Classification Techniques

JIGSAW ACADEMY

Analytics for Professionals

Classification Problems

- Which of these are malignant?

- Will this guy buy my product/service?

- Will this guy default on loan repayment?

- Are these transactions fraudulent?

- Is this fever malaria, dengue or typhoid?

- Which employees may chrun this qtr?

Classification Problems

- To this website visitor, should I show a travel, mortgage or a savings ad?

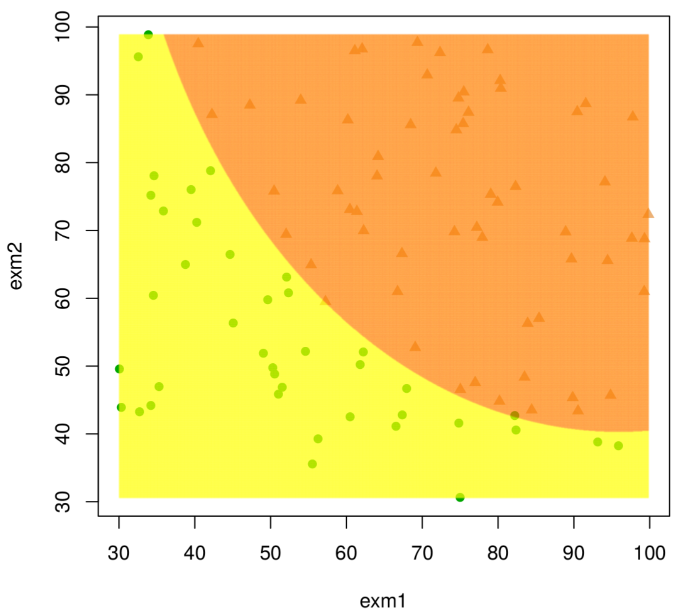
- Handwritten digits recognition

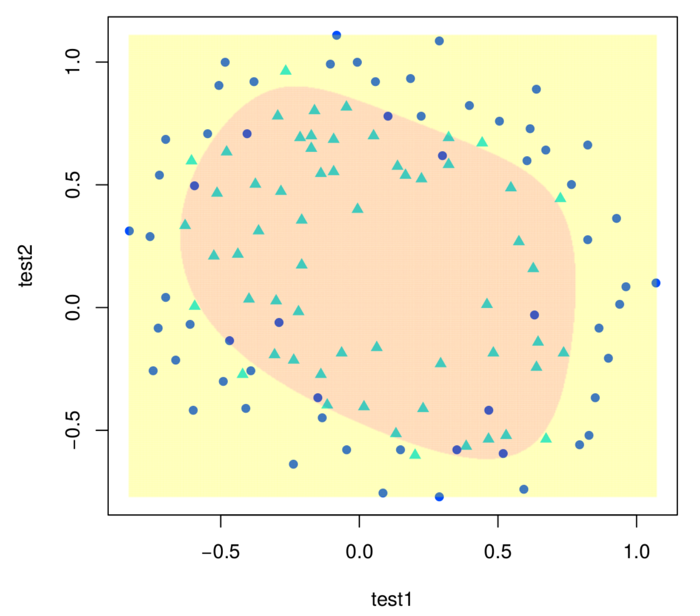
Data characteristics

- Observations have been (and will be) made on independent ‘sampling units’, ‘events’
  - cross-sectional data; not a panel or a timeseries data
- The dependent variable is categorical (obvious!)
  - some classes are rare (luckily, fraud, cancer)
- Independent, observed variables may be
  - categorical or continuous,
  - well behaved or noisy,
  - independent or inter-correlated,
  - linearly or non-linearly related to dependent
  - some independents strongly discriminate
  - underlying distribution of independents may be nicely behaved
  - residuals after fitting a model are not nicely behaved

Classification techniques

Objective: Classify a given observation into two or more classes.





- naïve Bayes
- k-nearest neighbours
- logistic regression
- decision trees
- support vector machines
- artificial neural networks

Bayes' Theorem

Bayes' Theorem, famously reads:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(A)P(B|A)}{P(B)}$$

where A and B are some two events.

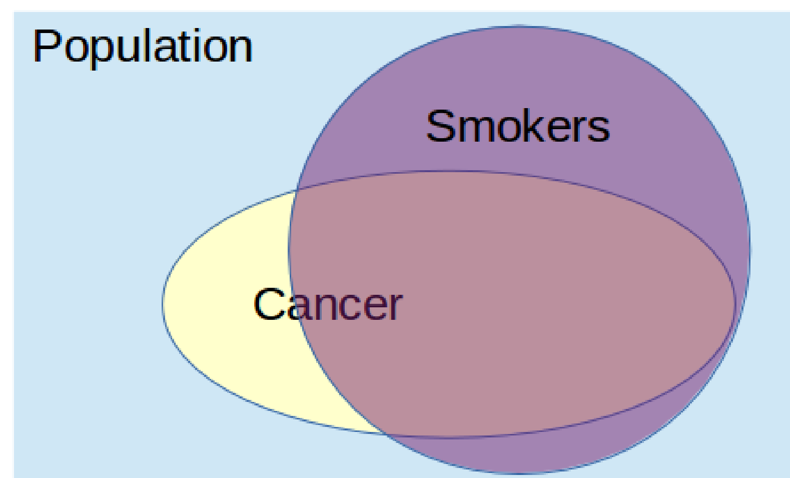
Better presented: Is it possible to improve our estimate of θ, cancer risk, if we know an easily observable condition X smoking?

$$P(\theta|X) = \frac{P(X \cap \theta)}{P(X)} = \frac{P(\theta)P(X|\theta)}{P(X)}$$

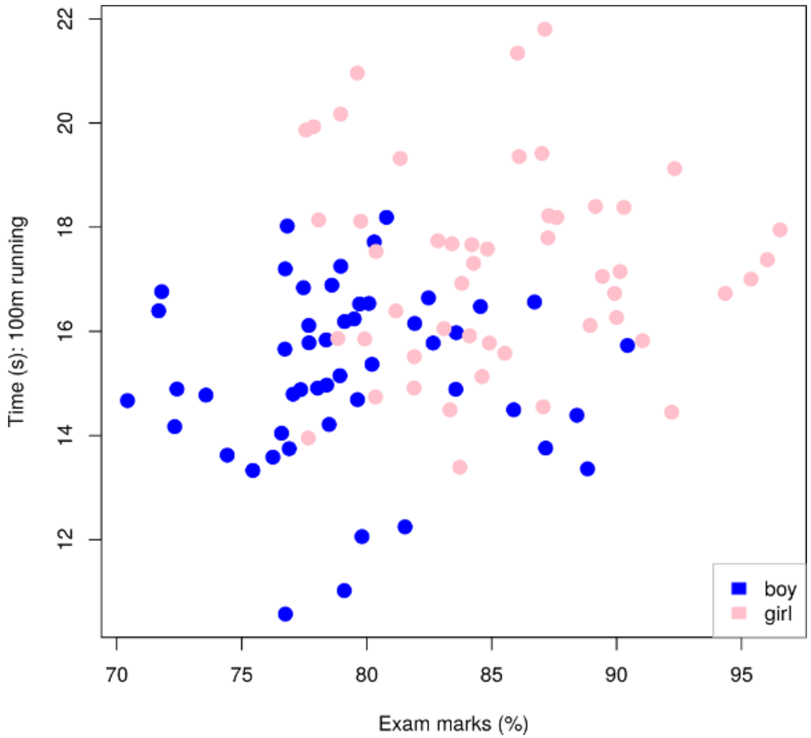
$P(X|\theta)$: Prior probability
$P(\theta|X)$: Posterior probability

This, in the historic data means

$$\frac{(\text{\# of smokers with cancer})}{(\text{\# smokers})}$$
$$= \frac{(\text{Overall risk of cancer})}{(\text{Overall smoking incidence})}$$
$$\times\ (\text{Proportion of smokers among cancer patients})$$



Naïve Bayes



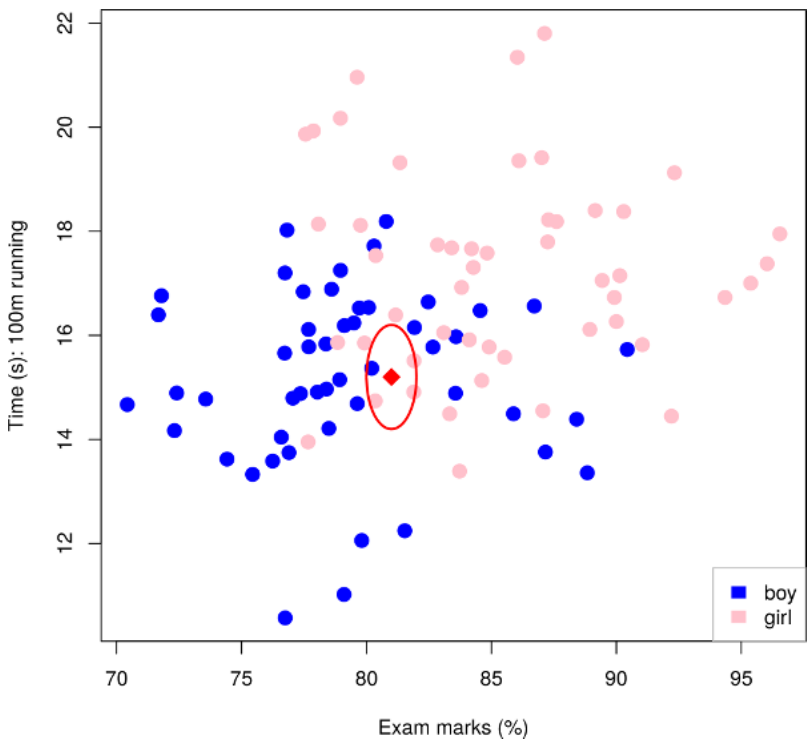
Consider a problem of identifying sex of a student based on

- Marks in exams (X_1)
- Timing in 100 m race (X_2)

Naïve Bayes algorithm

- If a prior probability of sex in the class is known, it is noted, otherwise proportions in the training set are used.
- Using training set, calculates the distributions of X_1 & X_2 independently within each class boy & girl.
- In the test set, given the observed marks and timing of a student, it calculates posterior probability of being either a boy or a girl.
- Highest probability class wins.

k-Nearest Neighbours



Consider a student's marks & timing (X_1, X_2): (81, 15)

k-NN algorithm

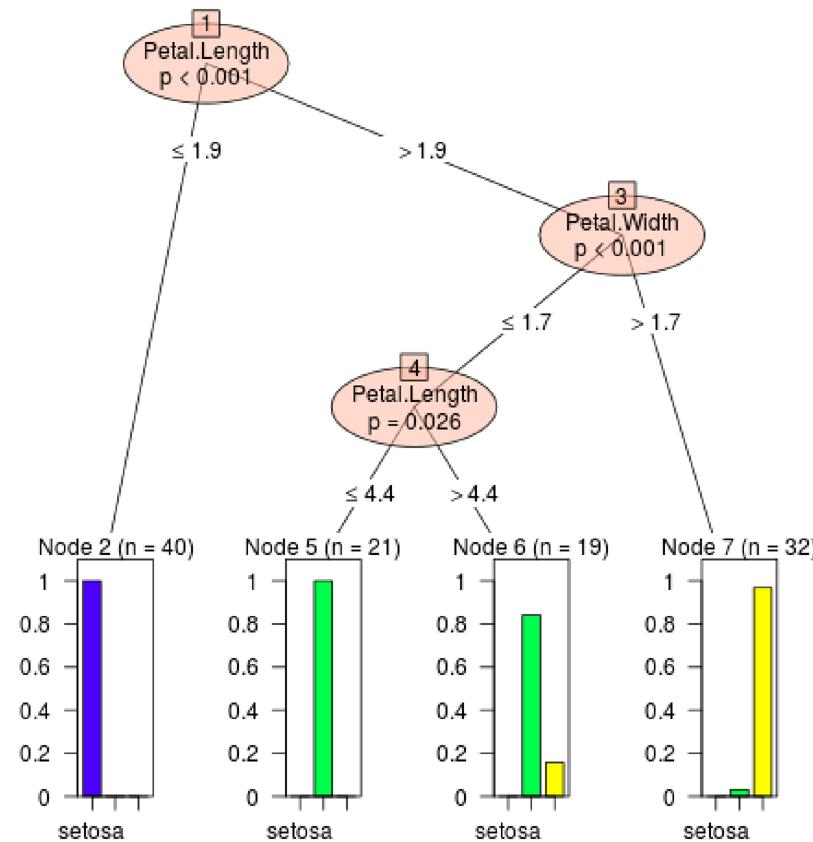
- finds the *k* nearest neighbours of this student in terms of (X_1, X_2)
- Class of this student = majority class of its *k* neighbours.
- Rules for breaking ties required.
- Choice of appropriate *k* is usually based on m-fold cross validation.
- 'nearest' $\Rightarrow$ 'Distance measure' – Minkowski p-norm
- Scaling of Xs matters!
- Search for neighbours for each new test case is a scan across all data points – expensive

Decision Trees

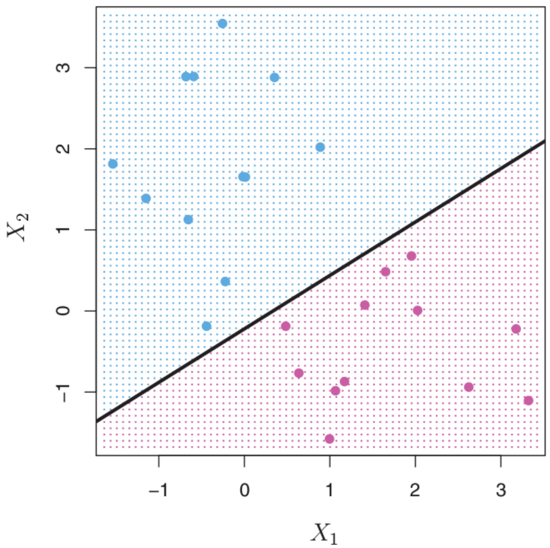
The algorithm is

- Greedy, top-down, divide-and-conquor
  - test every variable for a split
  - choose the one that gives highest increase in gini/entropy
  - send down subsets to child nodes and repeat.
- Typically single variable based splitting
- Binary trees
- handles noisy and missing data situations well
- Easy to convey, use, visualize

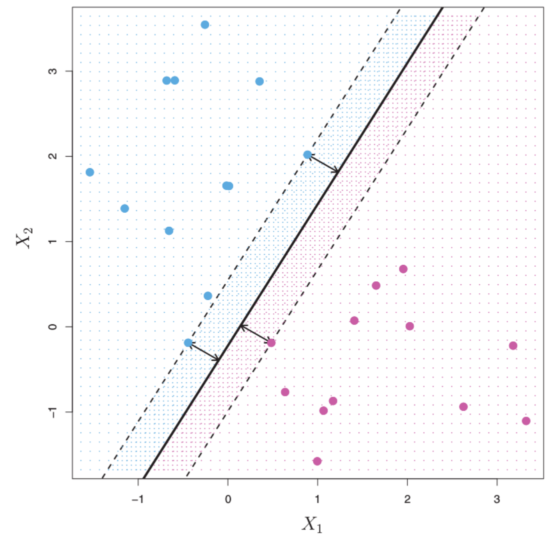
- Innovations: Random Forest, Boosting



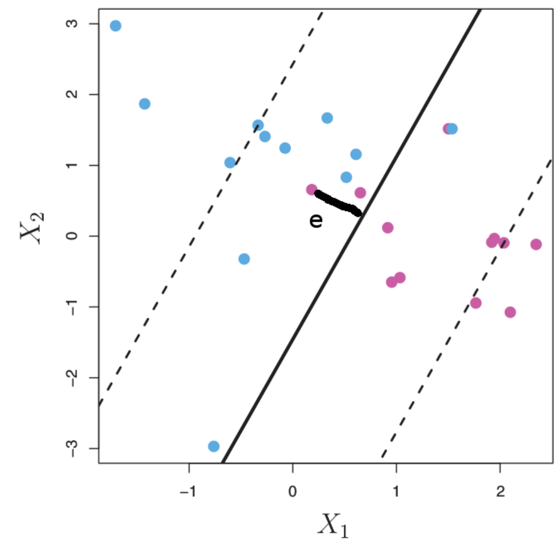
Support Vector Machines: Linear



Separating Hyperplane



Maximal Margin Classifier



Support Vector Classifier

If we assign the purple points $y_i = -1$ and the blue points $y_i = 1$ Then the separating hyperplanes satisfy

$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) > 0 \quad \forall i \in 1, \ldots, n$$

To get a plane 'farthest' from all points:

$$\max_{\beta_0 \ldots \beta_p} M$$

subject to $\sum_{i=1}^{p} \beta_i^2 = 1$

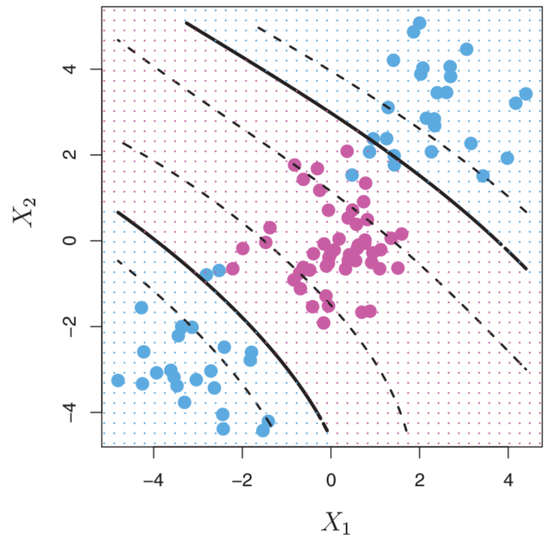
$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq M, \quad i = 1, \ldots, n$$

When some points cross over, the last constraint is modified to:

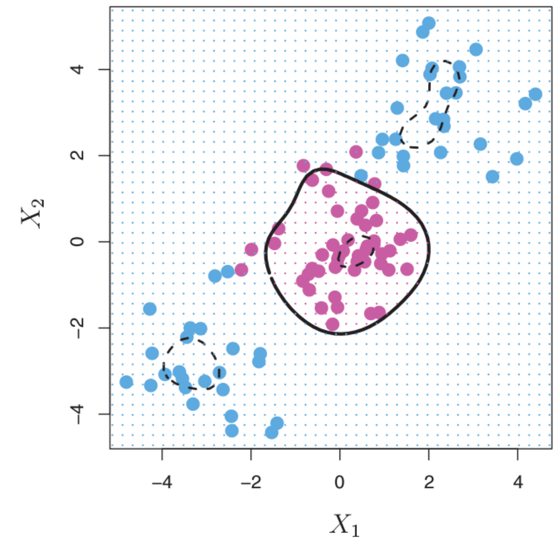
$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i), \quad i = 1, \ldots, n$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C, \quad C \geq 0$$

Support Vector Machines: Kernels



Polynomial Kernel



Radial Kernel

Polynomial Kernel of degree d

$$K(x_i, x_j) = (1 + \sum_{k=1}^{p} x_{ik}x_{jk})^d$$

$d > 1$, will have a solution

$$f(x^*) = \beta_0 + \sum_{i \in S} \alpha_i K(x_i, x^*)$$

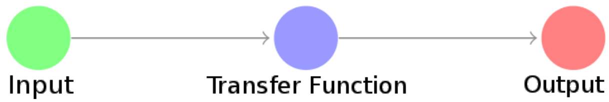
Radial Kernel takes the form

$$K(x_i, x_j) = exp(-\gamma \sum_{k=1}^{p} (x_{ik} - x_{jk})^2)$$

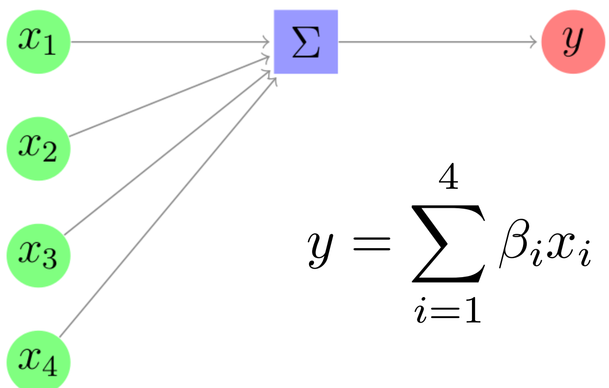
$$\gamma > 0$$

Graphical representation of models

- Model as a data flow:

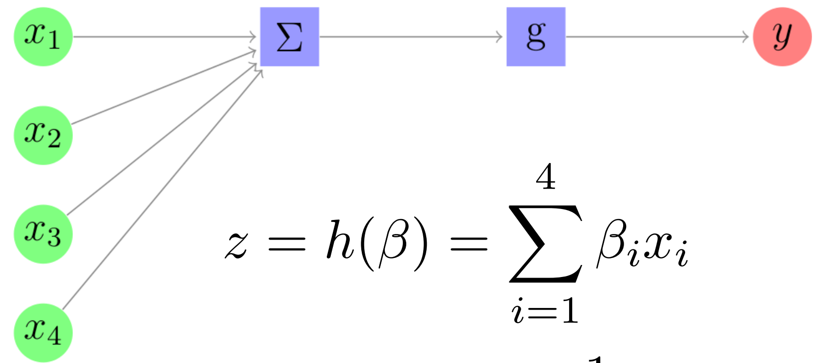


- Linear Regression as data flow:



$$y = \sum_{i=1}^{4} \beta_i x_i$$

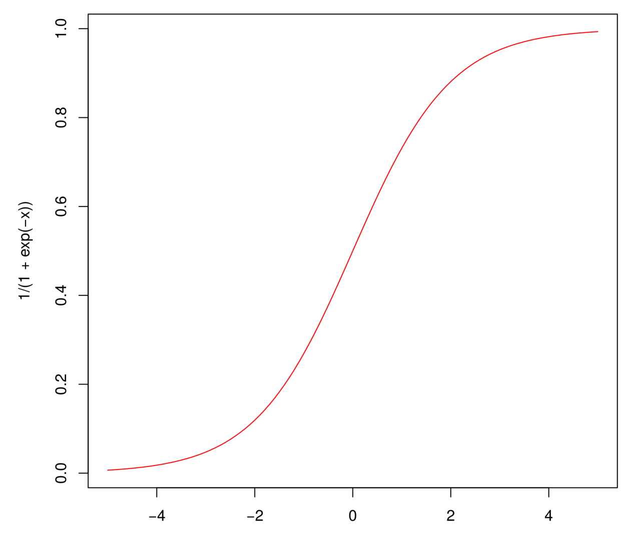
- Logistic Regression as data flow:



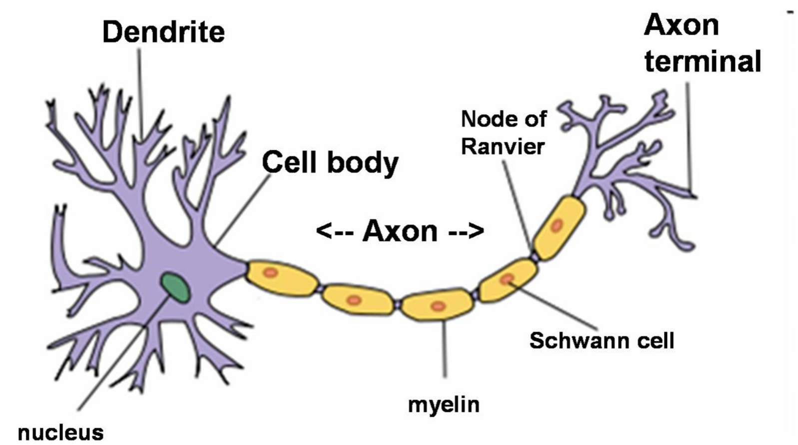
$$z = h(\beta) = \sum_{i=1}^{4} \beta_i x_i$$

$$g(z) = \frac{1}{1 + \mathrm{e}^{-z}}$$

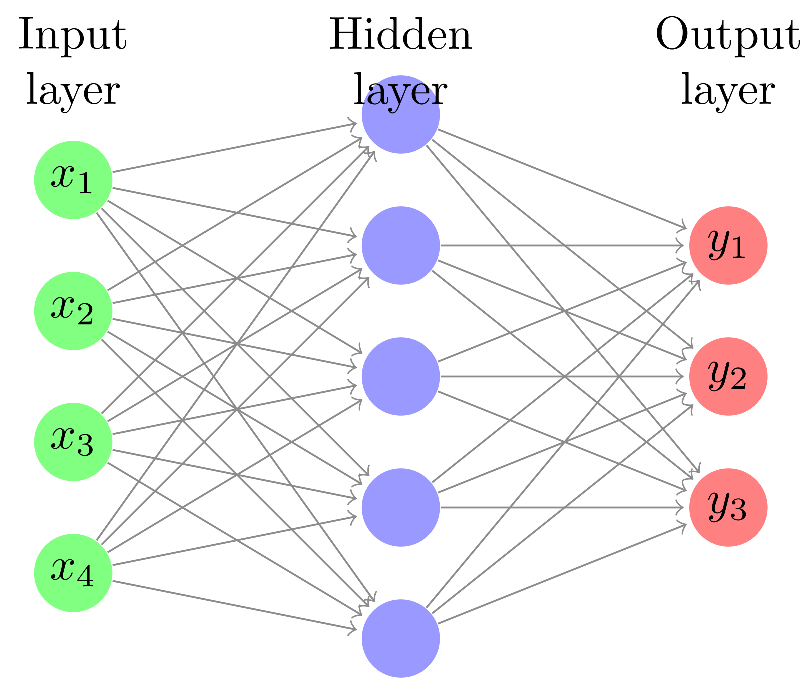
- Sigmoid transfer function



Artificial Neural Network: Inspiration

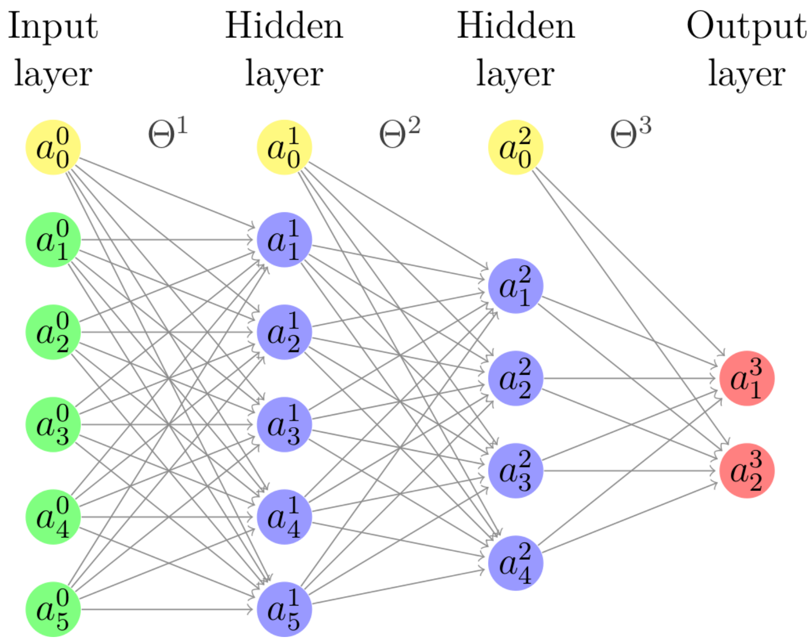


A computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs.



- Artificial Intelligence & cognition
- Inspired by biological neural networks – CNS
- ‘Sensory inputs’ ‘activate the neurons’ and the ‘activations’ are transmitted across the network until finally, ‘output neuron’ is activated.

Artificial Neural Network



- Input layer & nodes
- Output layer & nodes
- Optional Hidden layer(s) & nodes

- Transfer/Activation functions

$$\text{Sigmoid}: \quad \frac{1}{1 + \mathrm{e}^{-\sum \Theta_{ij} x_j^{(i)}}}$$

$$\text{Tanh}: \quad \log \frac{1 + \sum \Theta_{ij} x_j^{(i)}}{1 - \sum \Theta_{ij} x_j^{(i)}}$$

- Back propagation algorithm
  - forward activation of output
  - back propagation of errors
  - learning rate

R implementation notes

- naïve Bayes – `e1071::naiveBayes`, `klaR::NaiveBayes`
- k-nearest neighbours – `class::knn`, `kknn::kknn`
- logistic regression – `stats::glm`, `glm2::glm2`, `rms::lrm`, `VGAM::vglm`
- decision trees – `rpart::rpart`, `randomForest::randomForest`
- support vector machines – `e1071::svm`
- neural networks – `nnet::nnet`, `neuralnet::neuralnet`