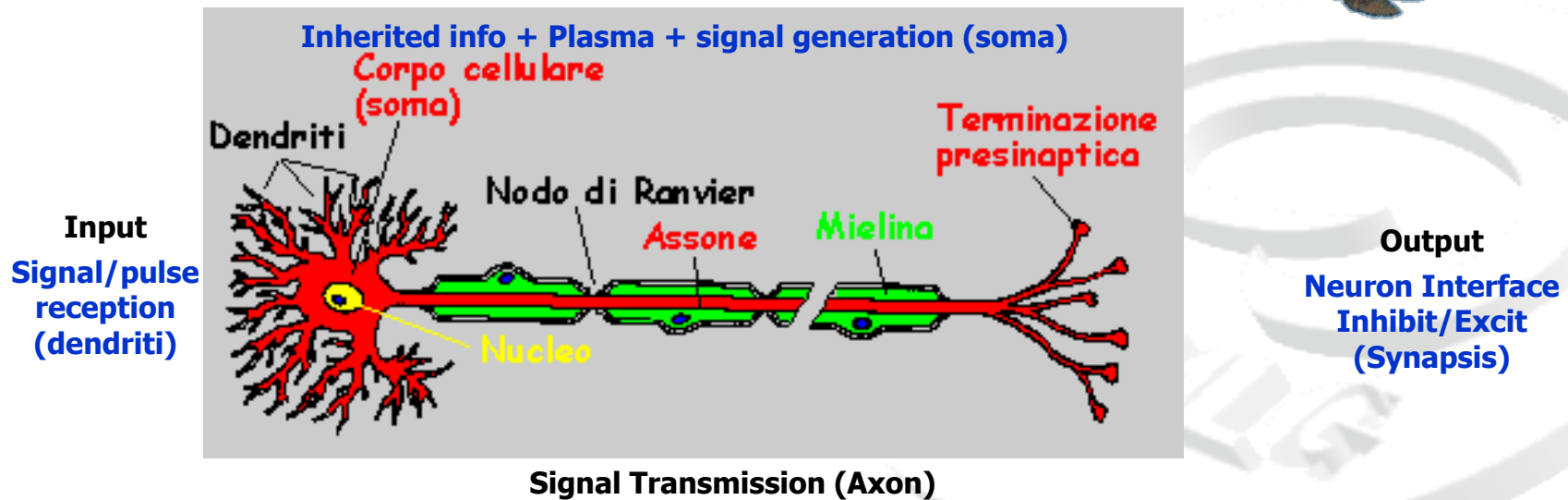


Artificial Neural Networks (ANN)

ANN emulate the Neural Network of the brain

- Human brain contains 10 billion neurons approx. Each one connected, at least, other 10.000 neurons (synapsis).
- Signals are transmitted through the neurons (electrochemical pulse) (modulated message)





Human Brain Cortex:

Cognitive Process

- time (milisegundos)
- Operation **Parallel Massive**
- Sequential 100 Steps





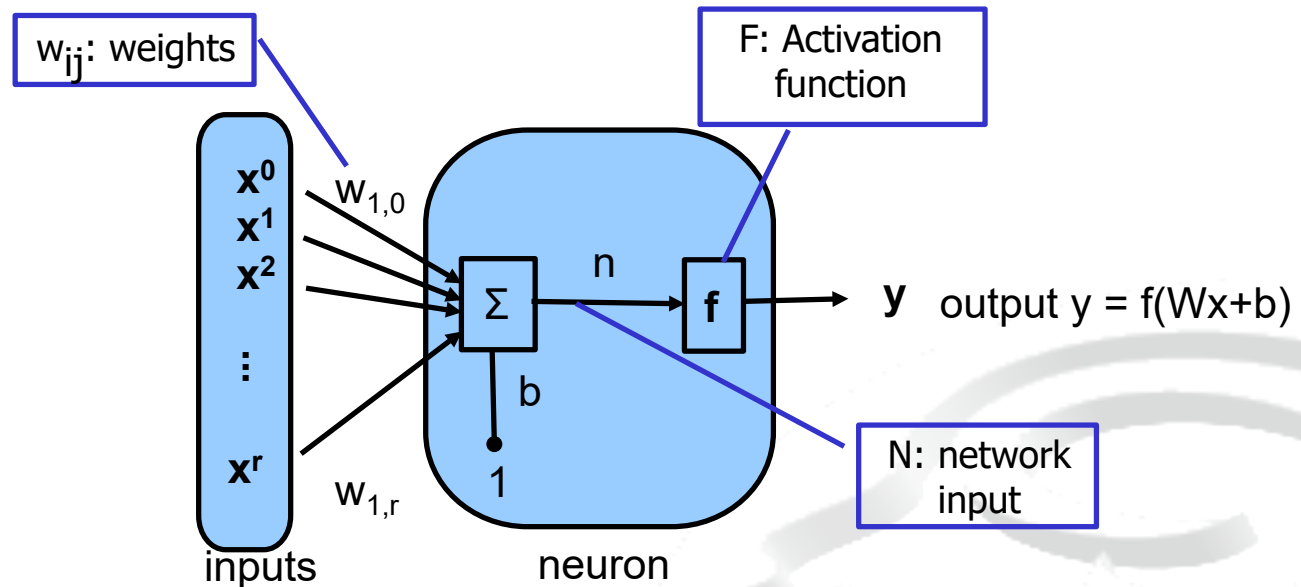
Main Advances

- 1943 W.McCulloch, W. Pitts: Modelo ANN (El Perceptrón)
- 1959 Bernard Widrow (SU): Filtro Adaline (Adaptative Linear Neuron) y el Madaline (Multiple Adaline) Basado en (ANN)
- 1969 Minsky y Papert: El Perceptrón (limitaciones).
- 1982 J. Hopfield: Memoria Asociativa "Redes de Hopfield".
- 1986 Rumulhart, Hunton y Williams : redescubren el BPL algoritmo de "back-propagation learning" (Paul Werbor, 1974)
- 1989 K. Hornik, M. Stinchcombe, H. White: Multi-FANN y Aproximación Universal



Artificial Neural Networks (ANN)

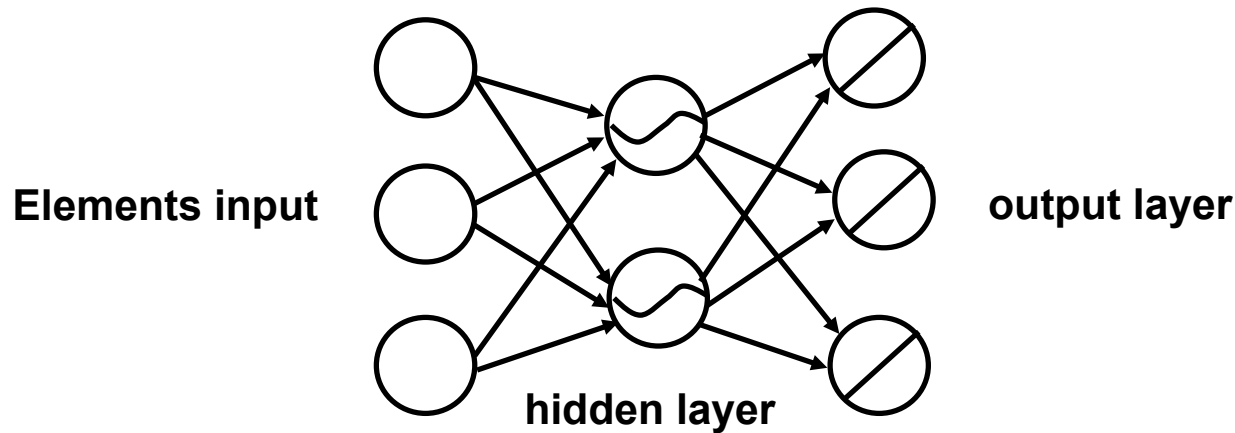
A simple model of one artificial neuron



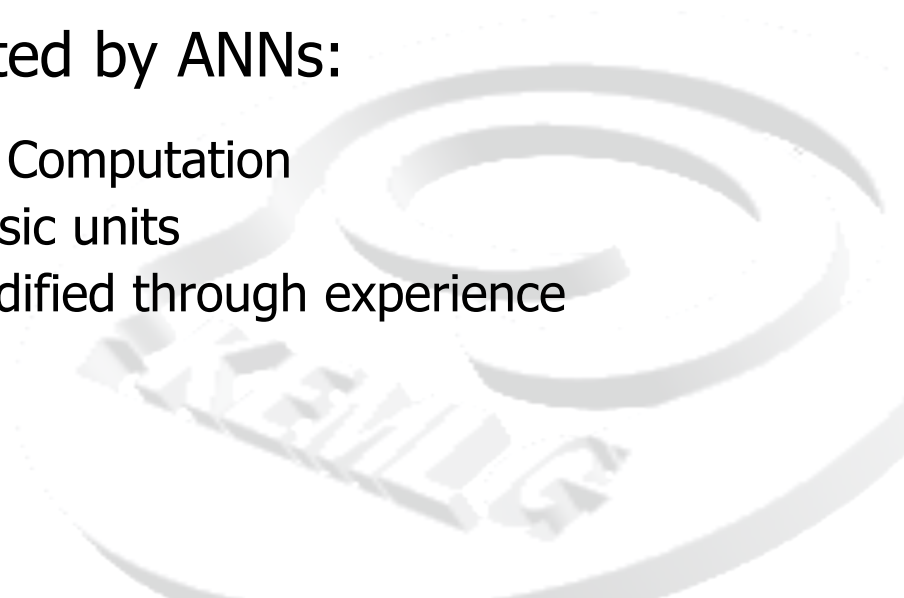


Artificial Neural Networks (ANN)

Structure of a multi-layer network

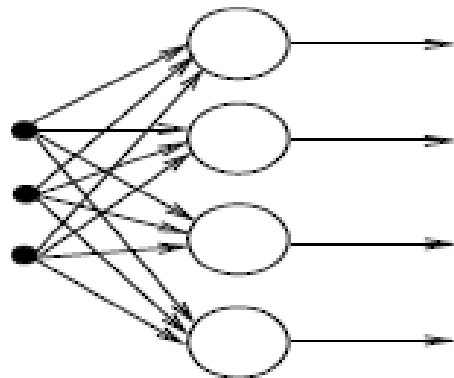


- Brain properties emulated by ANNs:
 - Parallel and Distributed Computation
 - Dense connection of basic units
 - Connections can be modified through experience
 - Learning is constant

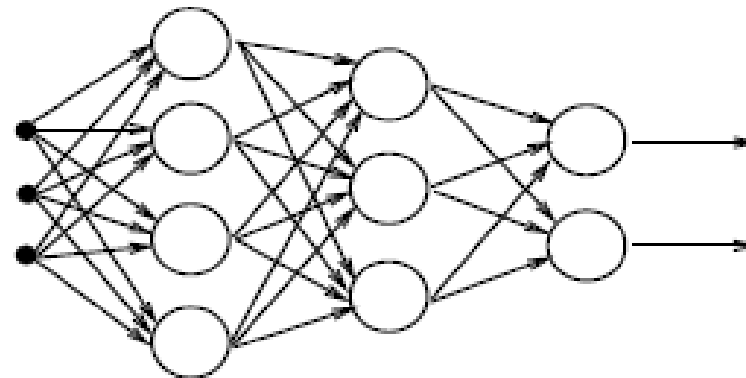




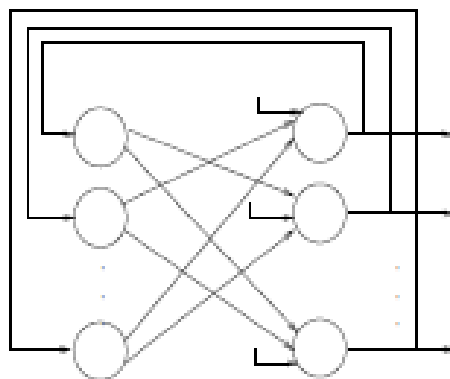
Architecture (Topology)



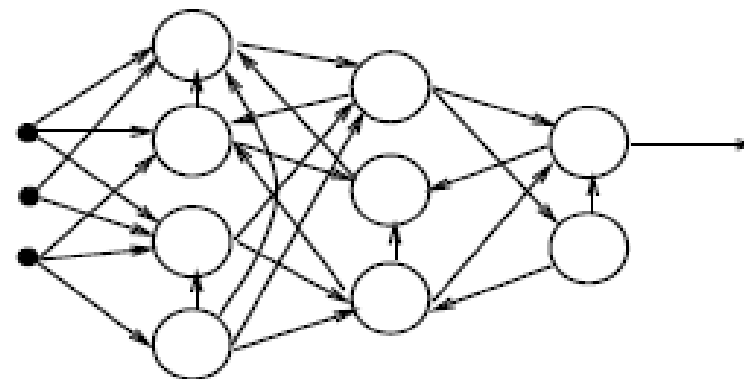
single-layer feed-forward



multi-layer feed-forward



single-layer recurrent

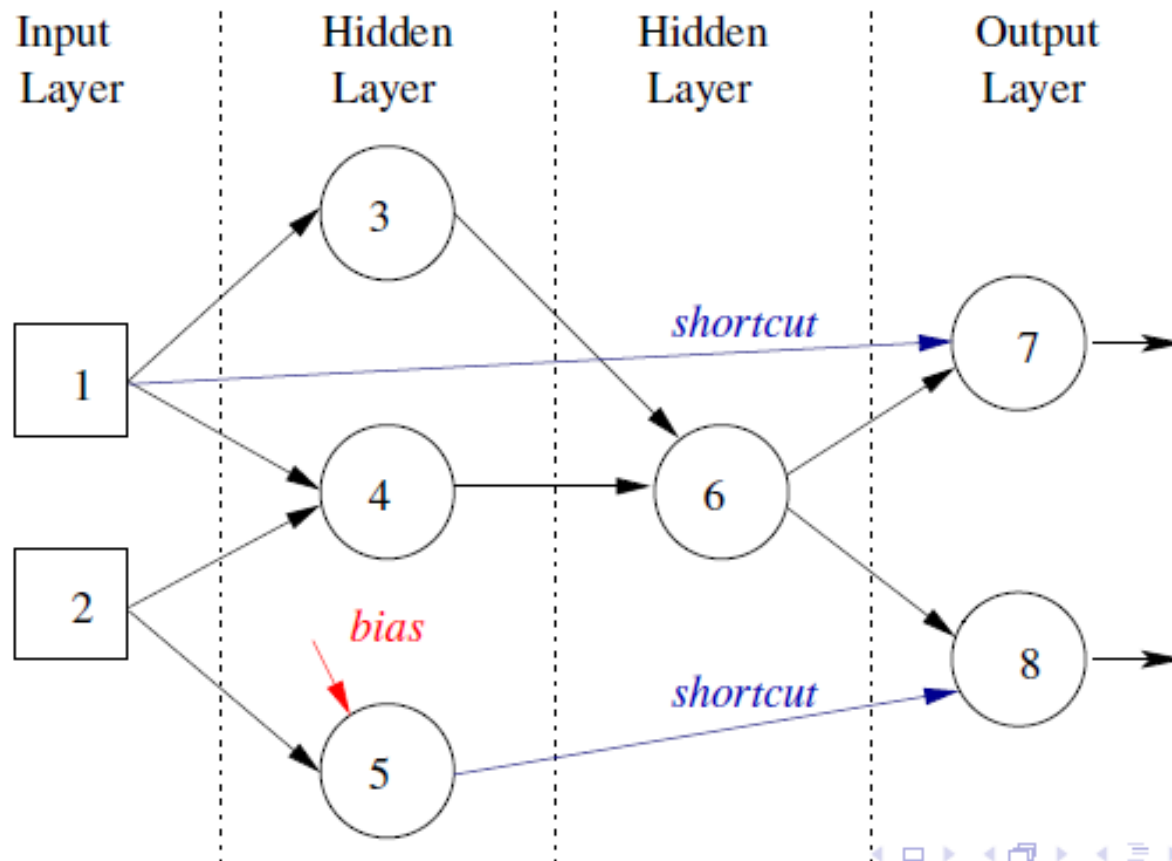


multi-layer recurrent

Multilayer Perceptrons (MLPs)

[Bishop, 1995][Haykin, 2009]

- Most popular Neural Network type;
- Uses **feedforward** connections and nodes are organized in **layers**;

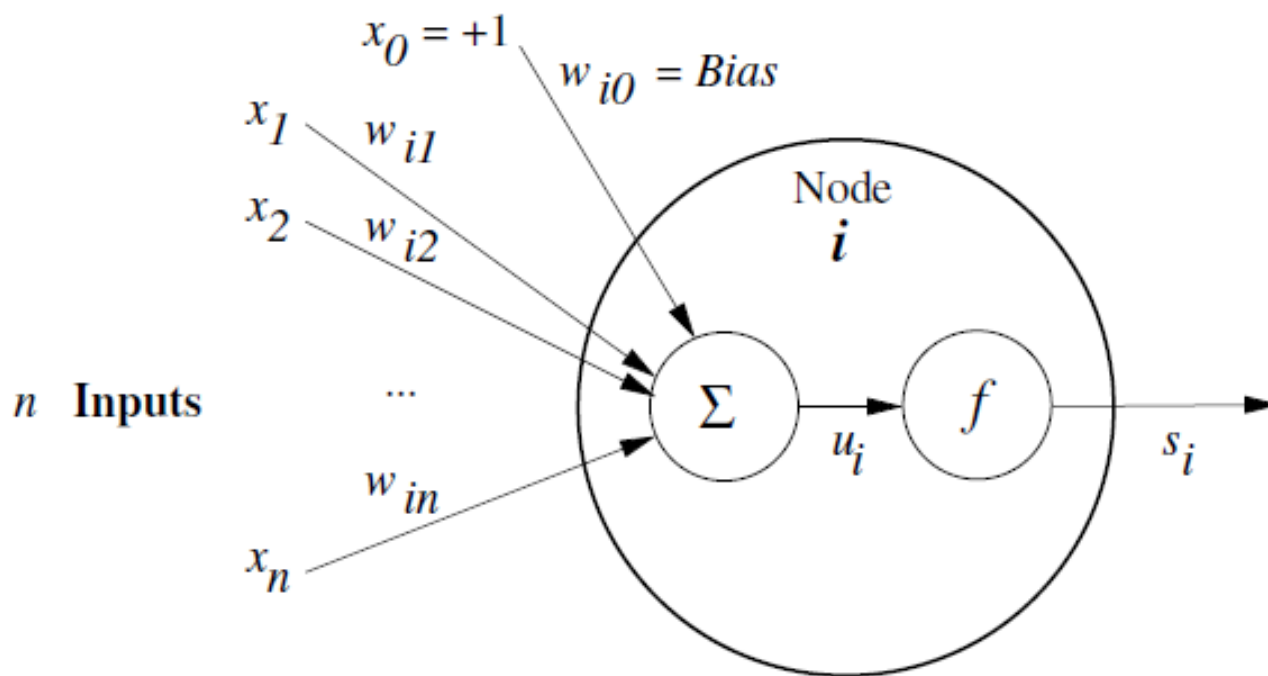


Multilayer Perceptrons (MLPs)

[Bishop, 1995][Sarle, 2005]

- Feedforward neural network where each node **outputs** an activation function applied over the weighted sum of its **inputs**:

$$s_i = f(w_{i,0} + \sum_{j \in I} w_{ij} \times s_j)$$





Activation Functions

- Threshold

$c > 0$

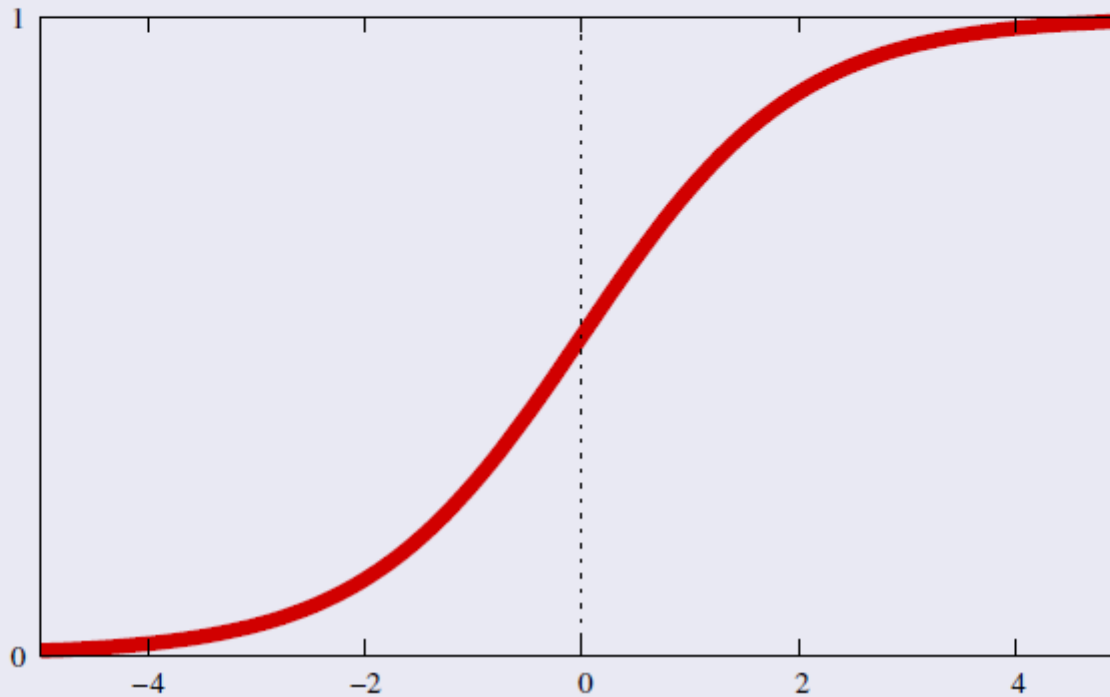
$$f(a) = \begin{cases} 1 & \text{si } a \geq \frac{1}{c} \\ 0 & \text{si } a < 0 \\ ca & \text{e.t.o.c.} \end{cases}$$

$$f'(a) = \begin{cases} 0 & \text{si } a \in (-\infty, 0) \cup (1/c, \infty) \\ c & \text{etoc} \end{cases}$$



Activation functions

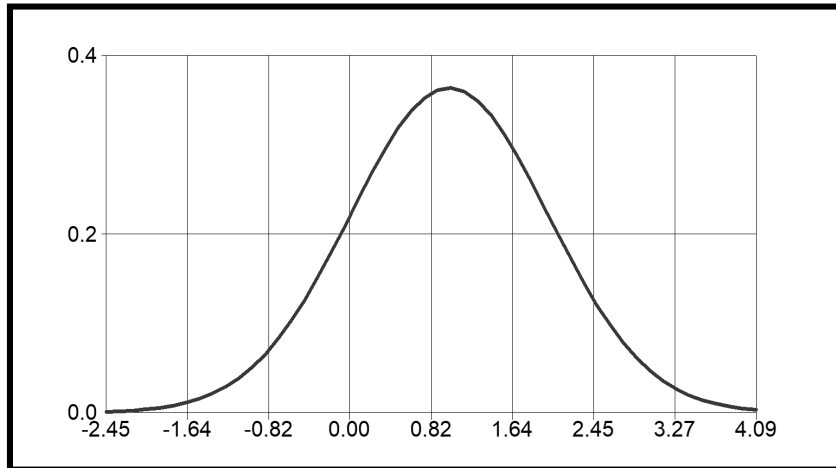
- Linear: $y = x$;
- Tanh: $y = \tanh(x)$;
- **Logistic** or Sigmoid (most used): $y = \frac{1}{1+e^{-x}}$;





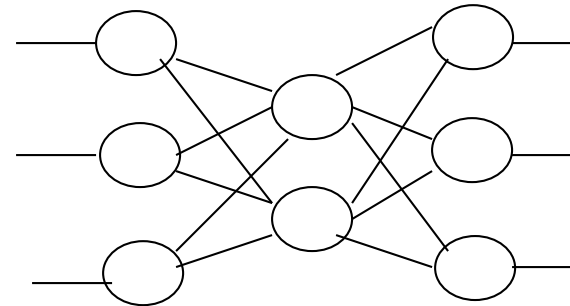
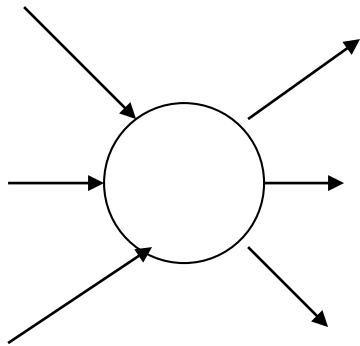
Activation Functions

Radial based





Artificial Neural Networks (ANN)



- Coding or Training Step
(fit the parameters w_i , use training set, (x_i, y_i))
- Decoding or Classification Step
- Perceptrons, Backpropagation technique and Kohonen Maps.



Training the ANN

- Find weights that minimize the residuals sum of squares

$$E_{tot}(W) \equiv \sum_{p=1}^P E(W)$$

$$E(W) \equiv \frac{1}{2} \sum_{q=1}^{N_L} [z_{L_q}(x) - t_q(x)]^2$$

z_{L_q} prediction provided in the neuron of output layer

$t_q(x)$real response value

different error functions for numerical or qualitative response

Initialize wi randomly -1, 1

Numerical optimization (local optimum) method over the space of parameters (based on the error gradient of all layers)

Termination Criterion (min improvement, nr of iterations)



Properties

- **Popularity** - the most used Neural Network, with several software tools available;
- **Universal Approximators** - general-purpose models, with a huge number of applications (e.g. classification, regression, forecasting, control or reinforcement learning);
- **Nonlinearity** - when compared to other data mining techniques (e.g. multiple regression) MLPs often present a higher predictive accuracy;
- **Robustness** - good at ignoring irrelevant inputs and noise;
- **Explanatory Knowledge** - Difficult to explain when compared with other algorithms (e.g. decision trees), but it is possible to extract knowledge from trained MLPs (e.g. if-then rules, sensitivity analysis);



- Set initial model configuration details (e.g. number of hidden layers of MLP, SVM kernel type).

Common MLP setup (e.g. R tool):

- Often, it is better to perform **one** classification/regression task per model;
- The number of input nodes is defined by the task;
- Use of one hidden layer of H nodes with logistic functions;
- **Binary classification**: one output node with logistic function;
- **Multi-class classification**: N_C output linear nodes ($f(x) = x$) and the softmax function is used to transform these outputs into class probabilities;
- **Regression**: one linear output neuron.



Gradient-descent [Riedmiller, 1994]:

- **Backpropagation (BP)** - most used, yet may be slow;
- Other algorithms: **Backpropagation with Momentum; QuickProp; RPROP; BGFS, Levenberg-Marquardt, ...**

Evolutionary Computation [Rocha et al., 2007]

- May overcome local minima problems;
- Can be applied when no gradient information is available (reinforcement learning).

KEMLG



- The MLP weights are randomly initialized within small ranges (e.g. $[-0.7; 0.7]$);
- Each training may converge to a different (local) minima;

Solutions

- Use of N_R **multiple** trainings, selecting the *MLP* with lowest error;
- Use an **ensemble** with N_R MLPs, where the final output is given as the average of the MLPs.

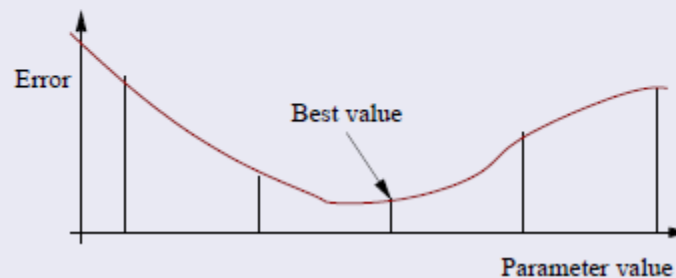




- Powerful learners (MLP, SVM) have several hyperparameters that need to be set/tuned;
- Such parameters can be set using: heuristic rules, simple grid-search or more advanced optimization algorithms (e.g. Evolutionary Computation) [Rocha et al., 2007];

Grid-Search

- One (or more) parameters are scanned through a given range;
- Range example for MLP hidden nodes: $H \in \{0, 2, 4, \dots, 20\}$;
- Variants: two-level greedy grid-search (search at the first level and then a second pass is taken, using a smaller range and step).



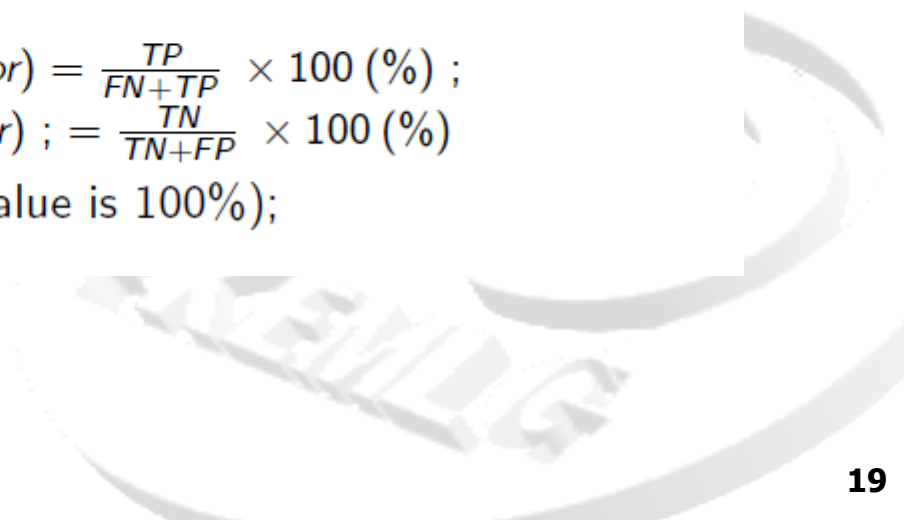


Confusion matrix [Kohavi and Provost, 1998]

- Matches the **predicted** and **actual** values;
- The 2×2 *confusion matrix*:

↓ actual \ predicted →	negative	positive
negative	TN	FP
positive	FN	TP

- Three accuracy measures can be defined:
 - the **Accuracy** = $\frac{TN+TP}{TN+FP+FN+TP} \times 100$ (%) (use if FP/FN costs are equal);
 - TPR or **Sensitivity** (*Type II Error*) = $\frac{TP}{FN+TP} \times 100$ (%) ;
 - TNR or **Specificity** (*Type I Error*) ; = $\frac{TN}{TN+FP} \times 100$ (%)
- The higher, the better (the ideal value is 100%);





- Multi-class confusion matrix example [Cortez et al., 2009]:

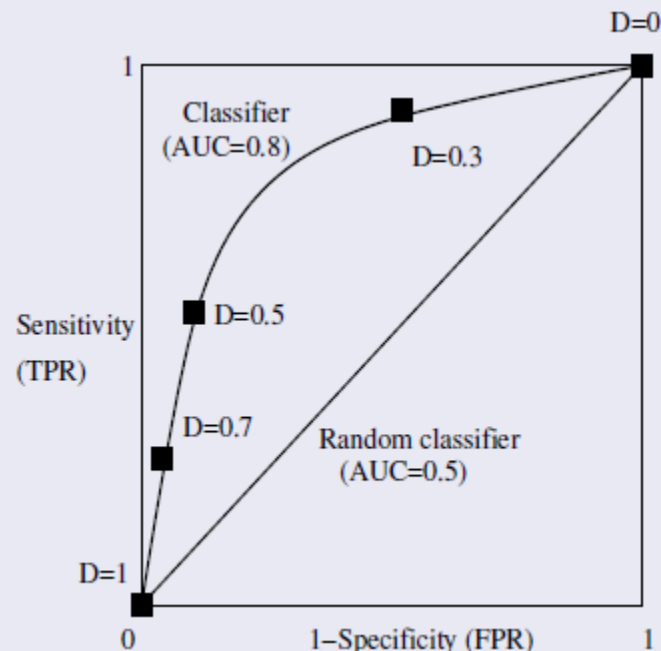
Actual class	Red wine predictions					White wine predictions				
	4	5	6	7	8	4	5	6	7	8
3	1	7	2	0	0	0	2	17	0	0
4	1	36	15	1	0	19	55	88	1	0
5	3	514	159	5	0	7	833	598	19	0
6	0	194	400	44	0	4	235	1812	144	3
7	0	10	107	82	1	0	18	414	441	7
8	0	0	10	8	0	0	3	71	43	59
9						0	1	3	2	0

KEMLIG



Receiver Operating Characteristic (ROC) [Fawcett, 2006]

- Shows the behavior of a 2 class classifier ($y \in [0, 1]$) when varying a decision parameter $D \in [0, 1]$ (e.g. True if $y > 0.5$, $D = 0.5$);
- The curve plots $FPR = 1 - TNR$ (x-axis) vs TPR (Sensitivity);
- Global performance measured by the **Area Under the Curve (AUC)**:
 $AUC = \int_0^1 ROC dD$ (the perfect AUC value is 1.0).





Given a dataset with the function pairs $x_1 \rightarrow y_1, \dots, x_N \rightarrow y_N$, we can compute:

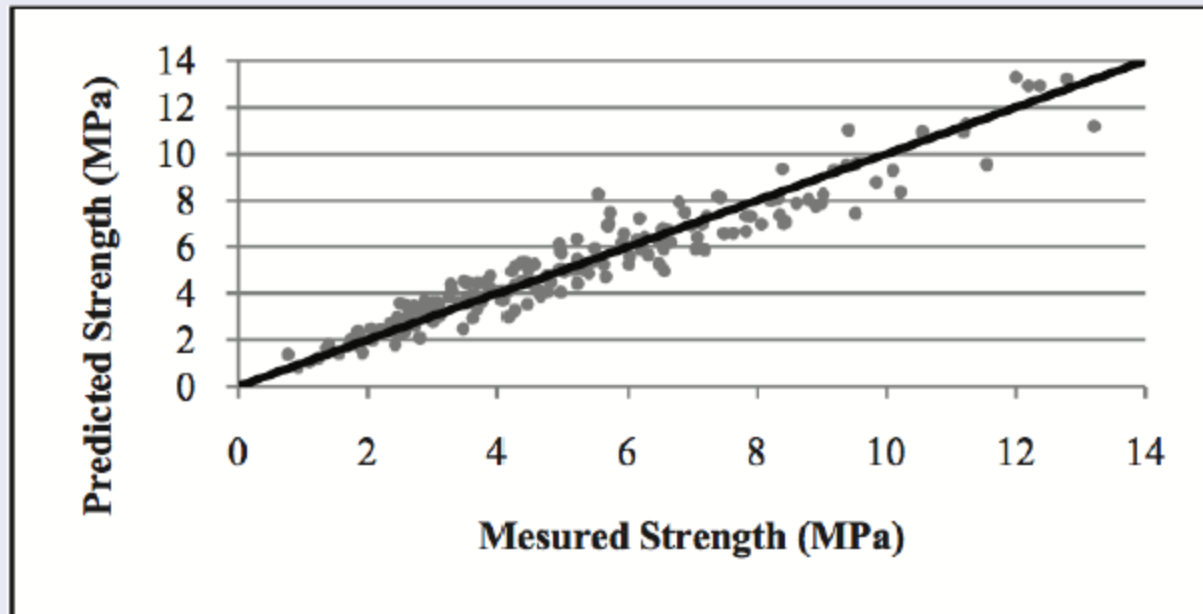
Error metrics

- **Mean Absolute Error/Deviation (MAE):** $MAE = \frac{\sum_{i=1}^N |e_i|}{N}$
- **Sum Squared Error (SSE):** $SSE = \sum_{i=1}^N e_i^2$
- **Mean Squared Error (MSE):** $MSE = \frac{SSE}{N}$
- **Root Mean Squared Error (RMSE):** $RMSE = \sqrt{MSE}$
- **Relative Absolute Error (RAE, scale independent):**
 $RAE = MAE / MAE_{\text{baseline}} \times 100 (\%)$, where baseline often denotes the average predictor.
- **Relative Squared Error (RSE, scale independent):**
 $RSE = SSE / SSE_{\text{baseline}} \times 100 (\%)$
- The lower, the better (ideal value is 0).



Scatter plot: desired (x -axis) vs predicted (y -axis) values

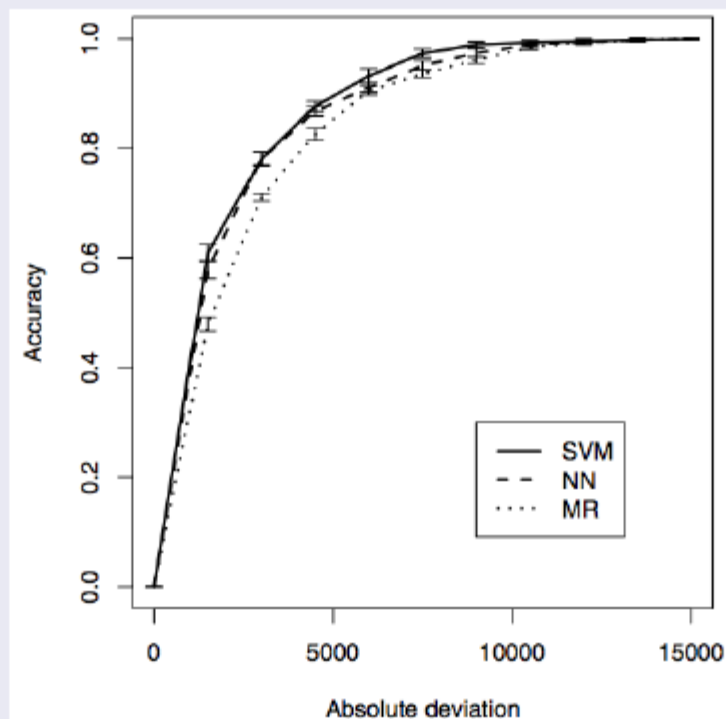
- Used to assess the prediction quality of a regression model;
- The perfect fit is the diagonal line;
- Civil engineering application example [Tinoco et al., 2009]:





Regression Error Characteristic (REC) curves [Bi and Bennett, 2003]

- Used to compare several regression models;
- The curve plots the error tolerance (absolute deviation, x-axis) versus the percentage of points predicted within the tolerance (y-axis);



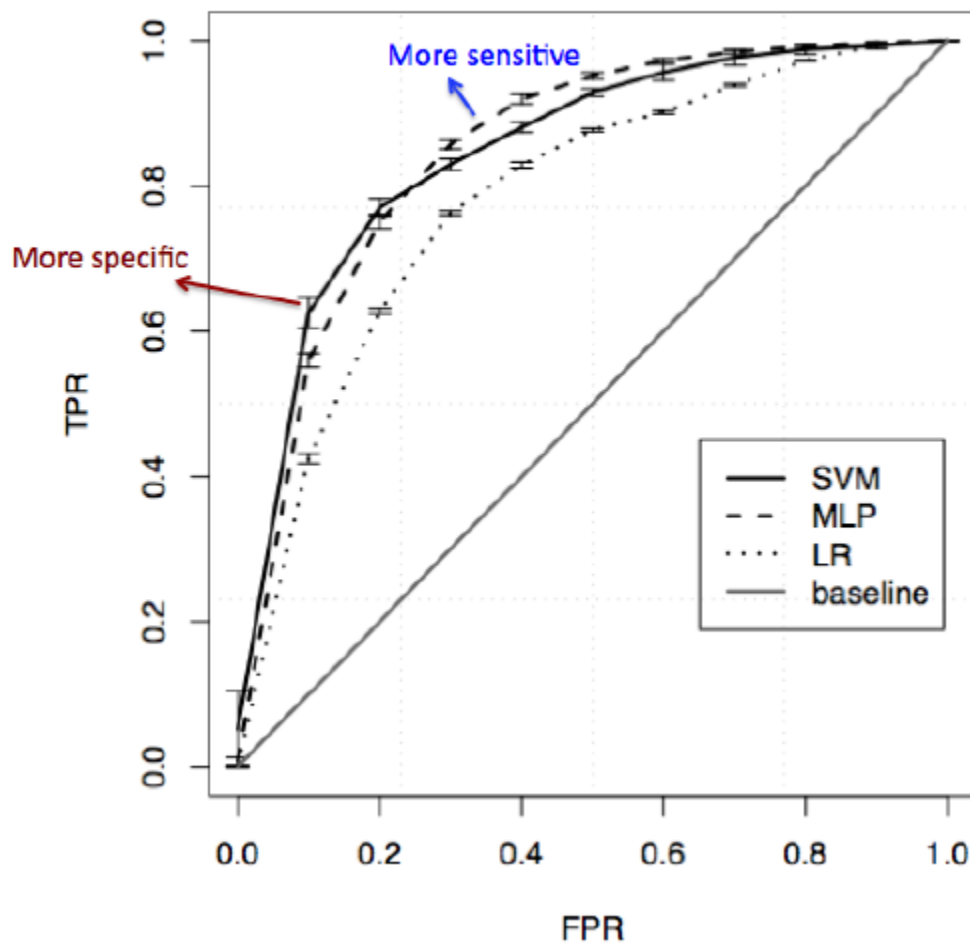


- The aim is to assess if the DM model meets the **business goals** and if it is **interesting**.
- **Interestingness**: does the model makes sense to the domain experts and unveils useful or challenging information?
- **Business impact**: by using such model, what is the gain achieved?

KEM/UPC

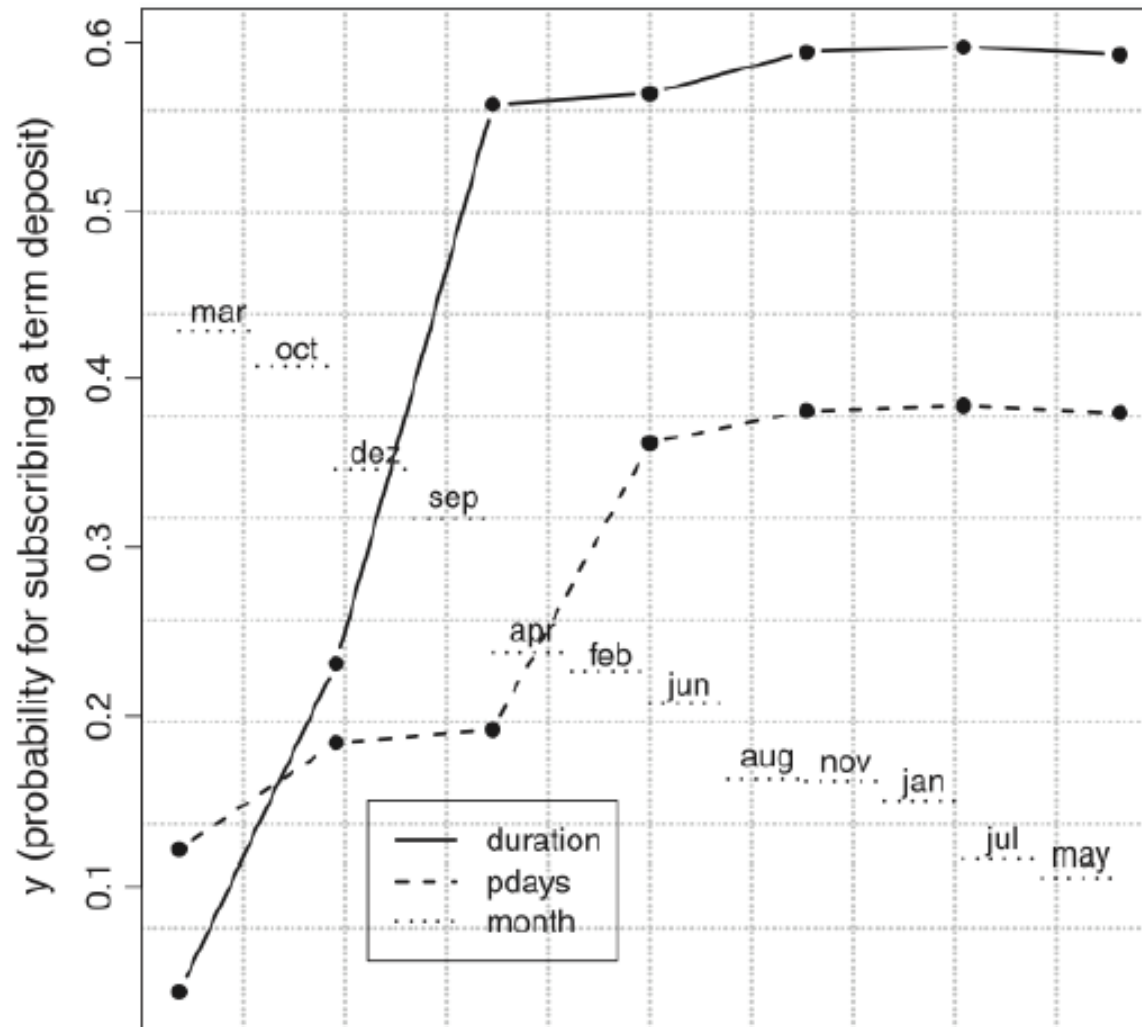


- Business impact example: ROC Curve and benefit-cost analysis (FPR-TPR trade-off).



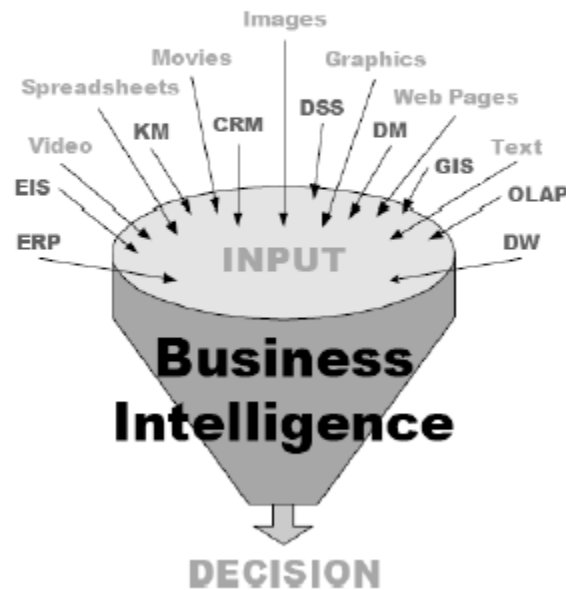


Visualization of Input Effect: VEC curve (Bank Marketing)





- BI systems often use DM (for knowledge extraction and **prediction**).

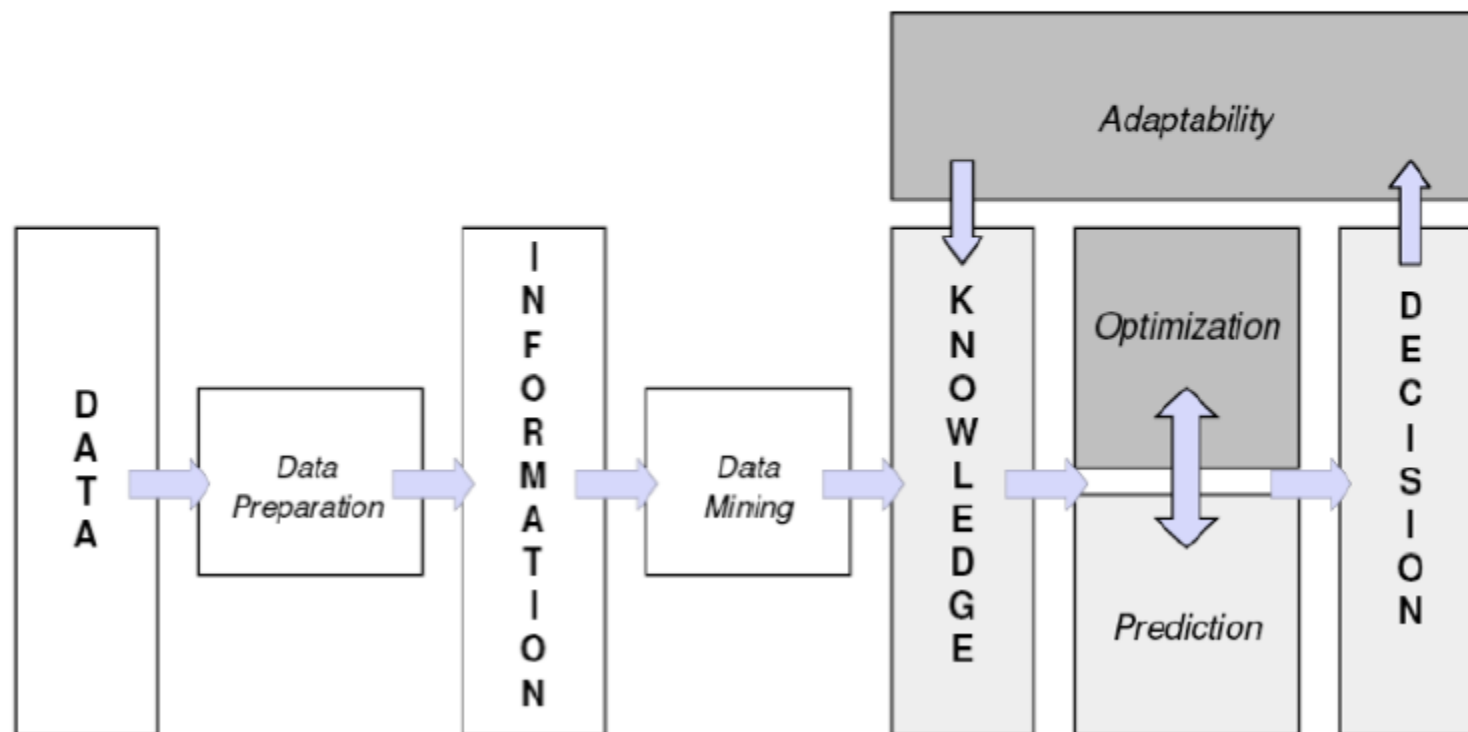


KEMLIG



Adaptive Business Intelligence (ABI)

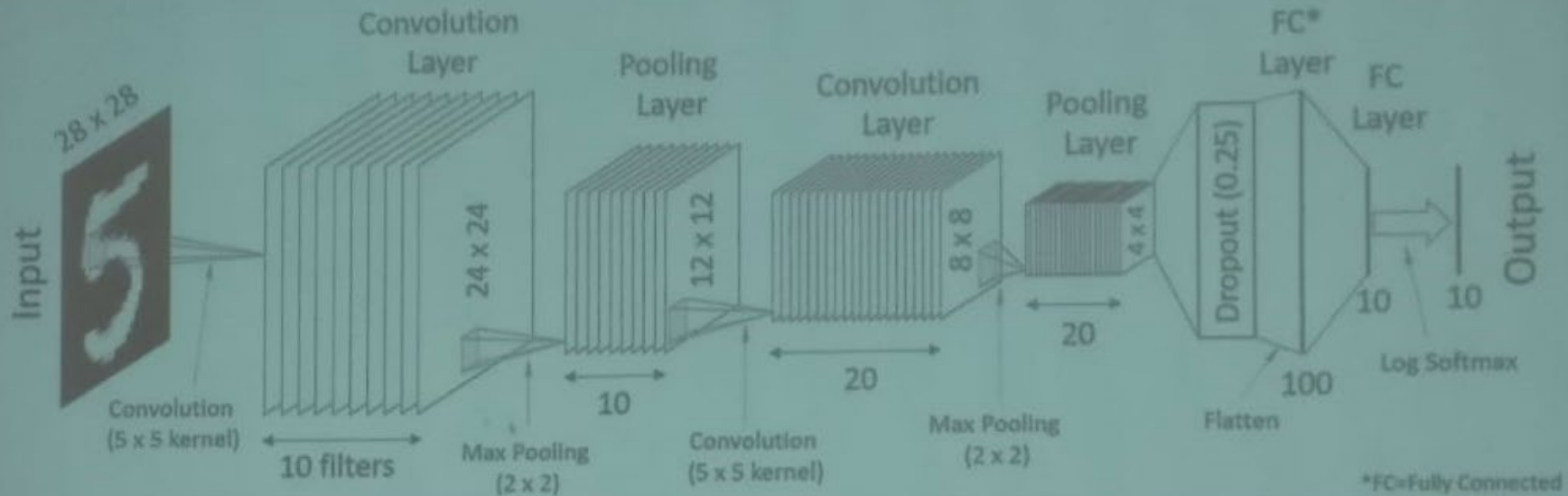
[Michalewicz et al., 2006]





Deep learning

Convolutional Neural Network



Challenge 2: build a CNN with >99% accuracy