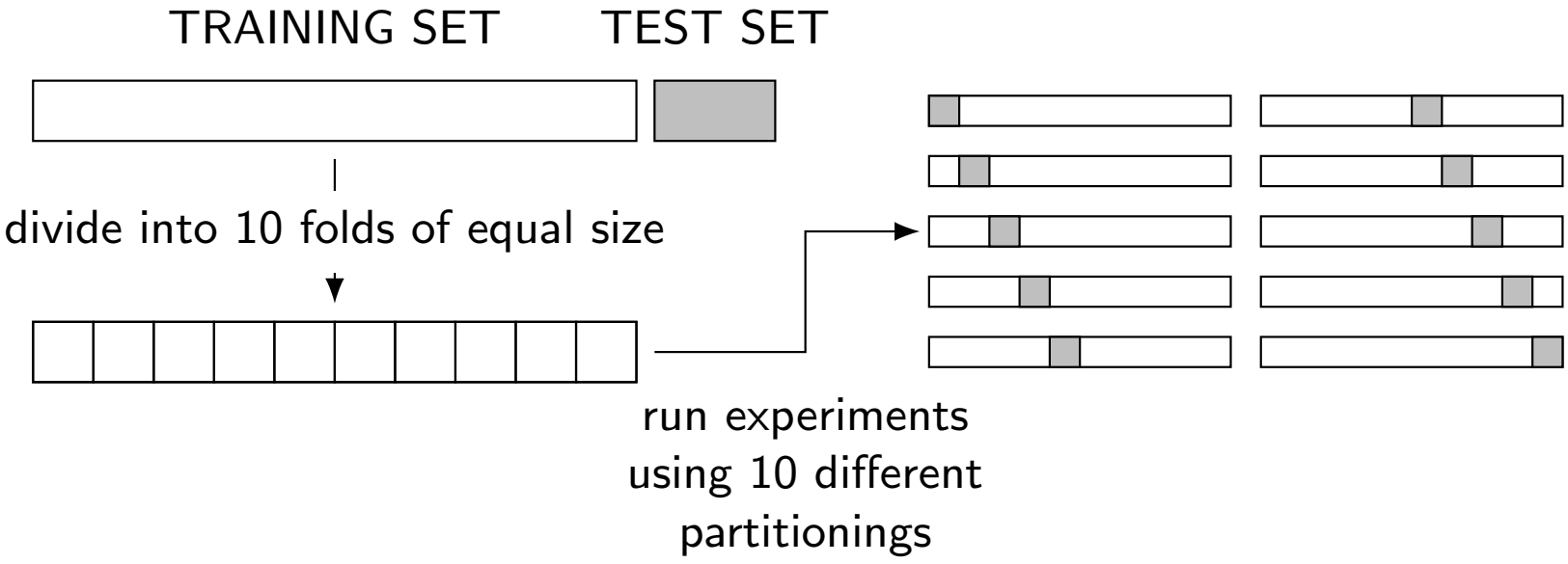


Balance

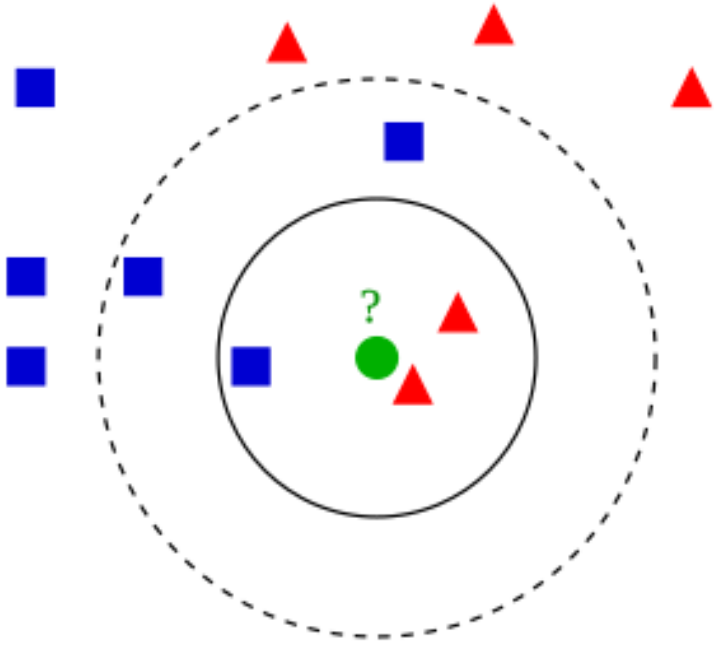


Overfitting

Cross validation



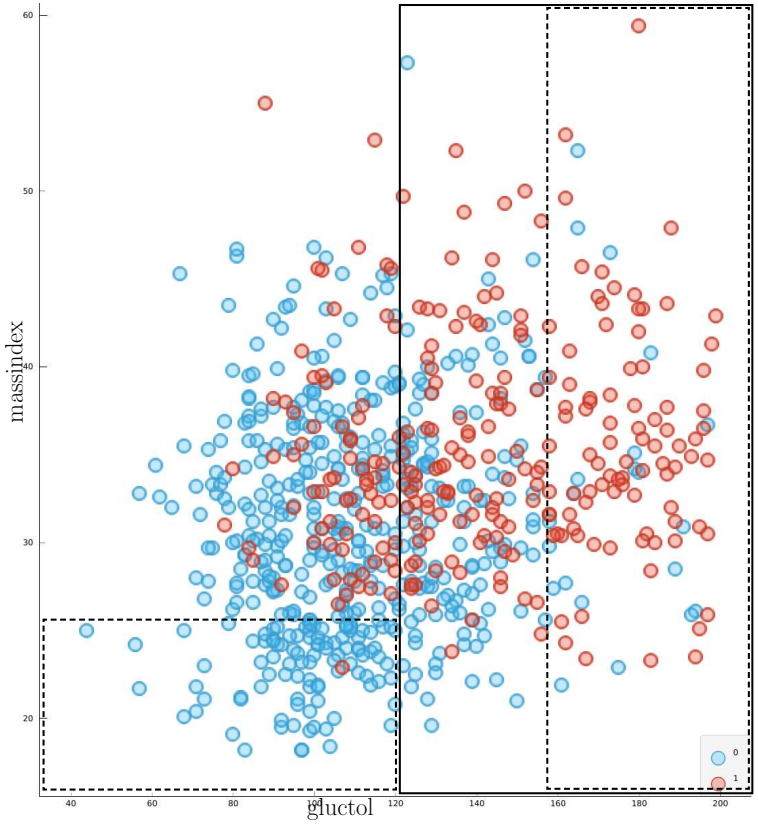
KNN K-nearest Neighbours

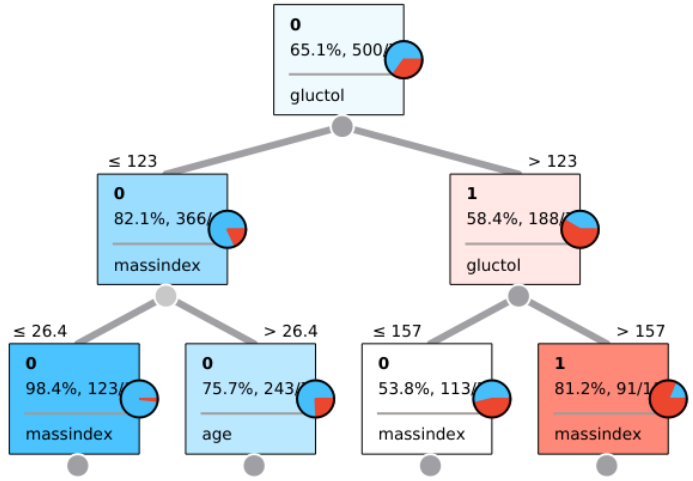


Main Parameters

- k : number of neighbours
- neighbour weights
- distances

Classification tree

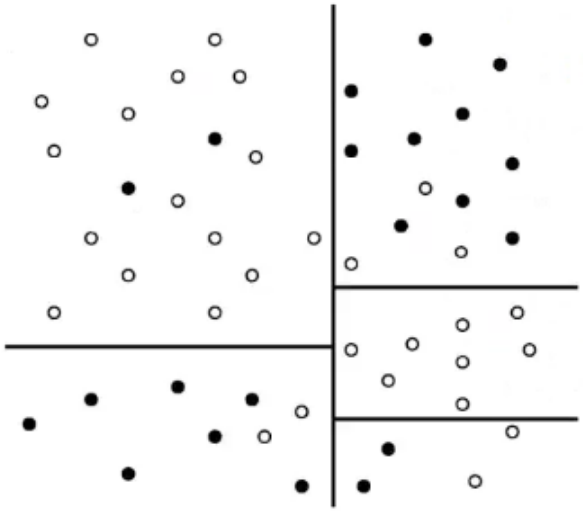




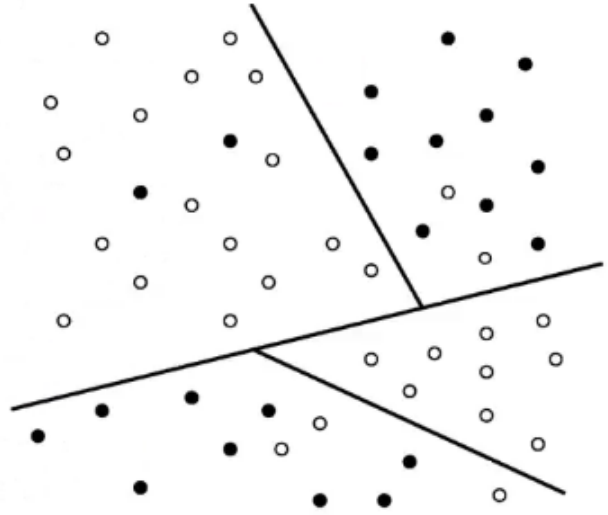
Tree types

- Binary tree (zero/two descendants)
- General trees
- Uni-variate tree ($X_j < b$)
- Multi-variate tree ($\sum_{j=1}^{n} w_j x_j = b$)

Classification tree

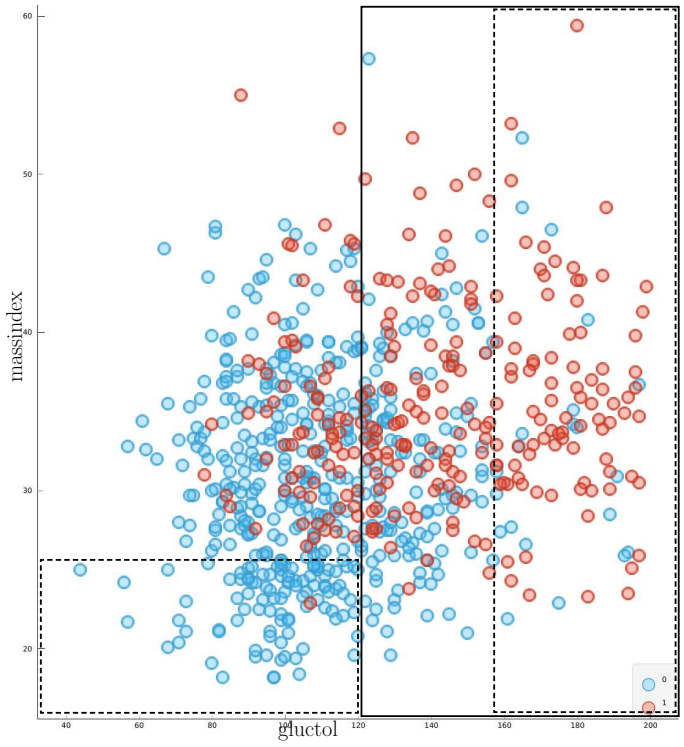


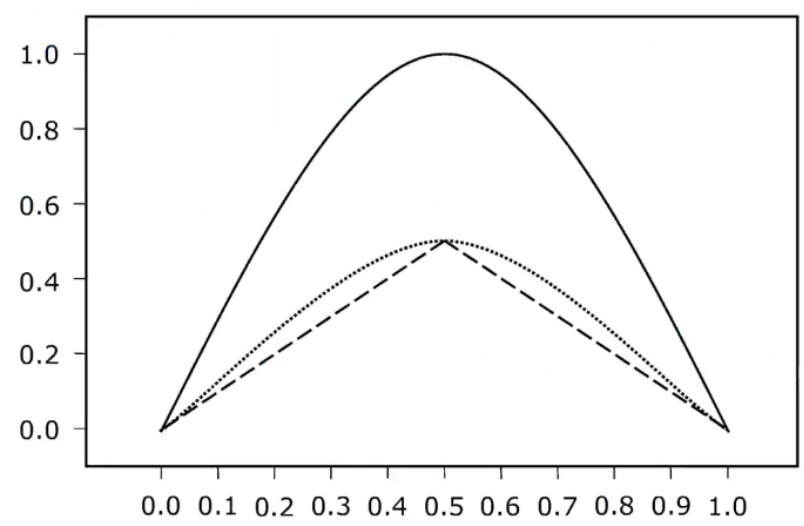
classification by an
axis parallel tree



classification by an
oblique tree

Classification tree





Split criteria

- **Gini index:** $1 - \sum_{h=1}^{H} f_h^2$
- **Entropy index:** $-\sum_{h=1}^{H} f_h \log_2 f_h$
- **Miss-classification index:** $1 - \max_h f_h$

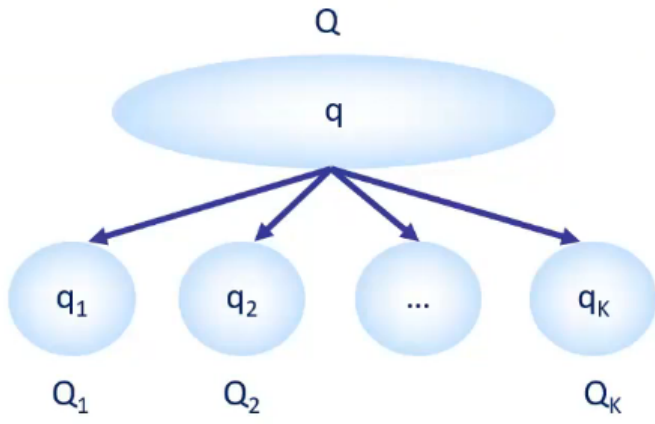
Classification tree

- **Impurity of a splitting rule**

$$I(q_1, q_2, \ldots, q_K) = \sum_{k=1}^{K} \frac{Q_k}{Q} I(q_k).$$

- **At each node select the rule minimizing the impurity or, equivalently, maximizing the information gain**

$$\begin{aligned}\Delta(q, q_1, q_2, \ldots, q_K) &= I(q) - I(q_1, q_2, \ldots, q_K) \\ &= I(q) - \sum_{k=1}^{K} \frac{Q_k}{Q} I(q_k).\end{aligned}$$



Naive Bayesian Classifier

- Bayes Theorem

$$P(y|\mathbf{x}) = \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})} = \frac{P(\mathbf{x}|y)\,P(y)}{\sum_{l=1}^{H} P(\mathbf{x}|y)\,P(y)}$$

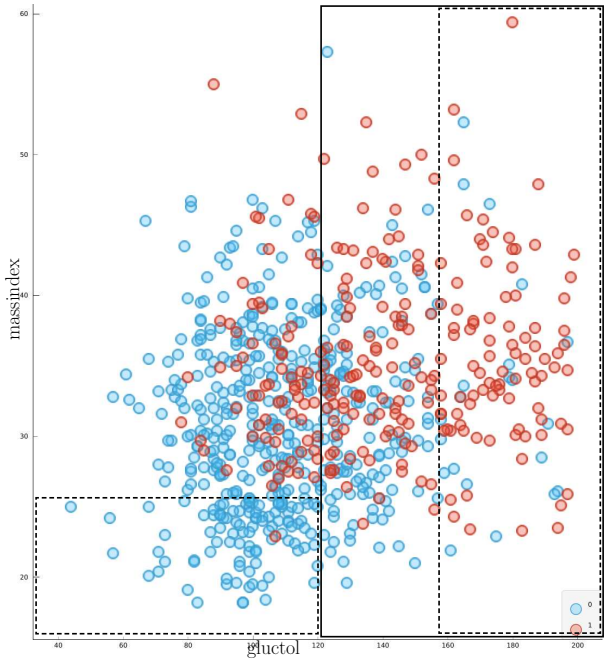
- Maximum a posteriori hypothesis

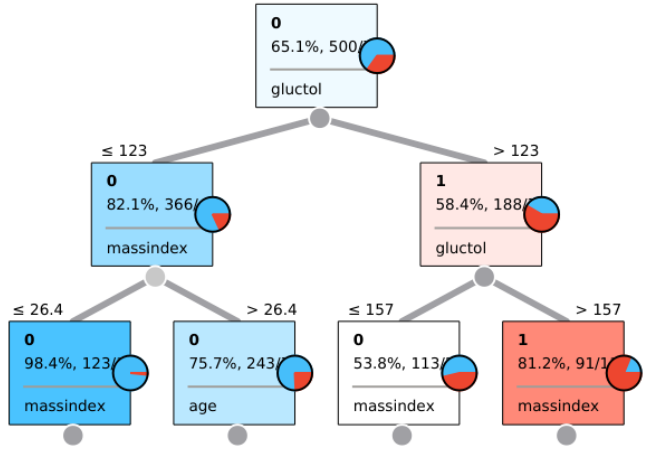
$$y_{MAP} = \arg\max_{y \in \mathcal{H}} P(y|\mathbf{x}) = \arg\max \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})}$$

- Independence (Naive)

$$P(\mathbf{x}|y) = P(x_1|y) \times P(x_2|y) \times \cdots \times P(x_n|y) = \prod_{j=1}^{n} P(x_j|y)$$

Decision tree





Main Parameters

- impurity measure: "gini", "entropy"
- max_depth
- min_samples_split: minimum number of samples to split an internal node
- min_sample_leaf: minimum number of samples required to be at a leaf node

Naive Bayesian Classifier

- Categorical/discrete attributes

$$P(x_j|y) = P(x_j = r_{jk}|y = v_h)$$

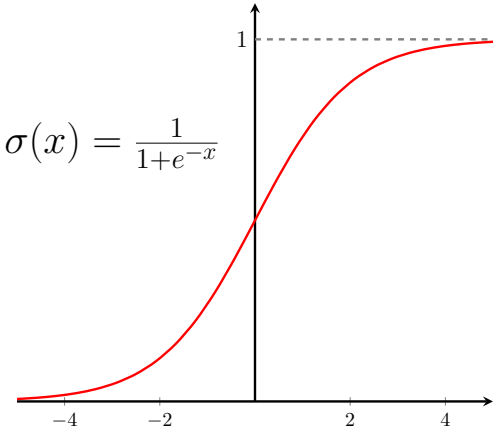
empirical frequency of the observed value on the class v_h

- Numerical attribute

$$P(x_j|y) = \frac{1}{\sqrt{2\pi\sigma_{jh}}} e^{-\frac{(x_j-\mu_{jh})^2}{2\sigma_{jh}^2}}$$

assuming Gaussian density with empirical parameters

Logistic regression



$$\log \frac{P(y=1|x)}{P(y=0|x)} = w_0 + w_1 x_1 + \cdots + w_n x_n$$
$$= w^\top x$$

$$P(y=0|x) = \frac{1}{1+e^{w^\top x}}, P(y=1|x) = \frac{1}{1+e^{-w^\top x}}$$

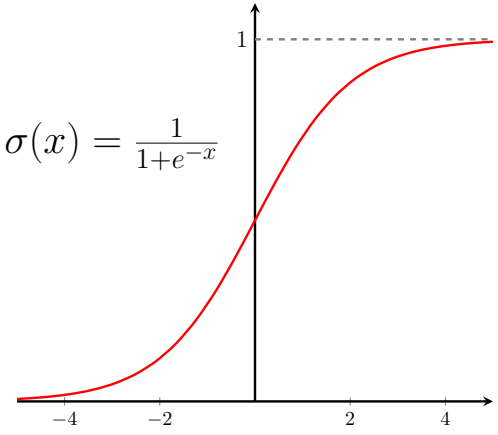
$$P(y|x) = (1+\exp(-y\,(w^\top x)))^{-1} \quad (y = \pm 1)$$

Likelihood

$$L(w \mid y; x) = \Pr(Y \mid X; w)$$
$$= \prod_i \Pr(y_i \mid x_i; w)$$

$$\max \log L(\theta \mid y; x) = \max \sum_{i=1}^{n} \log \Pr(y_i \mid x_i; w)$$
$$= \max - \sum_{i=1}^{n} \log(1 + \exp(-y\,(w^\top x))$$

Logistic regression



$$\log \frac{P(y=1|x)}{P(y=0|x)} = w_0 + w_1 x_1 + \cdots + w_n x_n$$
$$= w^\top x$$

$$P(y=0|x) = \frac{1}{1+e^{wx}}$$

$$\min_w \underbrace{\frac{1}{2}||w||^2}_{\text{regularization}} + C \sum_{i=1}^{n} \log(1 + \exp(-y_i (w^T x_i)))$$

Main Parameters

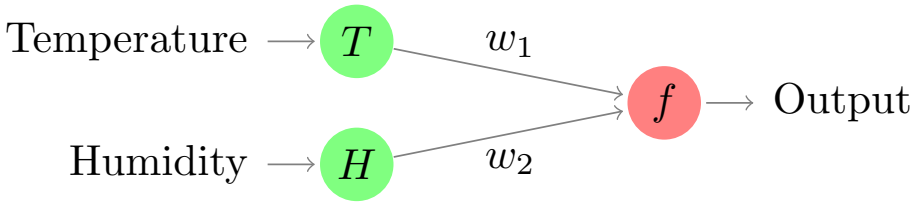
- C: Inverse of regularization strength
- Resolution algorithm parameters:
  - solver: lbfgs, newton-cg, liblinear, sag, saga.
  - tol: Tolerance for stopping criteria.
  - max_iter: max. number of iterations
  - n_jobs: Number of CPU cores

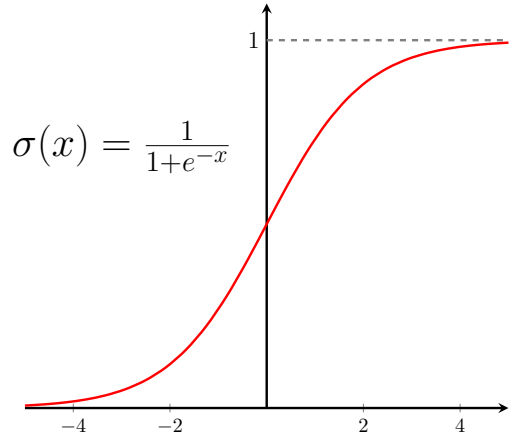
Multi-Layer Perceptron - small example

Temp. [C]	20	31	15	18	21
Humidity [%]	40	36	23	45	30
Prob. Rain	0.70	0.52	0.55	0.73	0.60

Multi-Layer Perceptron - small example

Temp. [C]	20	31	15	18	21
Humidity [%]	40	36	23	45	30
Prob. Rain	0.70	0.52	0.55	0.73	0.60





$$f(T, H, w_1, w_2) = \underbrace{\sigma}_{\text{activation function}}(w_1 \cdot T + w_2 \cdot H)$$

$$\max_{w_1, w_2} \sum_{i=1}^{5} [P_i(\text{rain}) - f(T_i, H_i, w_1, w_2)]^2$$

For a classification problem we can use the Likelihood as cost function.

Multi-Layer Perceptron - small example

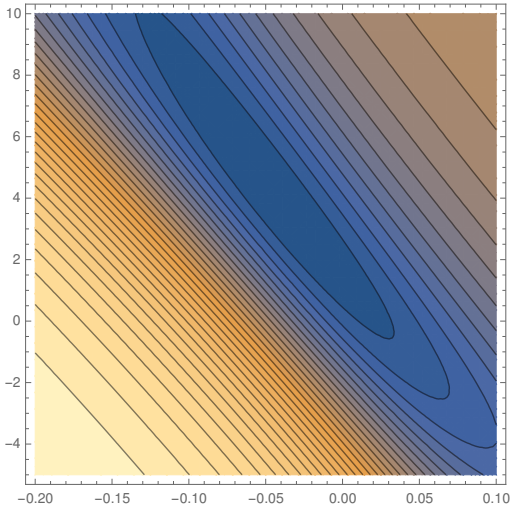
$$\max_{w_1,w_2} \sum_{i=1}^{5} [P_i(\text{rain}) - f(T_i, H_i, w_1, w_2)]^2$$

$$= \max \left[0.7 - 1/(1 + e^{-(w_1 \cdot 20 + w_2 \cdot 0.4)})\right]^2 + \left[0.52 - 1/(1 + e^{-(31 \cdot w_1 + w_2 \cdot 0.36)})\right]^2 +$$

Multi-Layer Perceptron - small example

$$\max_{w_1,w_2} \sum_{i=1}^{5} [P_i(\text{rain}) - f(T_i, H_i, w_1, w_2)]^2$$

$$= \max \left[0.7 - 1/(1 + e^{-(w_1 \cdot 20 + w_2 \cdot 0.4)})\right]^2 + \left[0.52 - 1/(1 + e^{-(31 \cdot w_1 + w_2 \cdot 0.36)})\right]^2 +$$

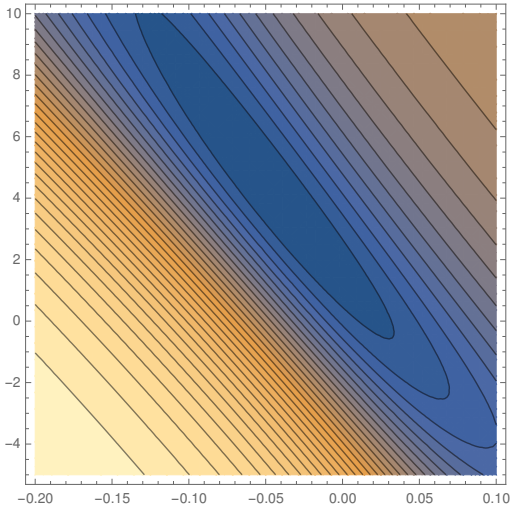


$$(w_1^*, w_2^*) = (-0.044, 4.147)$$

Multi-Layer Perceptron - small example

$$\max_{w_1,w_2} \sum_{i=1}^{5} [P_i(\text{rain}) - f(T_i, H_i, w_1, w_2)]^2$$

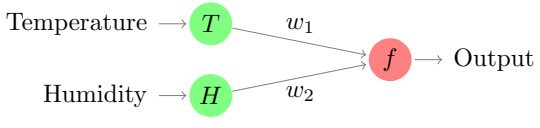
$$= \max \left[0.7 - 1/(1 + e^{-(w_1 \cdot 20 + w_2 \cdot 0.4)})\right]^2 + \left[0.52 - 1/(1 + e^{-(31 \cdot w_1 + w_2 \cdot 0.36)})\right]^2 +$$

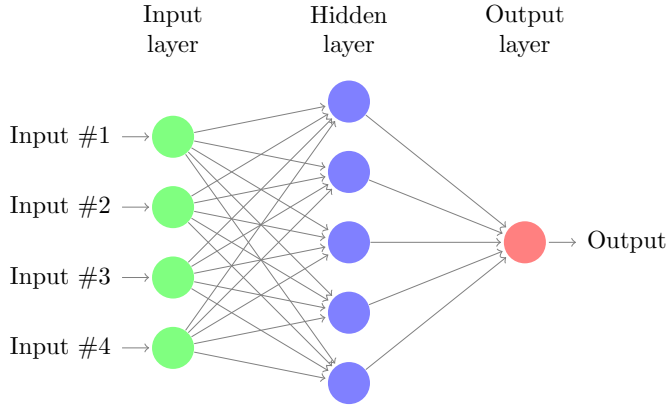


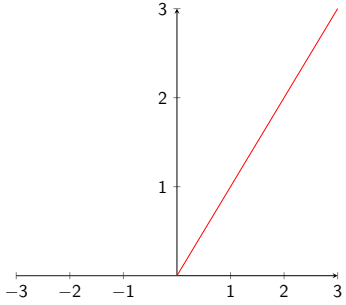
$$(w_1^*, w_2^*) = (-0.044, 4.147)$$

Temp. [C]	20	31	15	18
Humidity [%]	40	36	23	45
Prob. Rain	0.70	0.52	0.55	0.73
Predicted	0.70	0.56	0.58	0.75
Error	**0.0**	**-0.04**	**-0.03**	**-0.02**

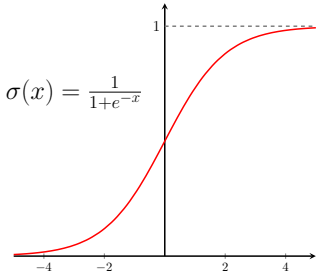
Multi-Layer Perceptron







ReLU

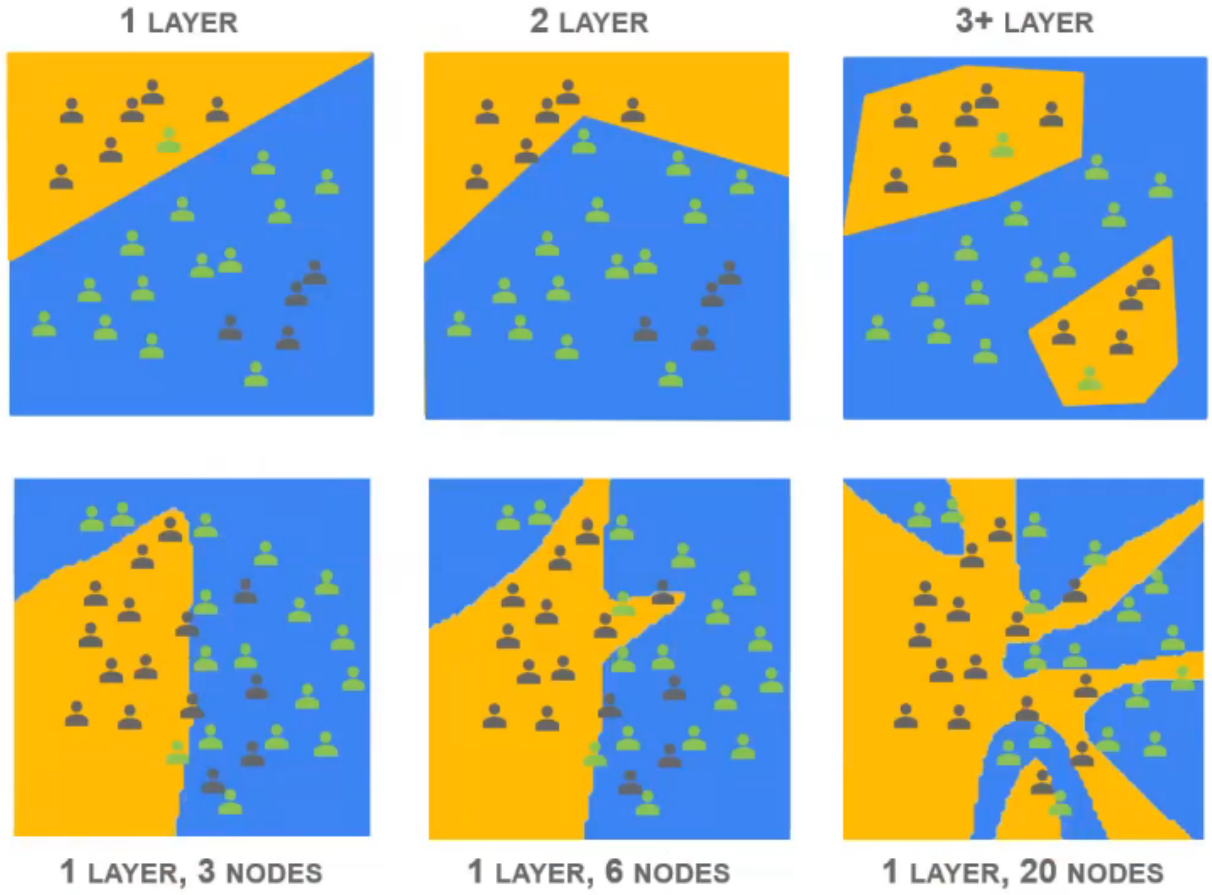


Sigmoid

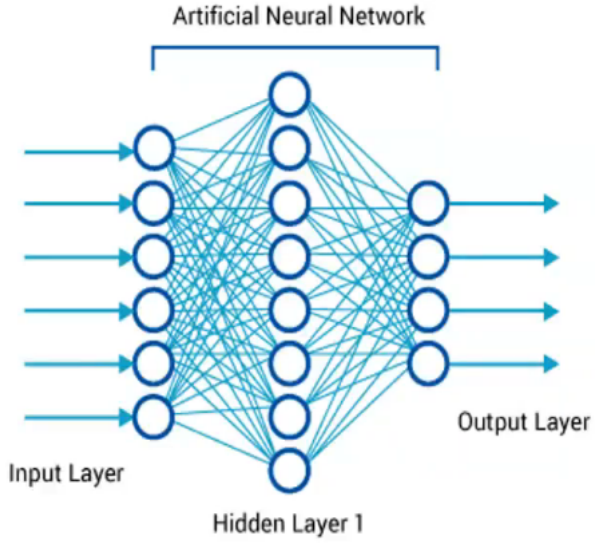
Main Parameters

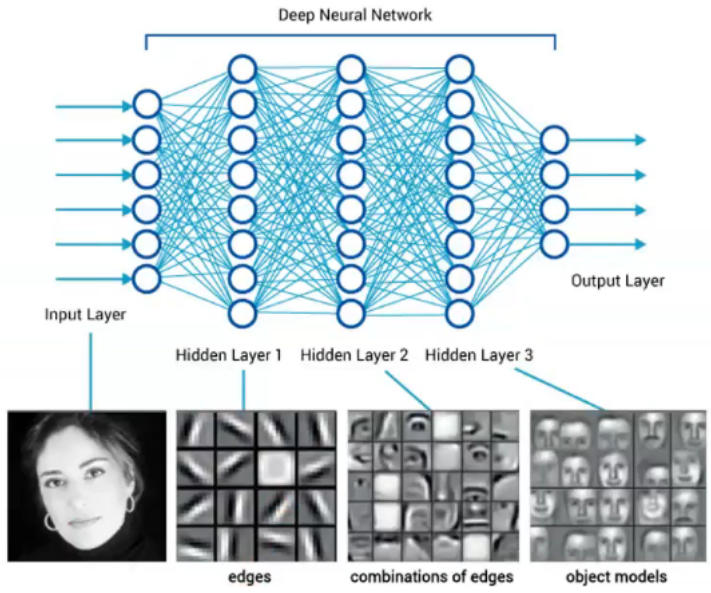
- hidden_layer_sizes: $(n_1, n_2, \ldots, n_L)$
- activation: identity, logistic, tanh, relu
- alpha regularization term parameter
- Resolution algorithm parameters: solver, tol, batch_size, learning_rate, max_iter.

Neural Network



Deep Learning





Classification and statistical learning theory

Given an hypothesis space $\mathcal{F}$, a predictive function $f \in \mathcal{F}$, and a loss function $V(y, f(\mathbf{x}))$, we define

- the **empirical risk** on a train set $\mathcal{T}$ as

$$R_{emp}(f) = \frac{1}{m} \sum_{i=1}^{m} V(y_i, f(\mathbf{x_i}))$$

- and the **expected risk** as

$$R(f) = \frac{1}{2} \int V(y, f(\mathbf{x}))\, dP(\mathbf{x}, y)$$

Overfitting arises when the difference between these errors is large.

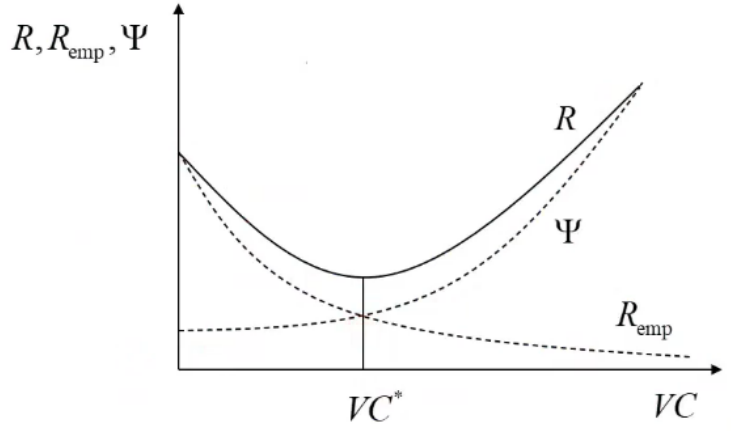
VC-dimension

The Vapnik–Chervonenkis (VC) dimension is the maximum number of points that can be correctly classified by classifiers in $\mathcal{F}$. We can prove that

$$R(f) - R_{emp}(f) \leq \sqrt{\frac{1}{t}(\gamma \log(2t/\gamma) - \log(\eta/4) + 1)} = \Psi(t, \gamma, \eta)$$

with probability $1 - \eta$, where

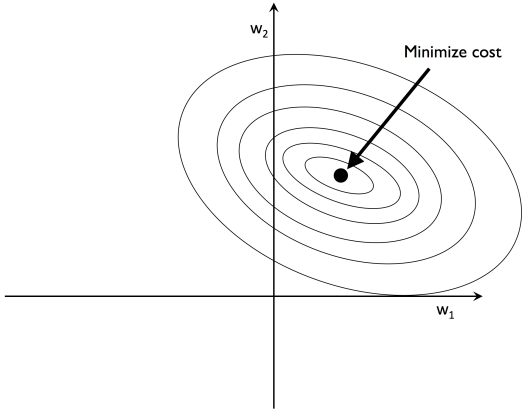
- t is the number of training points
- γ is the VC-dimension

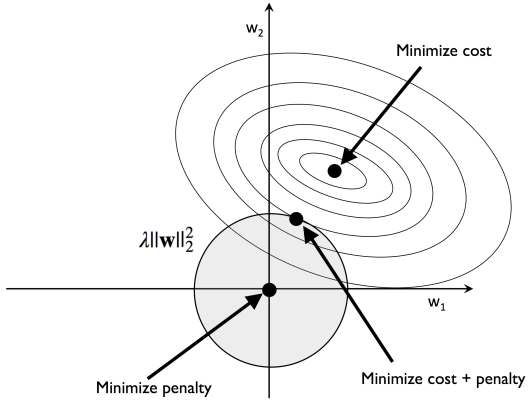


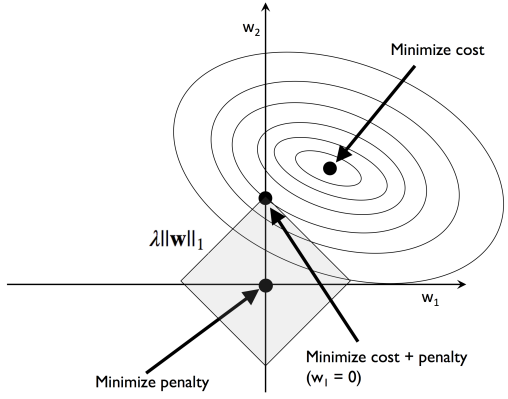
Regularization

Structural risk minimization (SRM)

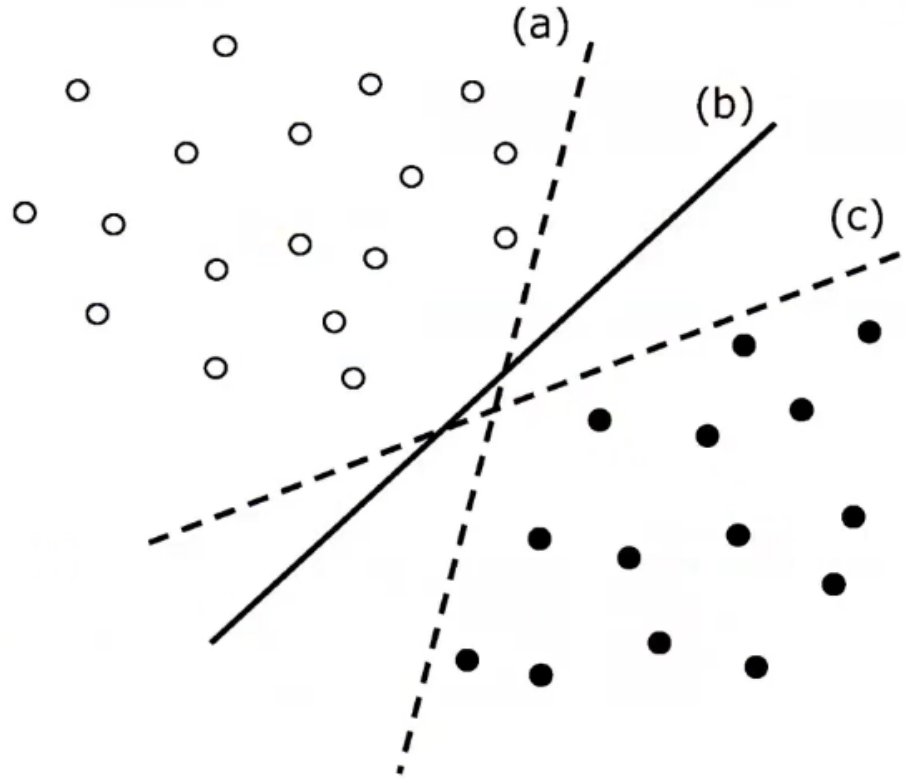
$$\hat{R}(f) = \frac{1}{t}\sum_{i=1}^{t} V(y_i, f(\mathbf{x_i})) + \lambda \| f \|_K^2$$



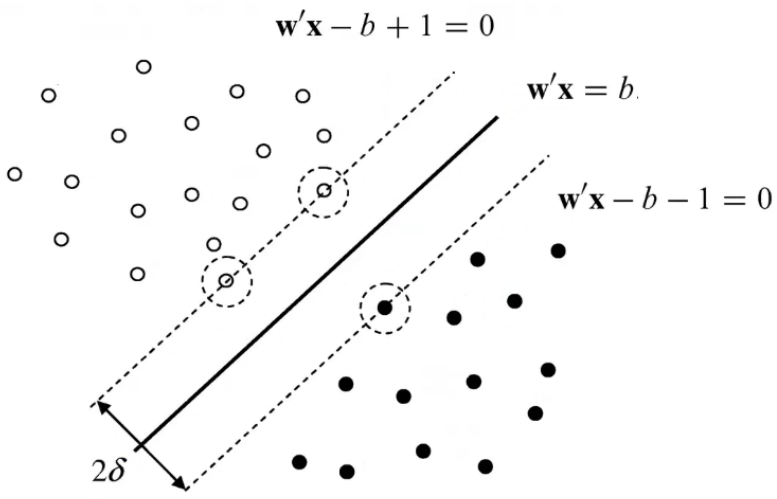




Support Vector Machine - linearly separable



SVM - linearly separable

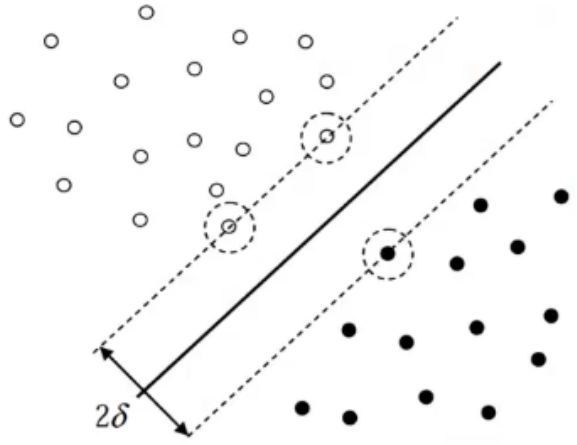


$$\delta = \frac{2}{\|w\|}, \qquad \|w\| = \sqrt{\sum_{j \in \mathcal{N}} w_j^2}$$

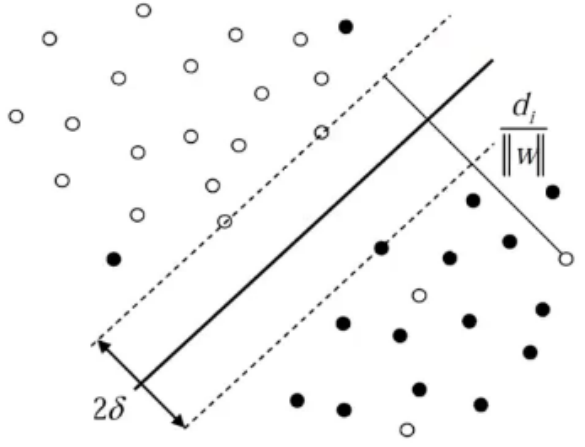
$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|_2^2$$
$$\text{s. t.} \quad y_i(\mathbf{w}'\mathbf{x}_i - b) \geq 1 \quad i \in \mathcal{M}$$

SVM - general case

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|_2^2$$
$$\text{s.t.} \quad y_i(\mathbf{w}'\mathbf{x}_i - b) \geq 1 \quad i \in \mathcal{M}$$



$$\min_{\mathbf{w},b,d} \quad \frac{1}{2}\|\mathbf{w}\|_2^2 + \lambda \sum_{i=1}^{m} d_i$$
$$\text{s.t.} \quad y_i(\mathbf{w}'\mathbf{x}_i - b) \geq 1 - d_i \quad i \in \mathcal{M}$$
$$d_i \geq 0 \quad i \in \mathcal{M}$$



SVM - formulation

$$L(\mathbf{w}, b, \mathbf{d}, \boldsymbol{\alpha}\, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|^2 + \lambda \sum_{i=1}^{m} d_i$$
$$- \sum_{i=1}^{m} \alpha_i [y_i(\mathbf{w}'\mathbf{x}_i - b) - 1 - d_i] - \sum_{i=1}^{m} \mu_i d_i$$

$$\frac{\partial L(\mathbf{w}, b, \mathbf{d}, \boldsymbol{\alpha}\, \boldsymbol{\mu})}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i = \mathbf{0},$$
$$\frac{\partial L(\mathbf{w}, b, \mathbf{d}, \boldsymbol{\alpha}\, \boldsymbol{\mu})}{\partial b} = \lambda - \alpha_i - \mu_i = 0,$$
$$\frac{\partial L(\mathbf{w}, b, \mathbf{d}, \boldsymbol{\alpha}\, \boldsymbol{\mu})}{\partial \mathbf{d}} = \sum_{i=1}^{m} \alpha_i y_i = 0,$$

SVM - solution

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i,$$

$$\lambda = \alpha_i + \mu_i,$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0.$$

$$L(\mathbf{w}, b, \mathbf{d}, \boldsymbol{\alpha}\, \boldsymbol{\mu}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{k=1}^{m} y_i y_k \alpha_i \alpha_k \mathbf{x}_i' \mathbf{x}_k$$

SVM - solution

Lagrangean function

$$L(\mathbf{w}, b, \mathbf{d}, \boldsymbol{\alpha}\, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|^2 + \lambda \sum_{i=1}^{m} d_i - \sum_{i=1}^{m} \alpha_i [y_i(\mathbf{w}'\mathbf{x}_i - b) - 1 + d_i] - \sum_{i=1}^{m} \mu_i d_i$$

Representation form

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i$$

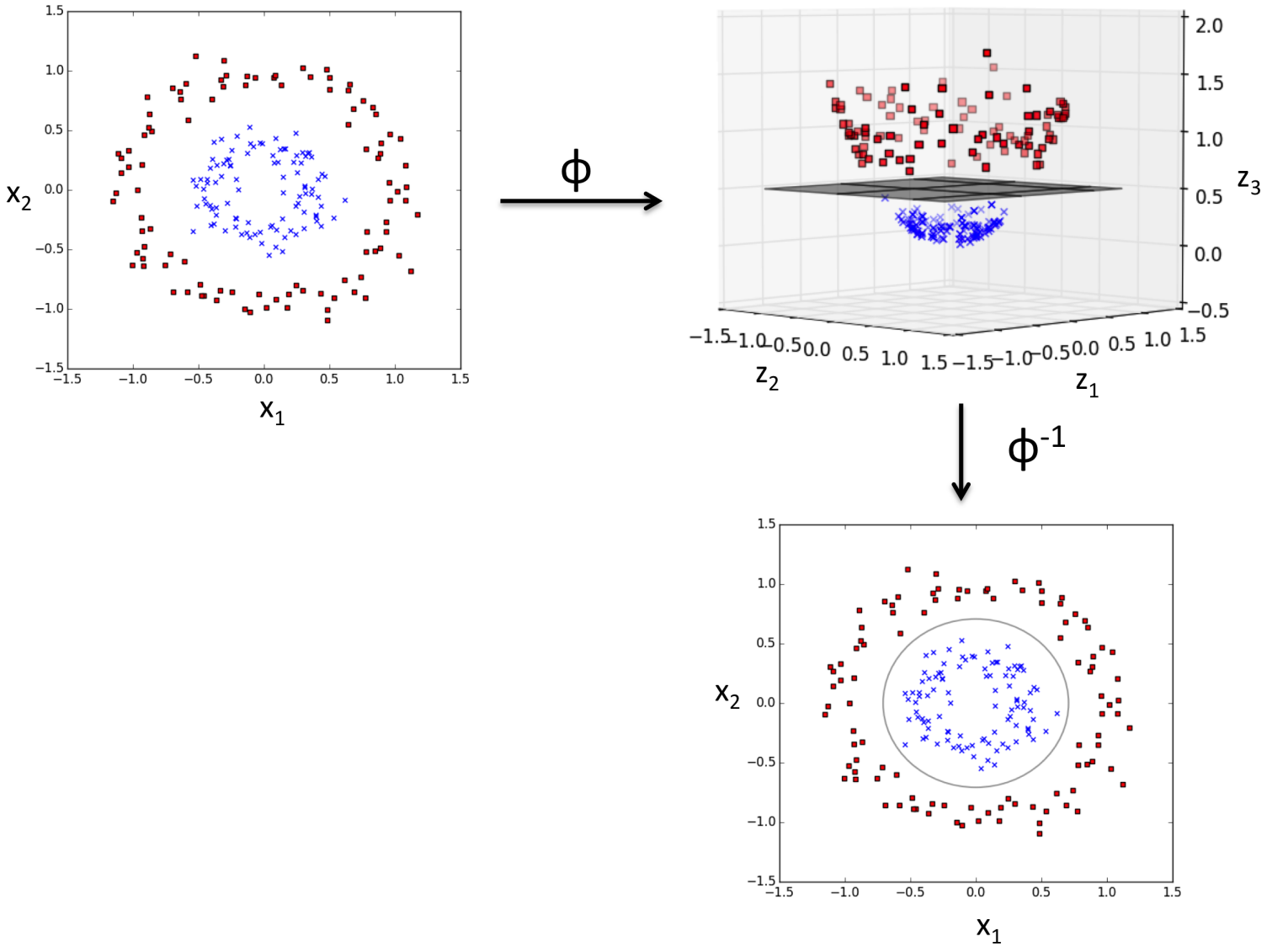
Lagrangean function expressed as

$$L(\mathbf{w}, b, \mathbf{d}, \boldsymbol{\alpha}\, \boldsymbol{\mu}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{h=1}^{m} y_i y_h \alpha_i \alpha_h \mathbf{x}_i' \mathbf{x}_h$$

Prediction

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w}'\mathbf{x} - b)$$

SVM - kernels



SVM - kernels

Kernel

$$\boldsymbol{\Phi}(\mathbf{x}_i)'\boldsymbol{\Phi}(\mathbf{x}_h) = k(\mathbf{x}_i, \mathbf{x}_h)$$

Lagrangean function

$$L(\mathbf{w}, b, \mathbf{d}, \boldsymbol{\alpha}\,\boldsymbol{\mu}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{h=1}^{m} y_i y_h \alpha_i \alpha_h k(\mathbf{x}_i, \mathbf{x}_h)$$

- **Existence of meaningful kernel functions is guaranteed by Mercer Theorem**
  - **polynomial kernels**

$$k(\mathbf{x}_i, \mathbf{x}_h) = (\mathbf{x}_i' \mathbf{x}_h + 1)^d$$

  - **radial basis kernels**

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

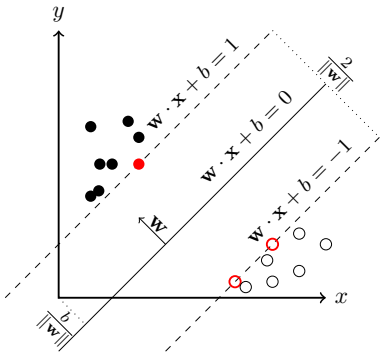
  - **neural networks kernels**

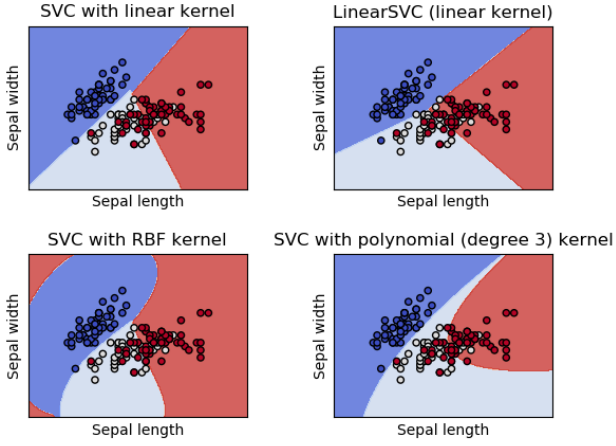
$$k(\mathbf{x}_i, \mathbf{x}_h) = \tanh(\kappa \mathbf{x}_i' \mathbf{x}_h - \delta)$$

Prediction

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i y_i \boldsymbol{\Phi}(\mathbf{x})'\boldsymbol{\Phi}(\mathbf{x}_i) - b\right)$$

SVM - general case





$$\min_{w,b,d} \quad \frac{1}{2}||w||^2 + C\sum_{i=1}^{m} d_i$$

$$\text{subject to } y_i(w^T \underbrace{\phi(x_i)}_{\text{kernel}} - b) \geq 1 - d_i,$$

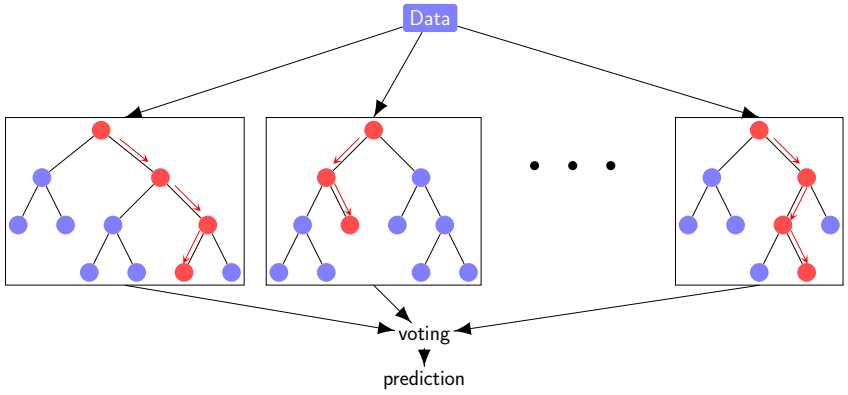
$$d_i \geq 0$$

Main Parameters

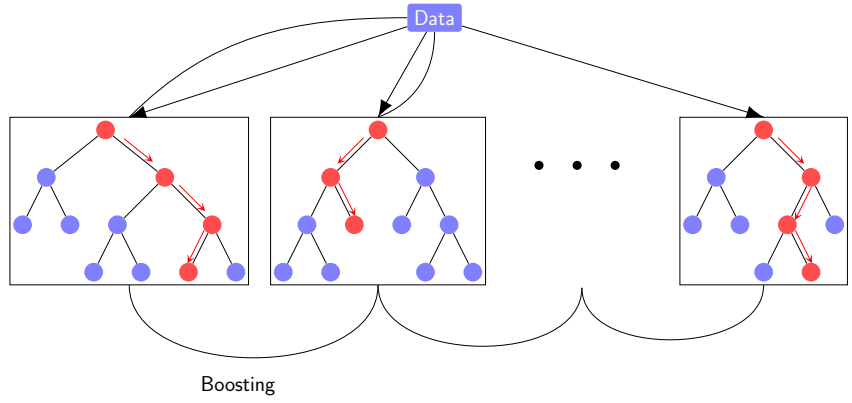
- C: Inverse of regularization strength
- kernel:
  - linear: $x'x$
  - poly: $(\gamma x'x + r)^d$
  - rbf: $exp(-\gamma||x - x'||^2)$
  - sigmoid: $\tanh(\gamma x'x + r)$
- degree(d), gamma(γ), coef0(r)
- Resolution algorithm parameters

Ensemble Methods

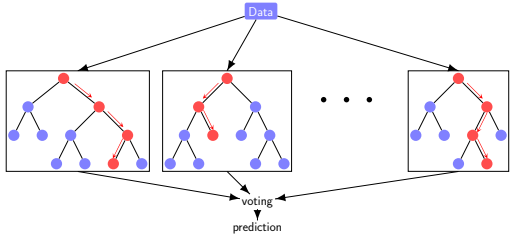
Bagging



Boosting



Random Forest

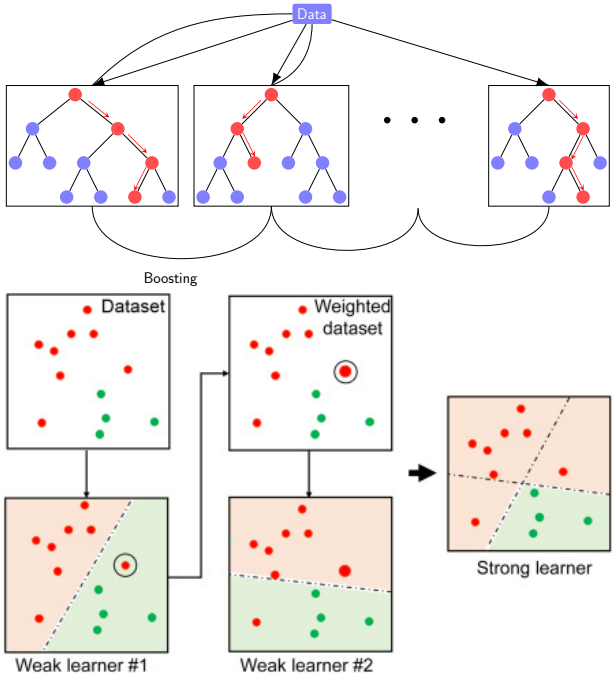


1. Create different (simple) tree models (stumps)
2. Each model is created with a subset of observation/features ($\sim$2m/3)
3. We combine the prediction of all trees

Main Parameters

- n_estimators: Number of trees
- max_features: Number of features selected for the split
- bootstrap=False: Use all samples
- Tree parameters

Adaboost



1. Assign equal weights to observations $w_i^{(0)} = 1/m$
2. For $k = 1, \ldots, K$
   - Select a sample of observations based on the weights.
   - Create the k-th weak learner and compute predictions $x^{(k)}$
   - Compute the model **weighted** error and assign its coefficient:

$$\alpha^{(k)} = \lambda \times \log((1 - error)/error)$$

   - Update sample weights:

$$w_i^{(k+1)} \propto w_i^{(k)} \times \exp(-\alpha^{(k)} y_i \hat{x}_i^{(k)})$$

3. Final weighted prediction

Main Parameters

- n_estimators: Number of estimators (K)
- base_estimator: Weak estimator type
- learning_rate: weights of estimator in final decision (λ)

Gradientboost

1. Train a weak learner F_0 and compute predictions $x^{(k)}$
2. For $k = 1, \ldots, K-1$
   - Compute the difference between the target y and the predictions of the current strong learner
   - Train a weak learner that predict this difference f_k
   - $F^{(k+1)} = F^{(k)} + \lambda f_k$