CORPORATE EDUCATION

Machine Learning & Big Data

Supervised Learning

Mauricio Soto - mauricioabel.soto@polimi.it

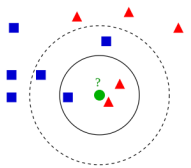
POLITECNICO DI MILANO
GRADUATE SCHOOL
OF BUSINESS

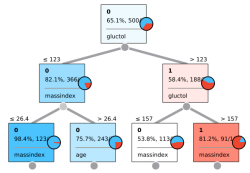
Mauricio Soto

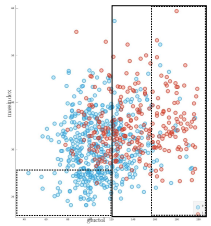
- Researcher at DIG-POLIMI
  - mauricioabel.soto@polimi.it
- Previous
  - Mathematical Engineering - U. of Chile
  - PhD. Computer Science - U. of Paris
  - Researcher - U. of Orléans, U. of Chile, UNIMIB
- Research interests
  - Graph theory
  - Evolutionary networks
  - Data representation (NLP)

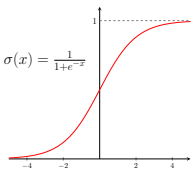
The program

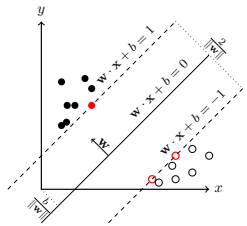
- Supervised Learning
  - Pre-processing
  - Classification
  - Regression
- Day 2
  - Clustering
  - (Introduction to NLP)

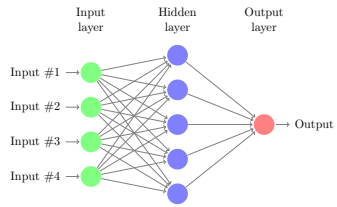








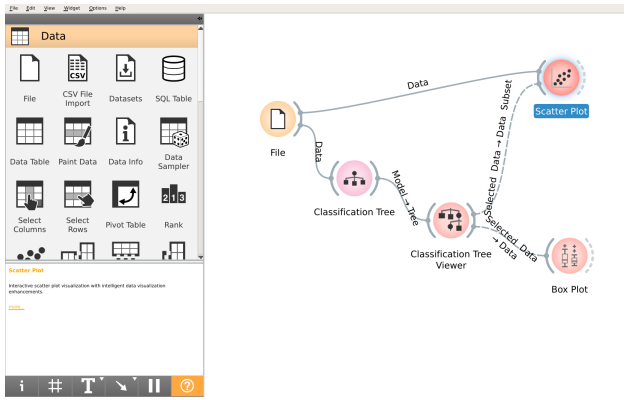




Programming Tools

1. Orange

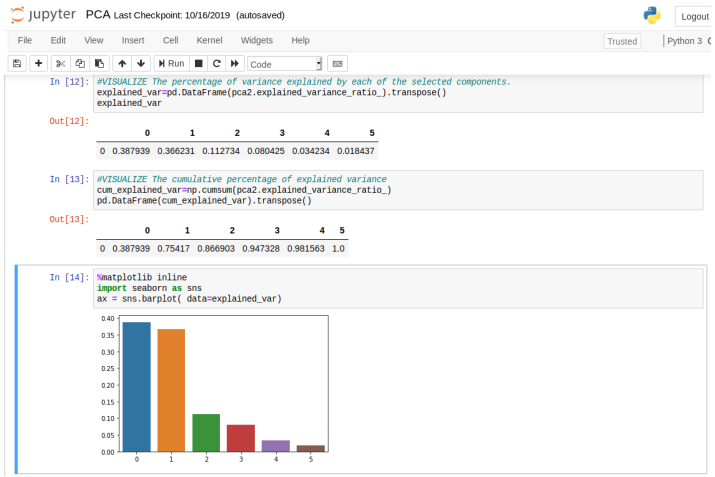
https://orange.biolab.si/



- Intuitive interface
- Fast development

2. Jupyter-Notebook (Anaconda)

https://www.anaconda.com/



- Advanced functions
- Customization

scikit learn

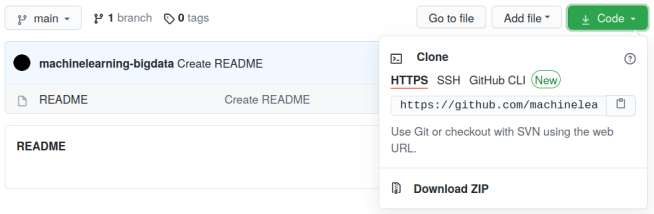
A library featuring various ML algorithms designed to inter-operate with the Python numerical and scientific libraries e.g. NumPy, Pandas.
https://scikit-learn.org/stable/

Setup: What we need

- Python $\geq$3.6
- Jupyter Notebook
- Pandas
- Numpy
- Sklearn
- Matplotlib
- Seaborn

Technical test

1. Go to https://github.com/machinelearning-bigdata/STM
2. Download the code (Download ZIP)



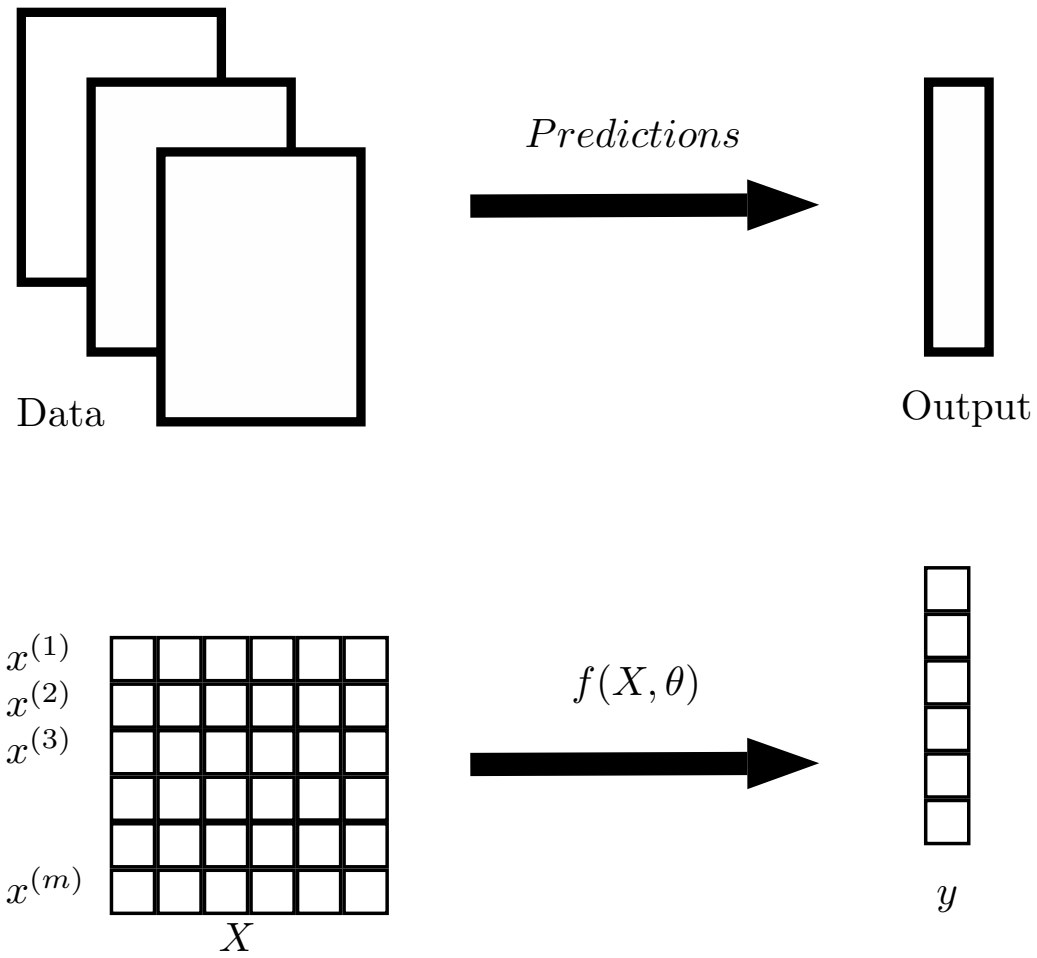
3. Extract the files in your computer
4. Open Jupyter-Notebook -> open the file `00_test.ipynb`
5. Open Orange -> open the file `00_test.ows`

A first problem: Bank telemarketing[1]

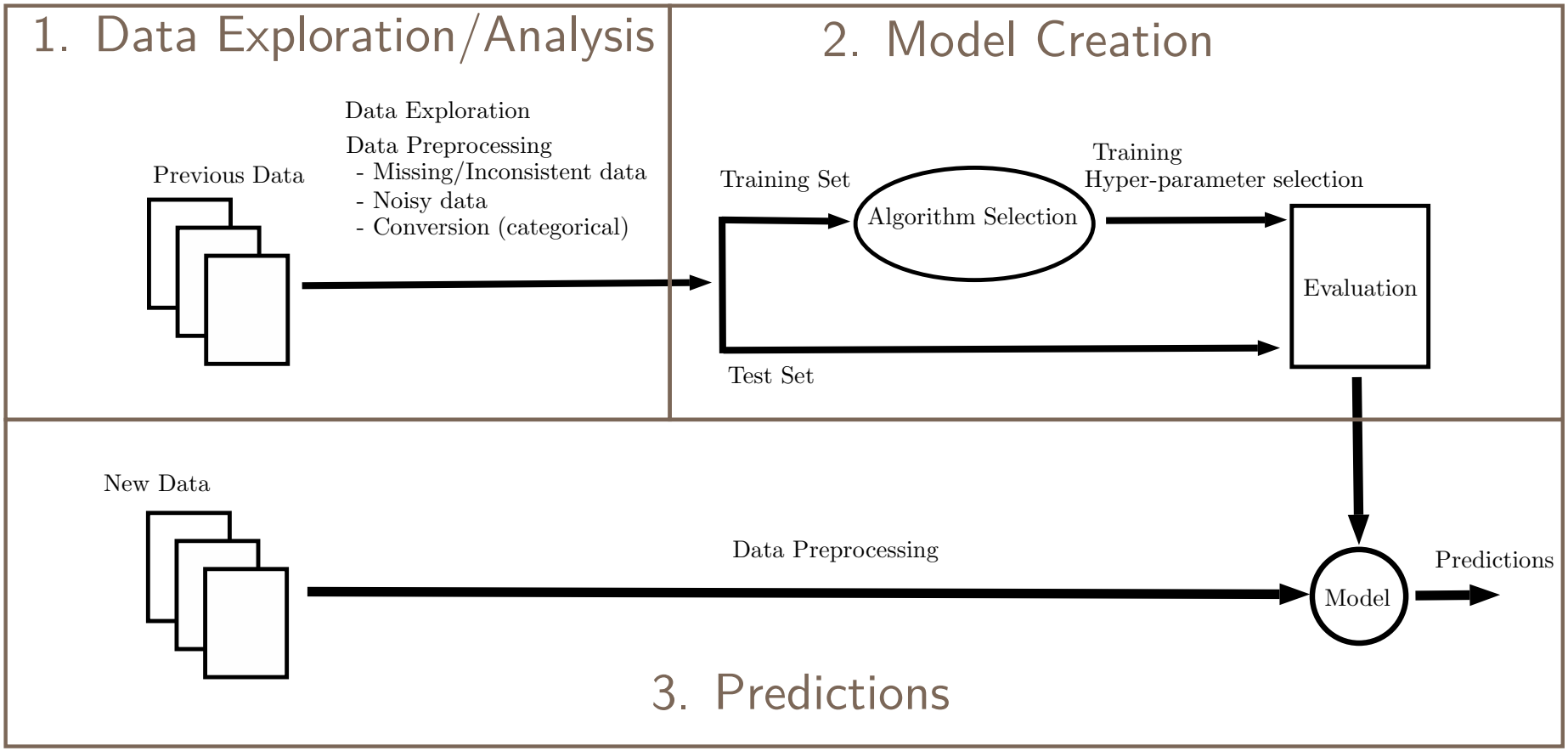
Attribute		Type	Description/Values
Personal	age	num	Age of the potential client
	job	cat	admin., blue- collar, entrepreneur, housemaid,... ,unknown
	marital_status	cat	divorced, married, single, unknown
	education	cat	basic.4y, basic.6y, basic.9y, high.school,.. unknown
Bank	default	cat	The client has 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.

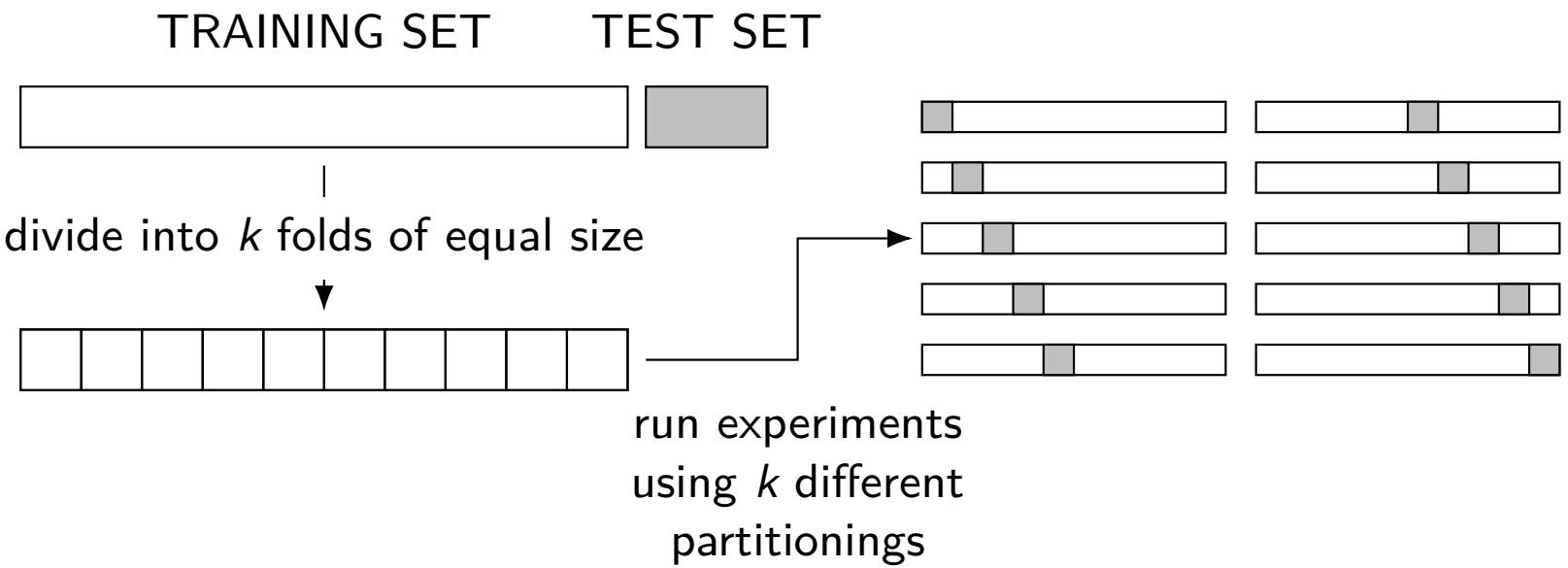
Supervised Learning



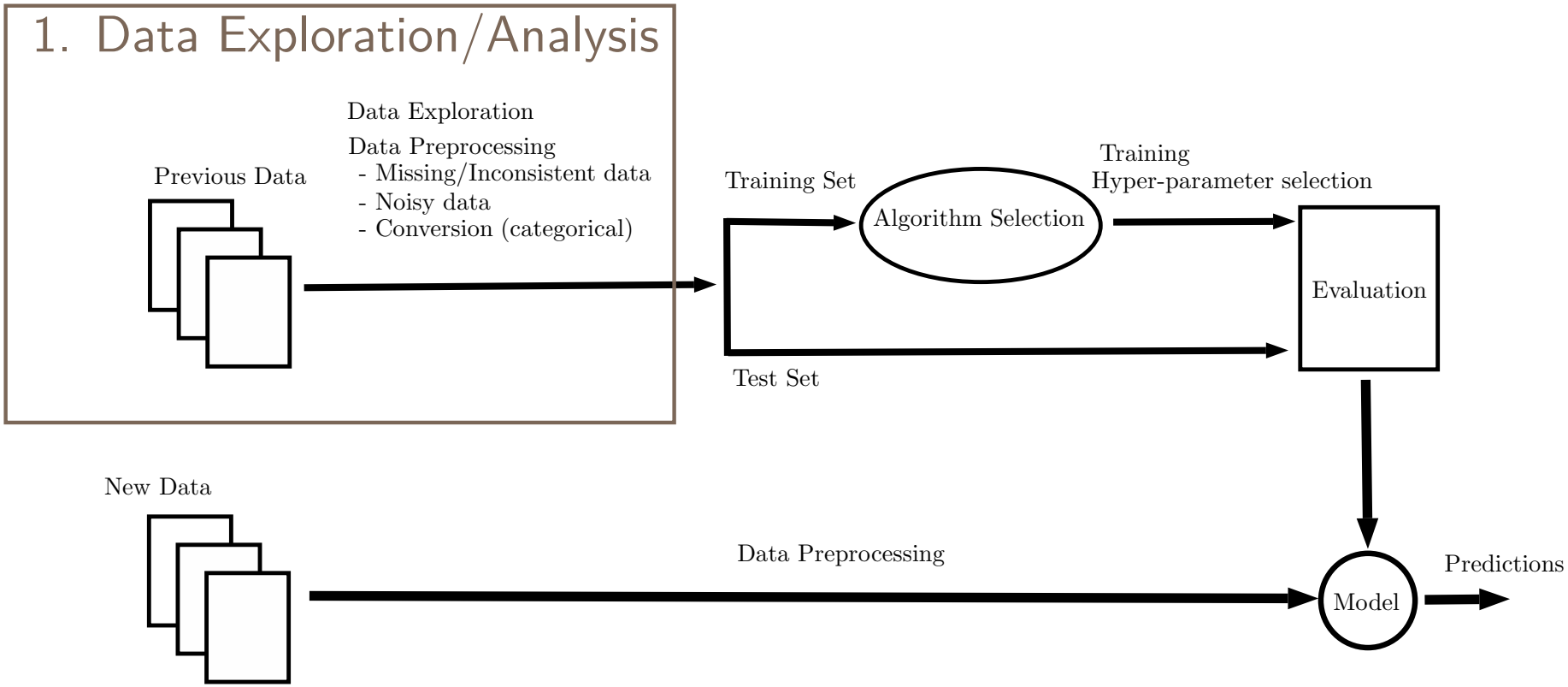
Supervised Learning Workflow



Cross validation

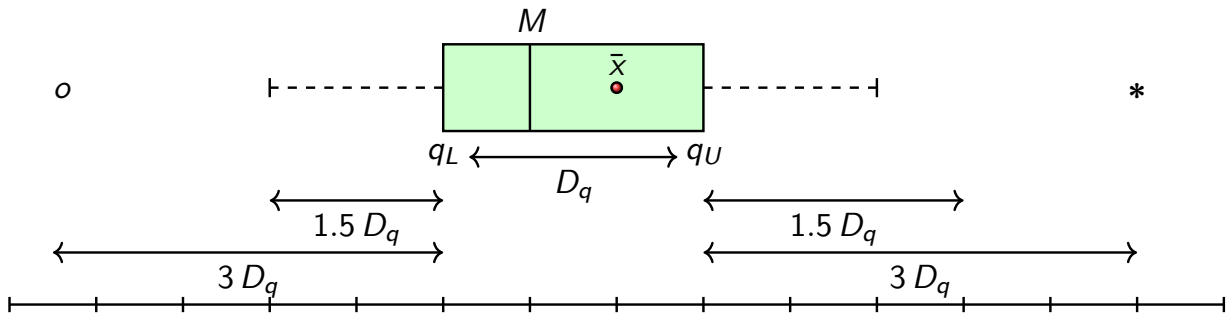


Data Pre-processing



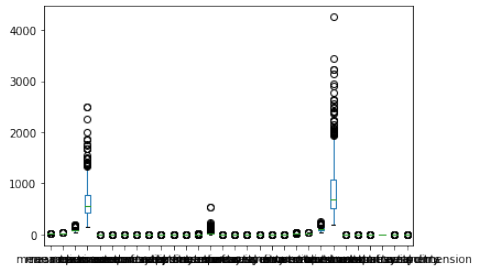
Data Exploration

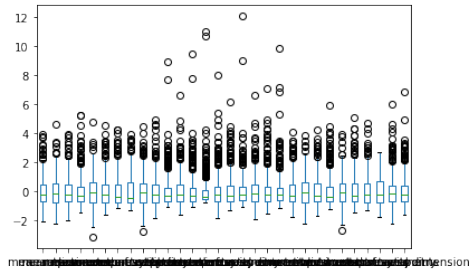
1. Uni-variate
   - Statistics
   - Histogram
   - Box-plot
2. Bi-variate
   - Scatter
   - Box-plot (by class)
3. Categorical
   - Contingency matrix



Data preparation

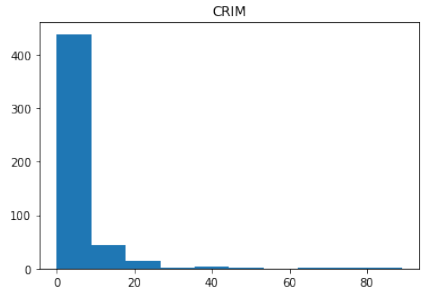
1. Data validation
   - Incomplete data (drop, replace)
   - Noisy data (Outliers)
2. Data transformation
   - Standardization
   - Discretization
   - Dummy variables
   - Feature construction
3. Data reduction
   - Sampling
   - Discretization
   - PCA: Principal Component Analysis

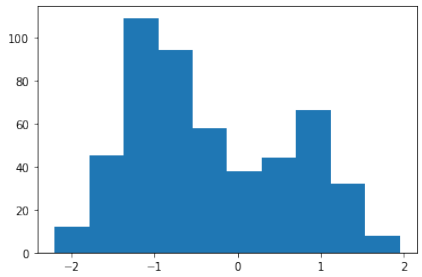




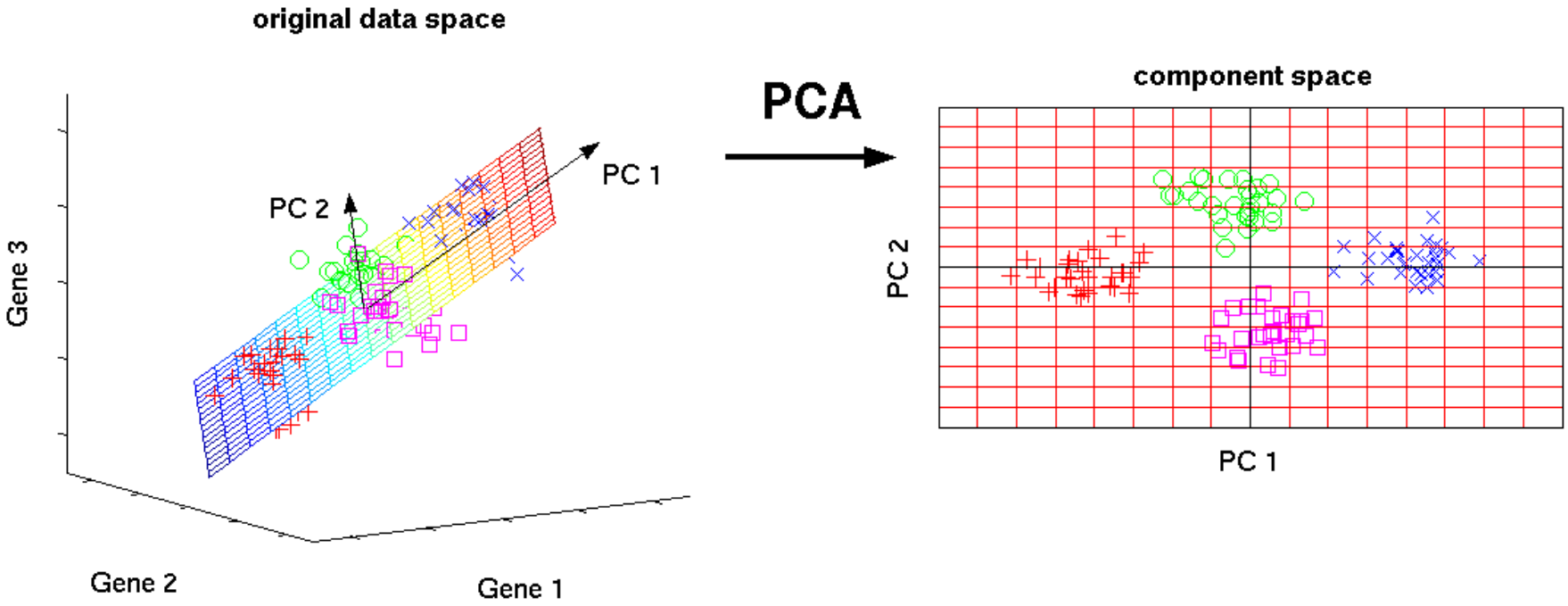
	type_of_food
0	fruit
1	vegetable
2	fruit
3	meat
4	fruit
5	vegetable

	fruit	meat	vegetable
0	1	0	0
1	0	0	1
2	1	0	0
3	0	1	0
4	1	0	0
5	0	0	1





Principal Component Analysis



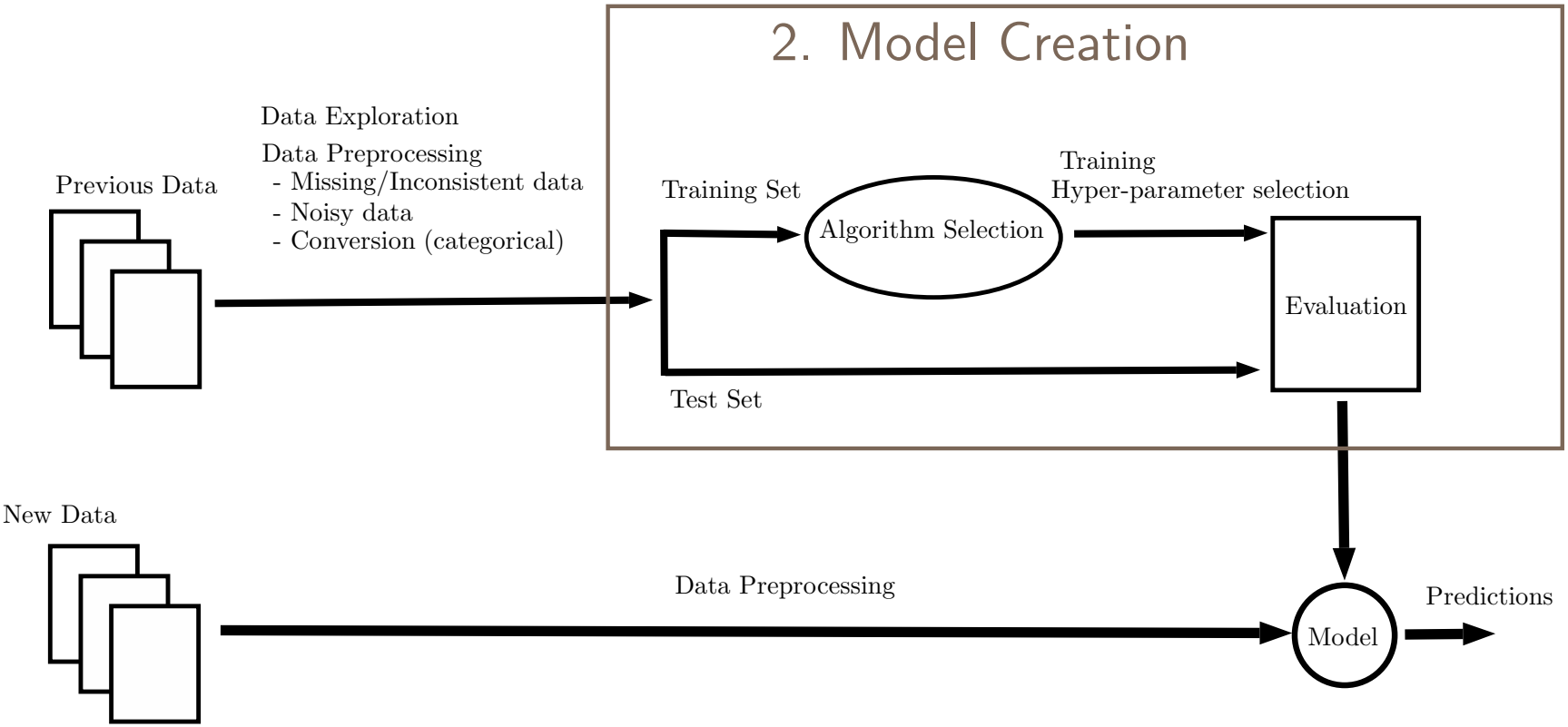
Example 2 - Customer Satisfaction Analysis

A survey in order to evaluate 20 different healthcare structures. 2000 customers have evaluated, with a 1-10 scale, each of six features of the service:

1. Courtesy
2. Clarity
3. Competence
4. Condition (of the structure)
5. Promptness (of the service)
6. Opening times

The problem: How can we represent the information in a simple and meaningful way?

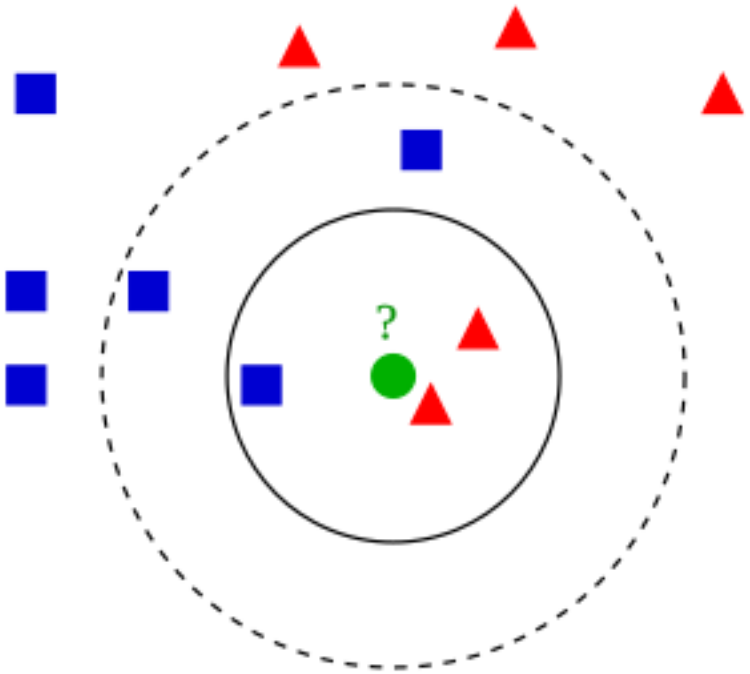
Model Creation/Evaluation



Classification Models

- Heuristics Methods
  - Nearest Neighbours
  - Classification Trees
- Probabilistic Methods
  - Bayesian Methods
- Optimization based Methods
  - Logistic regression
  - Support vector machine
  - Neural Networks

KNN K-nearest Neighbours



Main Parameters

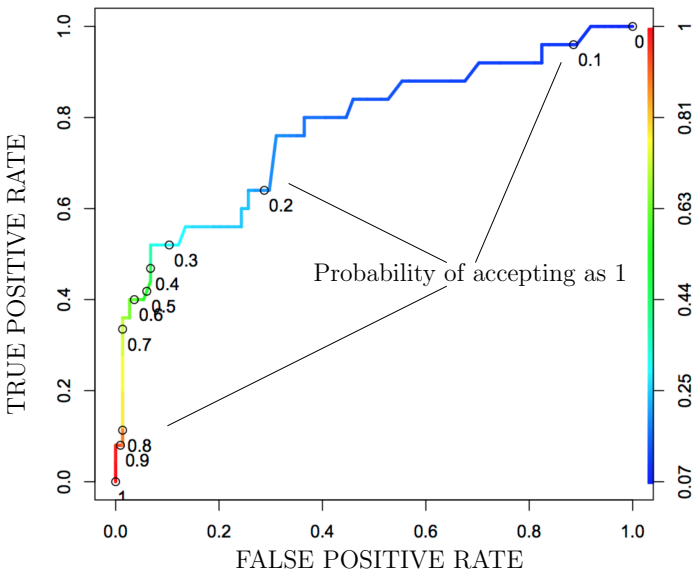
- k : number of neighbours
- neighbour weights
- distances

Quality measures

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

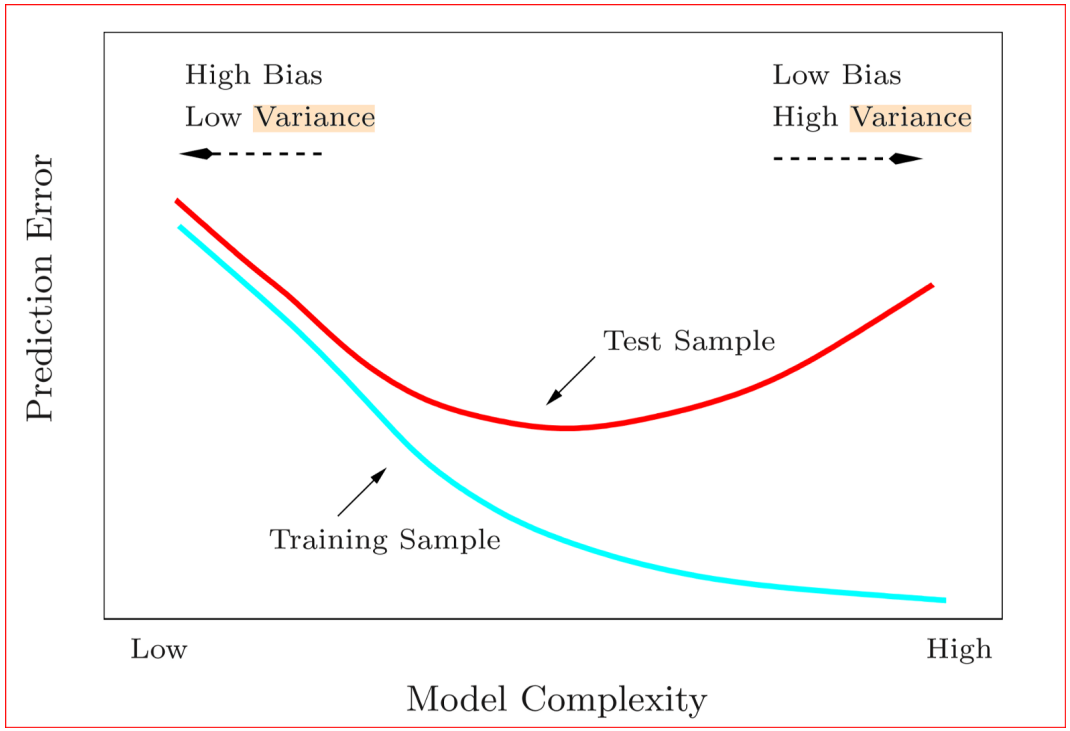
- Precision $= \frac{TP}{TP+FP}$
  "proportion of true positives among ***positive predictions****"*
- False Positive rate$= \frac{FP}{FP+TN}$
  "proportion of false positives among ***actual negatives****"*
- Recall (True Positive rate)$= \frac{TP}{FN+TP}$
  "proportion of true positives among ***actual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

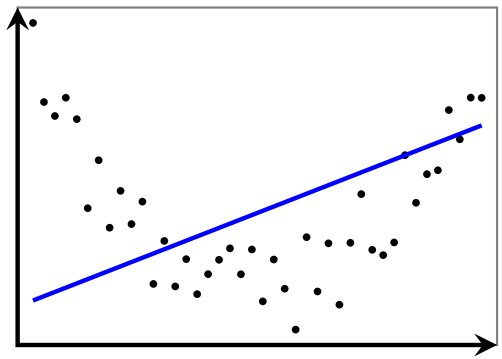
ROC curve & AUC



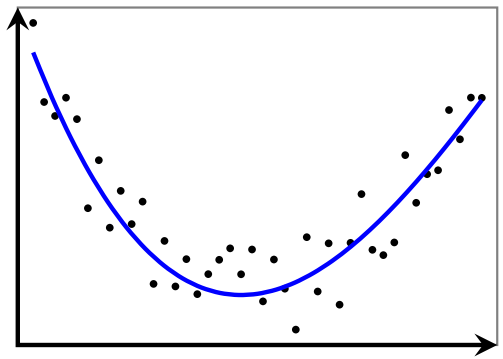
- If we accepting even with small probability then $TPR = FPR = 1$
- If we accepting just with high probability then $TPR = FPR = 0$
- The perfect classificator is the the point $(0, 1)$
- $AUC \in [0.5, 1]$ area under the curve is a quality measure of our algorithm.

Under/Over-fitting

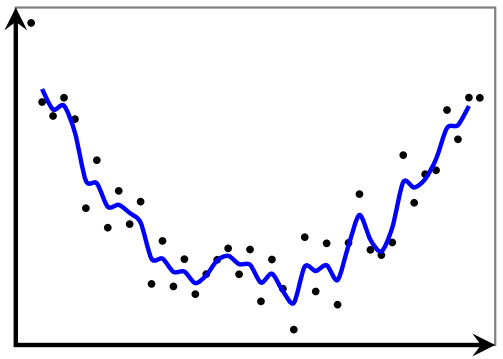




Underfitting

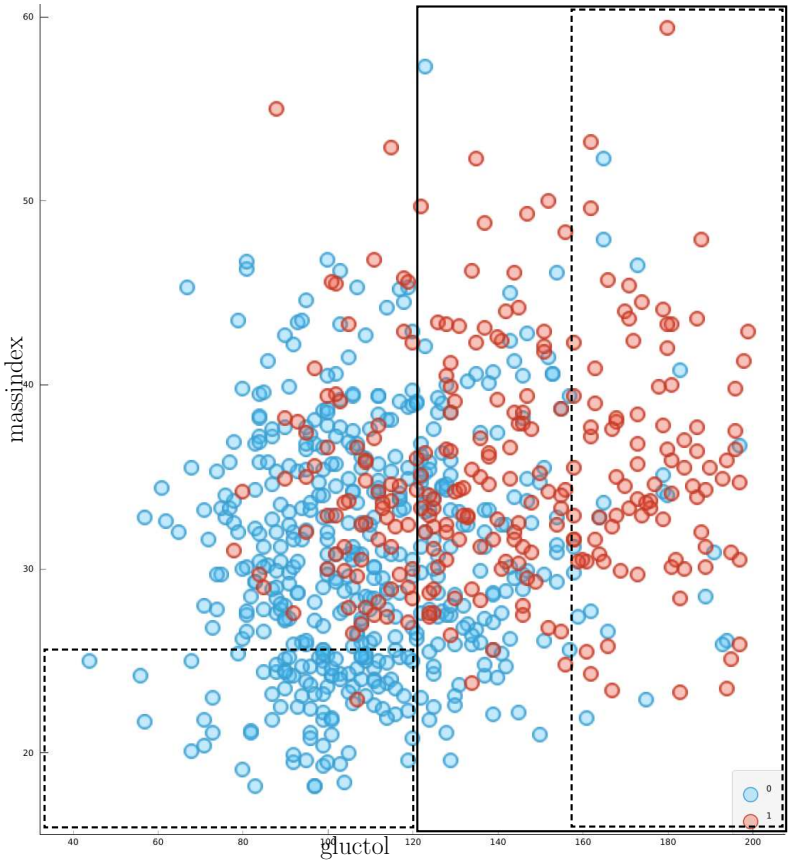


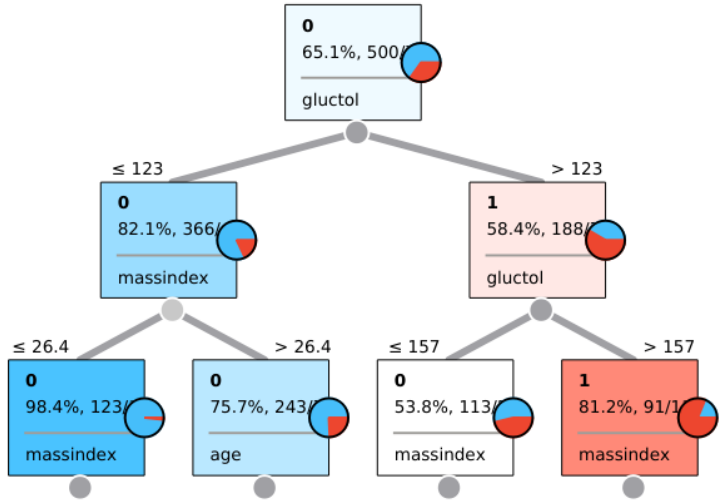
Balance



Overfitting

Decision tree





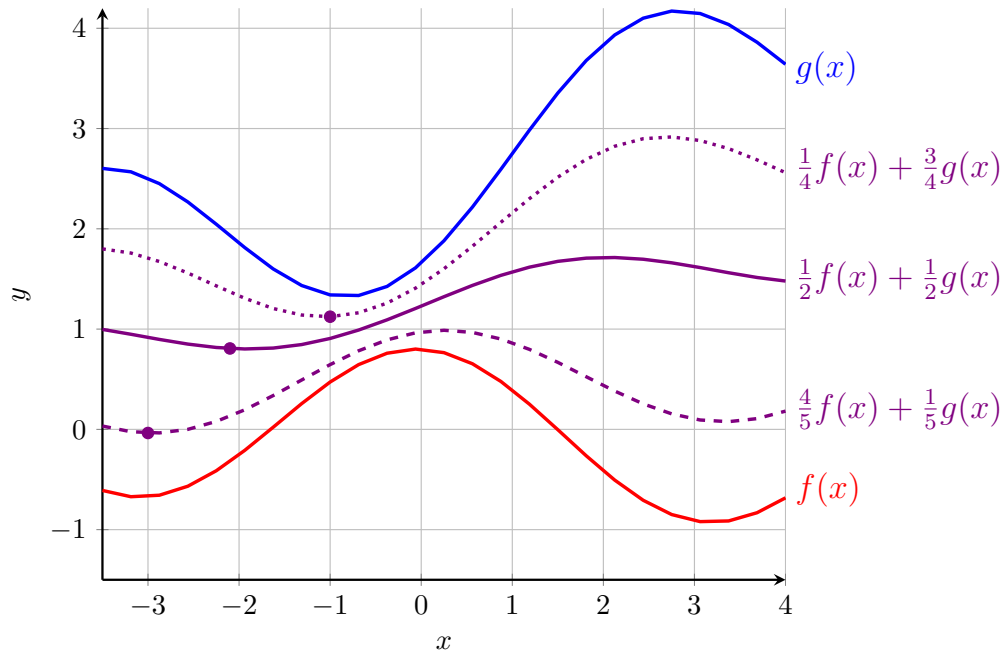
Main Parameters

- 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

Multi-objective optimization

$$\min_{x \in K} \lambda_1 \cdot f(x) + \lambda_2 \cdot g(x)$$

λ_i : pesi relativi

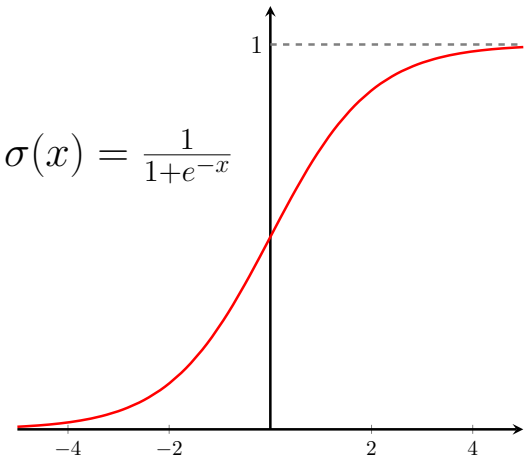


Logistic regression

We compute a weighted score:

$$w^\top x = w_0 + w_1 x_1 + \cdots + w_n x_n,$$

that we translate into a probability



$$P(y = 1|x) = \sigma(w^\top x)$$

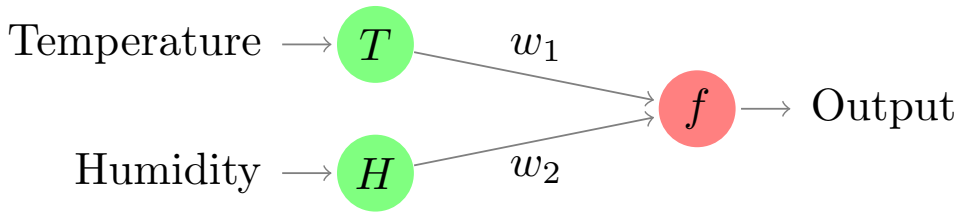
$$\min_w \underbrace{\frac{1}{2}||w||^2}_{\text{regularization}} + C \underbrace{\sum_{i=1}^{n} \log(1 + \exp(-y_i(w^T X_i)))}_{\text{likelihood (error)}}$$

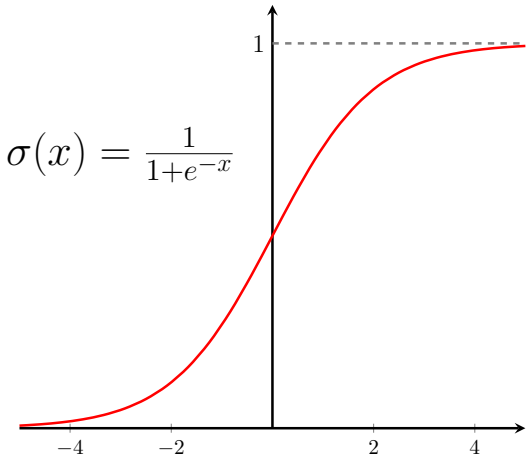
Main Parameters

- 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 int
  - 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





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

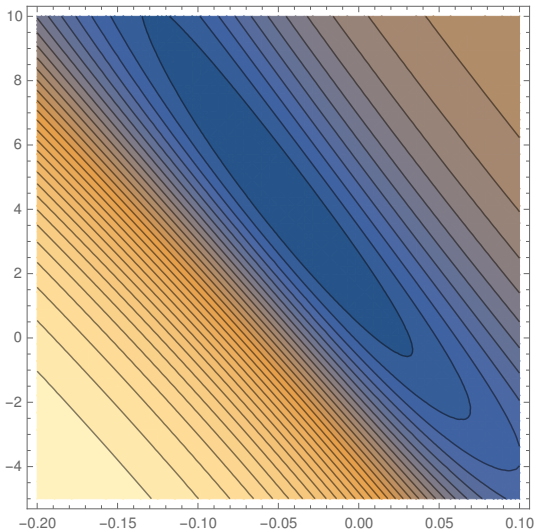
$$\max_{w_1, w_2} \sum_{i=1}^{5} [P_i(\text{rain}) - f(T_i, H_i, w_1, w_2)]^2$$

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

Multi-Layer Perceptron - small example

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

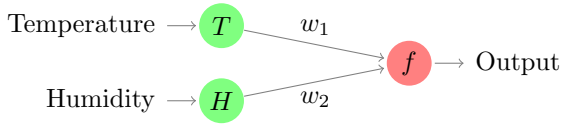
$$= \min \left[0.7 - 1/(1 + e^{-(w_1 \cdot 20 + w_2 \cdot 0.4)})\right]^2 + \left[0.52 - 1/(1 + e^{-(31 \cdot w_1 + w_2 \cdot 0.36)})\right]^2 + \cdots$$

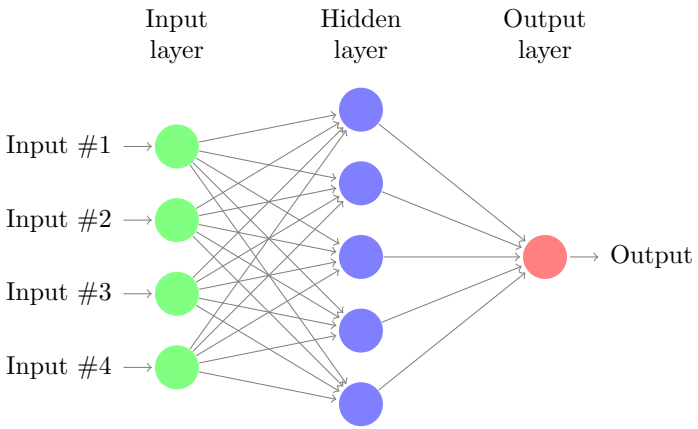


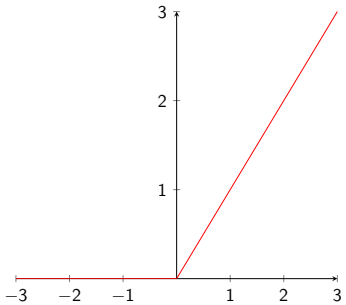
$$(w_1^*, w_2^*) = (-0.044, 4.147)$$

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
Predicted	0.70	0.56	0.58	0.75	0.60
Error	**0.0**	**-0.04**	**-0.03**	**-0.02**	**0.0**

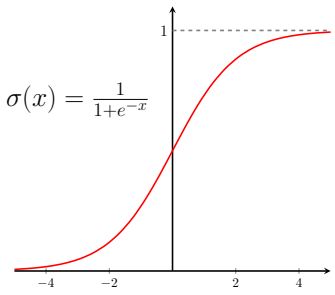
Multi-Layer Perceptron







ReLU

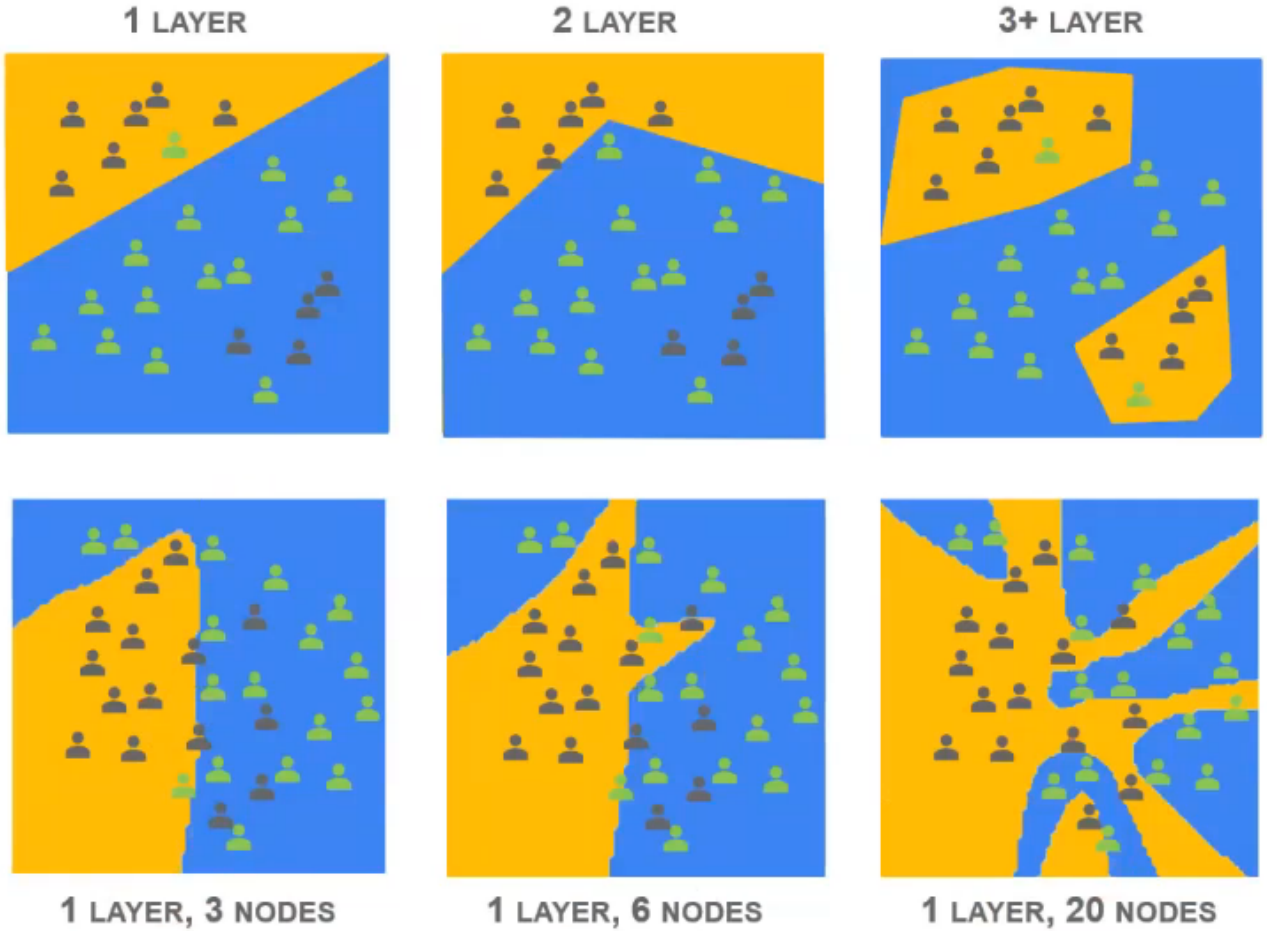


Sigmoid

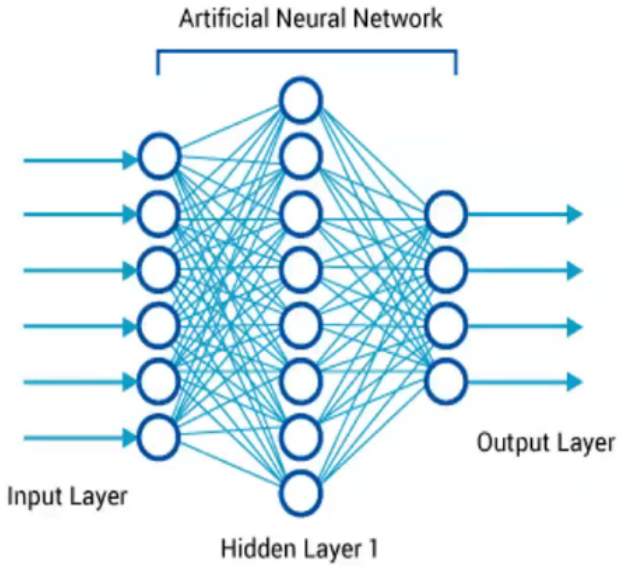
Main Parameters

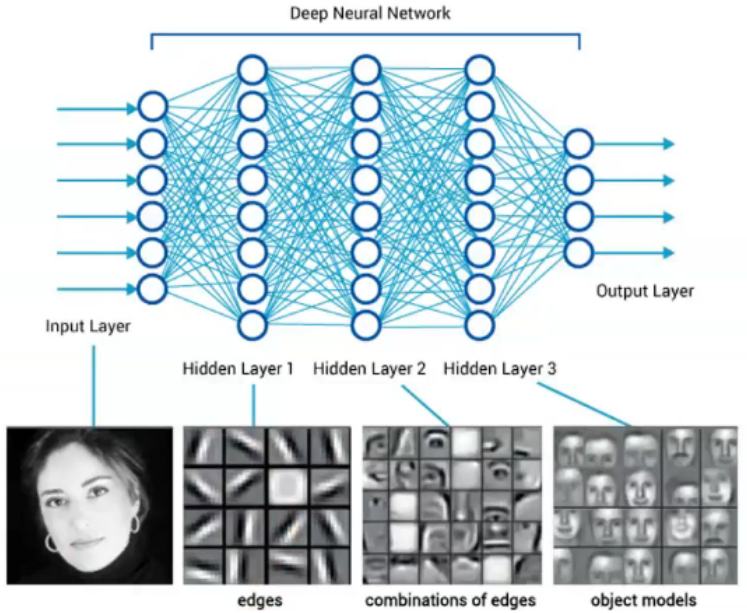
- hidden_layer_sizes: $(n_1, n_2, \ldots, n_L)$
- activation: identity, logistic, tanh, relu
- alpha regularization term parameter
- Resolution algorithm parameters: solver, tol, batch_size, learning_rate, max_iter.

Neural Network

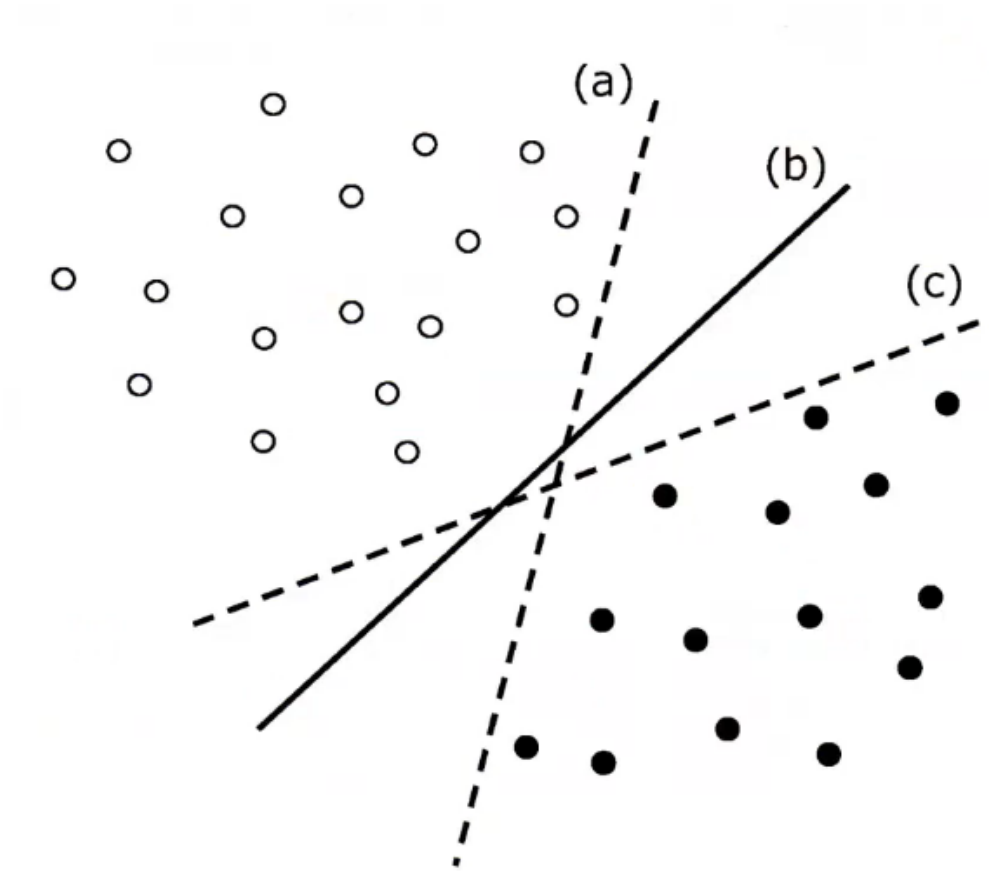


Deep Learning



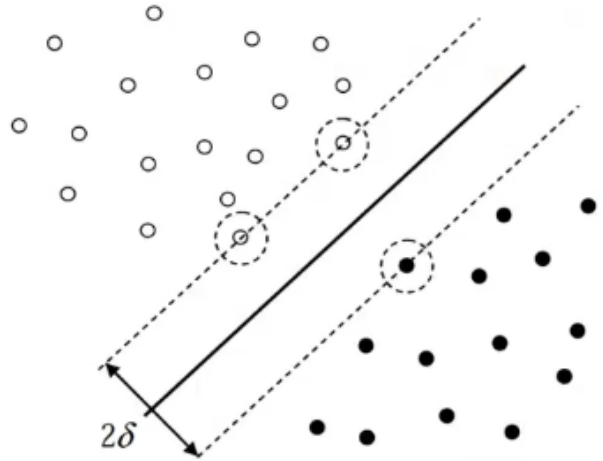


Support Vector Machine - linearly separable

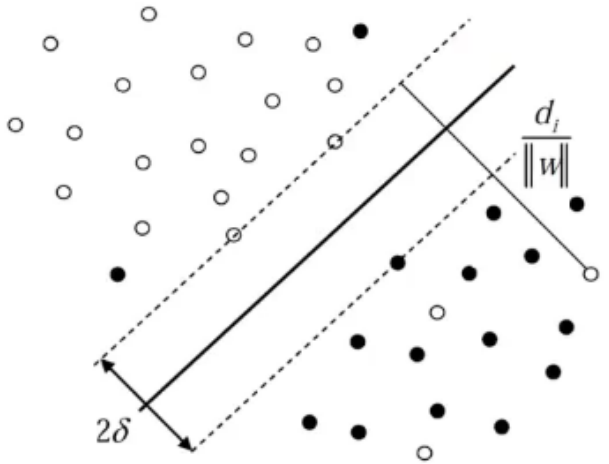


SVM - general case

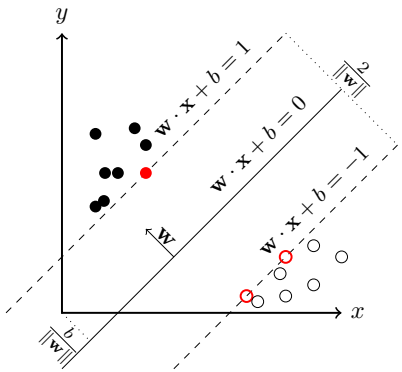
$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|_2^2$$
$$\text{s. t.} \quad y_i(\mathbf{w}'\mathbf{x}_i - b) \geq 1 \quad i \in \mathcal{M}$$

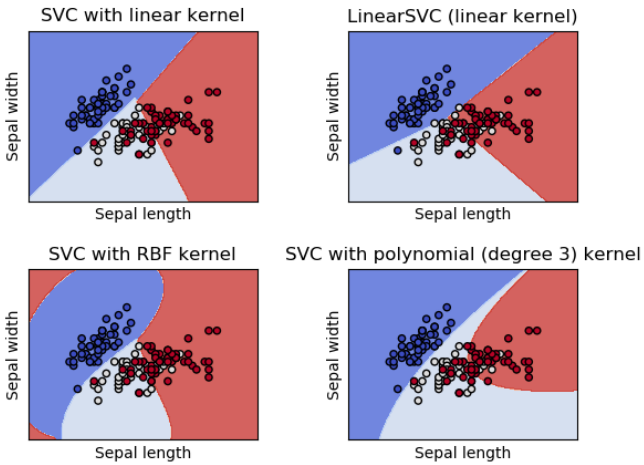


$$\min_{\mathbf{w},b,d} \quad \frac{1}{2}\|\mathbf{w}\|_2^2 + \lambda \sum_{i=1}^{m} d_i$$
$$\text{s. t.} \quad y_i(\mathbf{w}'\mathbf{x}_i - b) \geq 1 - d_i \quad i \in \mathcal{M}$$
$$d_i \geq 0 \quad i \in \mathcal{M}$$



SVM - general case





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

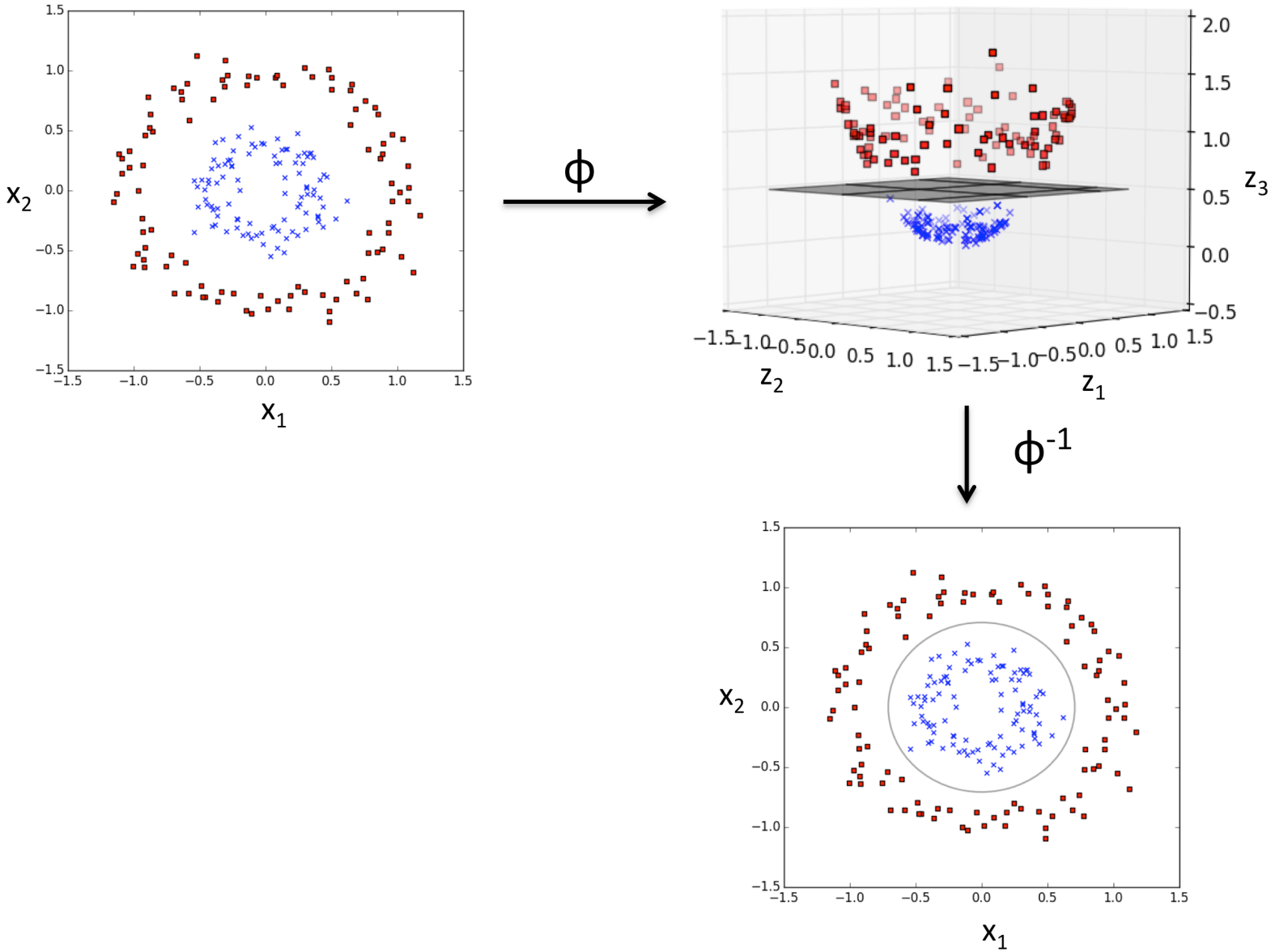
$$\text{subject to } y_i(w^T \underbrace{\phi(x_i)}_{\text{kernel}} - b) \geq 1 - d_i,$$

$$d_i \geq 0$$

Main Parameters

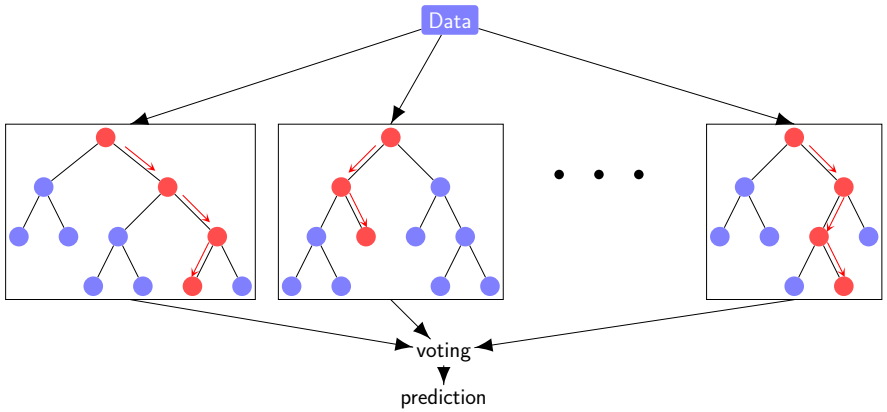
- C: Inverse of regularization strength
- kernel:
  - linear: $x'x$
  - poly: $(\gamma x'x + r)^d$
  - rbf: $exp(-\gamma||x - x'||^2)$
  - sigmoid: $\tanh(\gamma x'x + r)$
- degree(d), gamma(γ), coef0(r)
- Resolution algorithm parameters

SVM - kernels

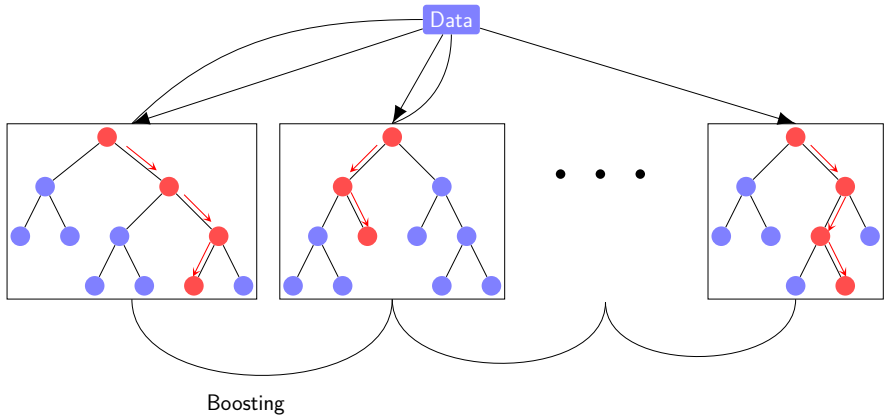


Ensemble Methods

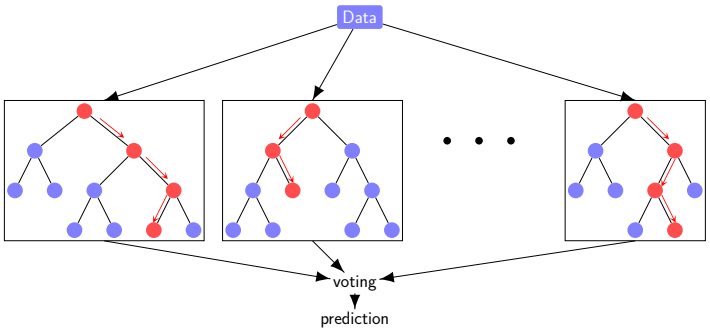
Bagging



Boosting



Bagging : Random Forest

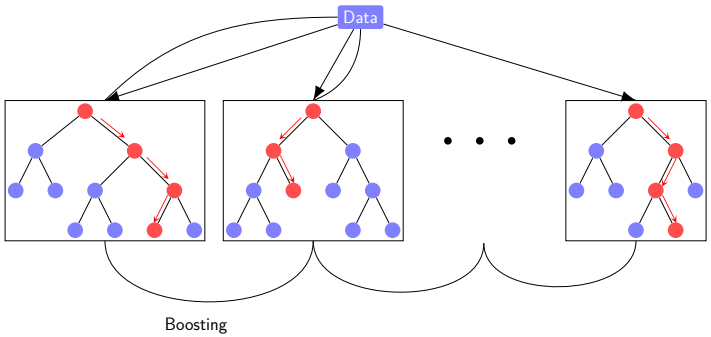


1. Create different (simple) tree models (stumps)
2. Each model is created with a subset of observation/features ($\sim$2m/3)
3. We combine the prediction of all trees

Main Parameters

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

Boosting: Adaboost

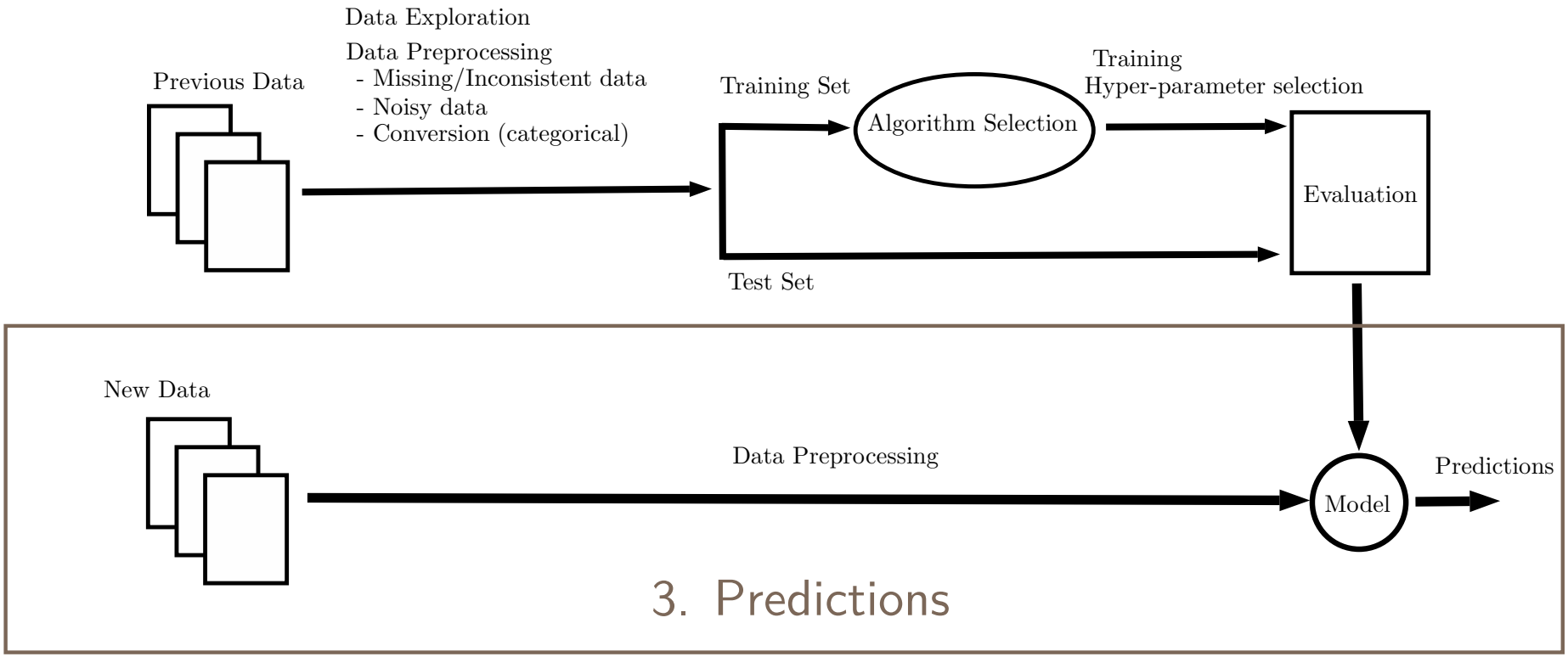


1. Assign equal weights to observations
2. Select a sample of observations based on the weights.
3. Create a first weak model
4. Evaluate the performance of the model and assign its weigh. $(\frac{1}{2}\log((1 - error)/error))$
5. Update sample weights
6. GOTO 2
7. Final combined prediction

Main Parameters

- n_estimators: Number of estimators
- base_estimator: Weak estimator type
- learning_rate: weights of estimator in final decision

Supervised Learning Workflow



Predictions

- Be sure to apply the **SAME** transformation (standardization, imputation, new variables, PCA, etc.) before apply the selected model.

Regression Models

- Heuristics Methods
  - Nearest Neighbours
  - Regression Trees
- Optimization based Methods
  - Linear models
  - Support vector machine
  - Neural Networks

Quality measures - Regression

- Coefficient of determination

$$R^2 = 1 - \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 / \sum_{i}(y_i - \bar{y})^2$$

- Mean Absolute Error :

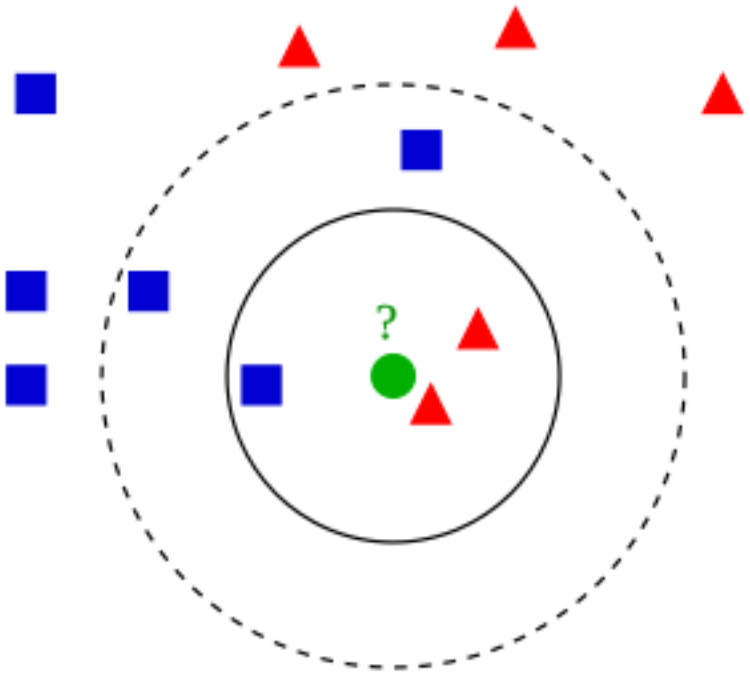
$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

- Mean Squared Error :

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- Root Mean Squared Error : $RMSE = \sqrt{MSE}$

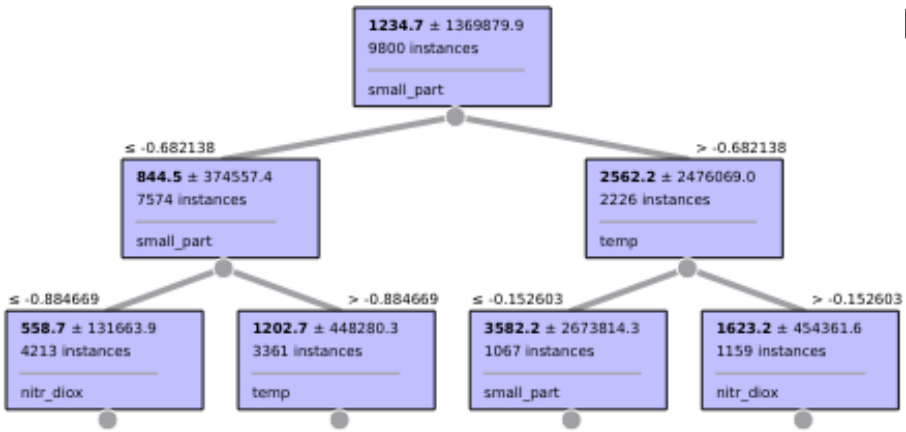
KNN K-nearest Neighbours



Main Parameters

- k : number of neighbours
- neighbour weights
- distances

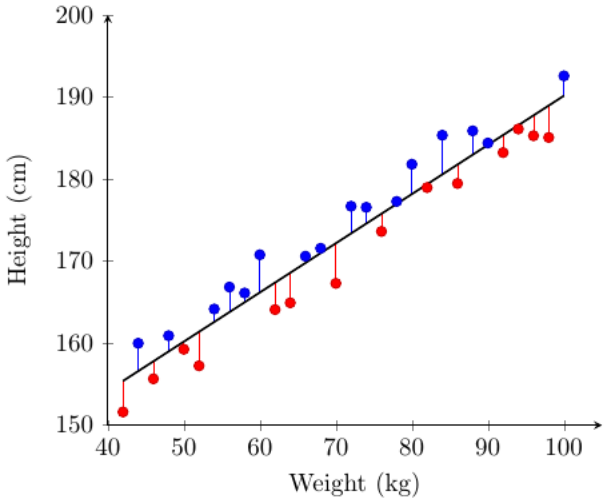
Regression tree



Main Parameters

- variability measure: "variance"
- 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

Linear Models



$$\min_{\mathbf{w}} RR(\mathbf{w}, \mathcal{D}) = \min_{\mathbf{w}} \lambda \|\mathbf{w}\|^2 + \sum_{i=1}^{m} (y_i - \mathbf{w}'\mathbf{x}_i)^2$$
$$= \min_{\mathbf{w}} \lambda \|\mathbf{w}\|^2 + (\mathbf{y} - \mathbf{X}\mathbf{w})'(\mathbf{y} - \mathbf{X}\mathbf{w}).$$

$$\min_{\mathbf{w}} LR(\mathbf{w}, \mathcal{D}) = \min_{\mathbf{w}} \lambda |\mathbf{w}| + \sum_{i=1}^{m} (y_i - \mathbf{w}'\mathbf{x}_i)^2$$
$$= \min_{\mathbf{w}} \lambda |\mathbf{w}| + (\mathbf{y} - \mathbf{X}\mathbf{w})'(\mathbf{y} - \mathbf{X}\mathbf{w}).$$

- **linear**

$$Y = w_1 X_1 + w_2 X_2 + \cdots + w_n X_n + b = \sum_{j=1}^{n} w_j X_j + b.$$

- **quadratic** $Y = b + wX + dX^2 \qquad Z = X^2$

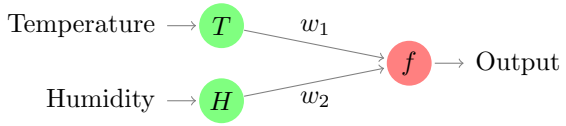
$$Y = b + wX + dZ.$$

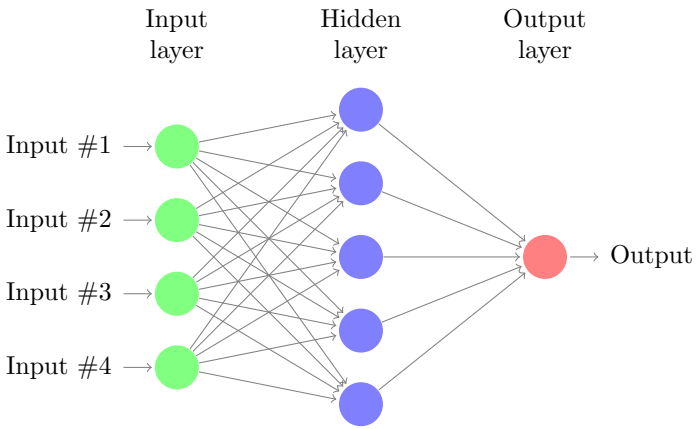
- **exponential** $Y = e^{b+wX} \qquad Z = \log Y. \qquad Z = b + wX.$

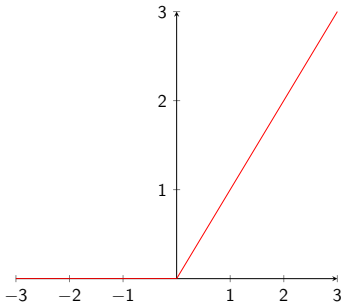
Main Parameters

- λ: regularization strength

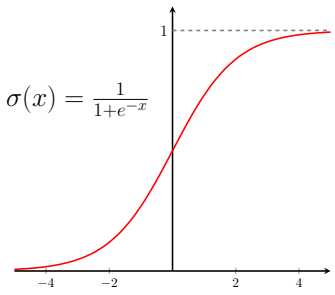
Multi-Layer Perceptron







ReLU

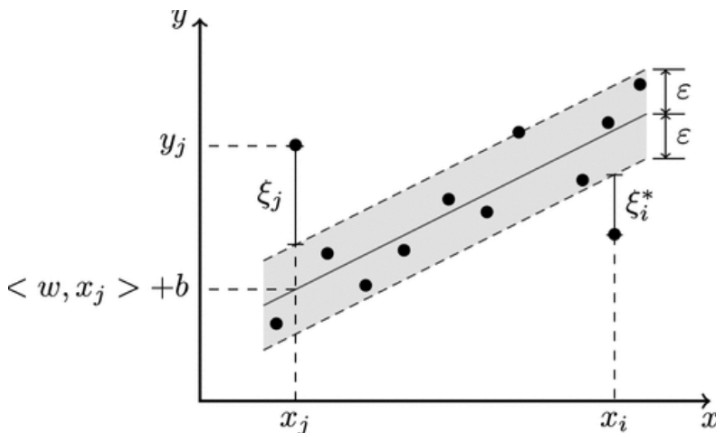


Sigmoid

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.

SVR



$$\min_{w,b,\zeta,\zeta^*} \frac{1}{2}\|w\| + C\sum_{i=1}^{n}(\zeta_i + \zeta_i^*)$$

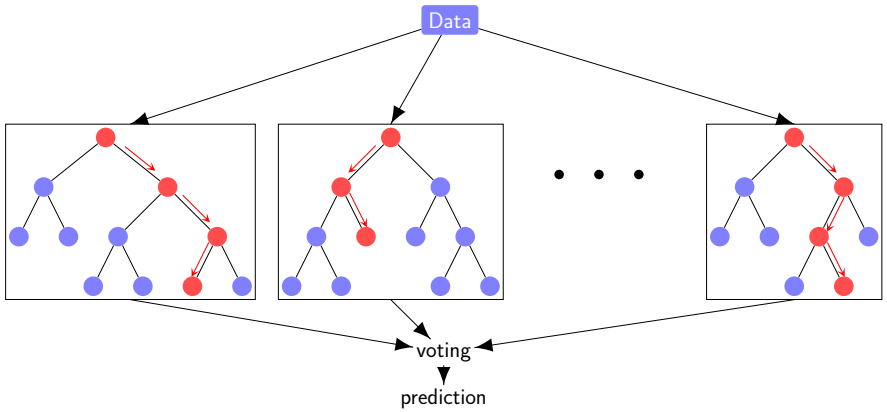
$$\text{subject to } y_i - w^T\phi(x_i) - b \leq \varepsilon + \zeta_i,$$
$$w^T\phi(x_i) + b - y_i \leq \varepsilon + \zeta_i^*,$$
$$\zeta_i, \zeta_i^* \geq 0, i = 1, ..., n$$

Main Parameters

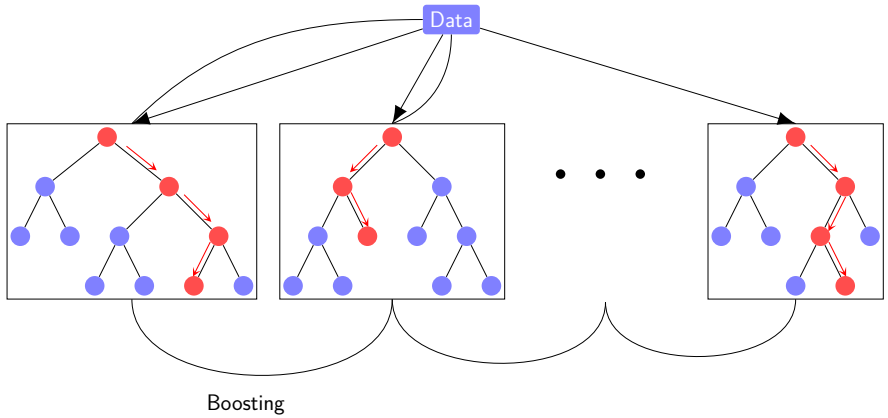
- C: inverse of regularization strength
- ε: tolerance
- kernel
- Resolution algorithm parameters

Ensemble Methods

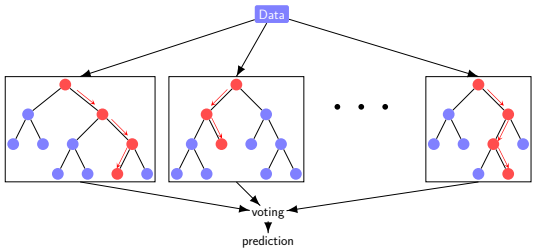
Bagging



Boosting



Random Forest

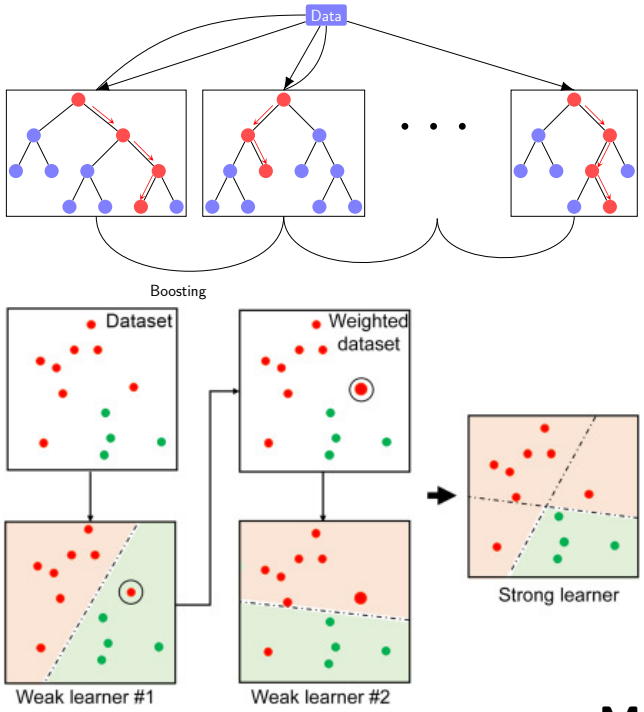


1. Create different (simple) tree models (stumps)
2. Each model is created with a subset of observation/features ($\sim$2m/3)
3. We combine the prediction of all trees

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
   - $F^{(k+1)} = F^{(k)} + \lambda f_k$