Machine Learning

- computer algorithms
- applied to data to generate info
- volume, velocity, variety
- e.g. risk classification of motor insurance policyholders via in-car monitoring devices
- e.g. detection of possible fraudulent insurance claims
- e.g. cause codes of insurance claims
- e.g. chatbots for customer inquiries
- e.g. targeted ads on websites

Machine Learning

- $t = f(x_1, x_2, x_3, \ldots)$

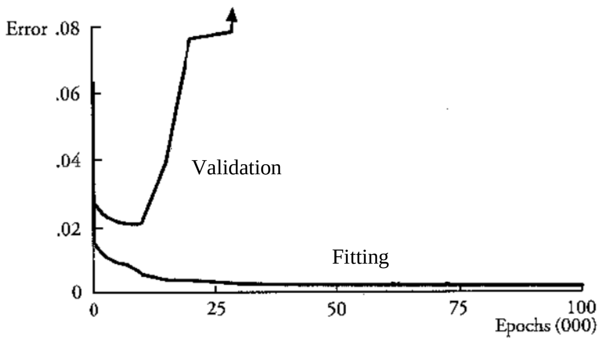
  t : target value

  x_i : ith features / covariates / explanatory variables

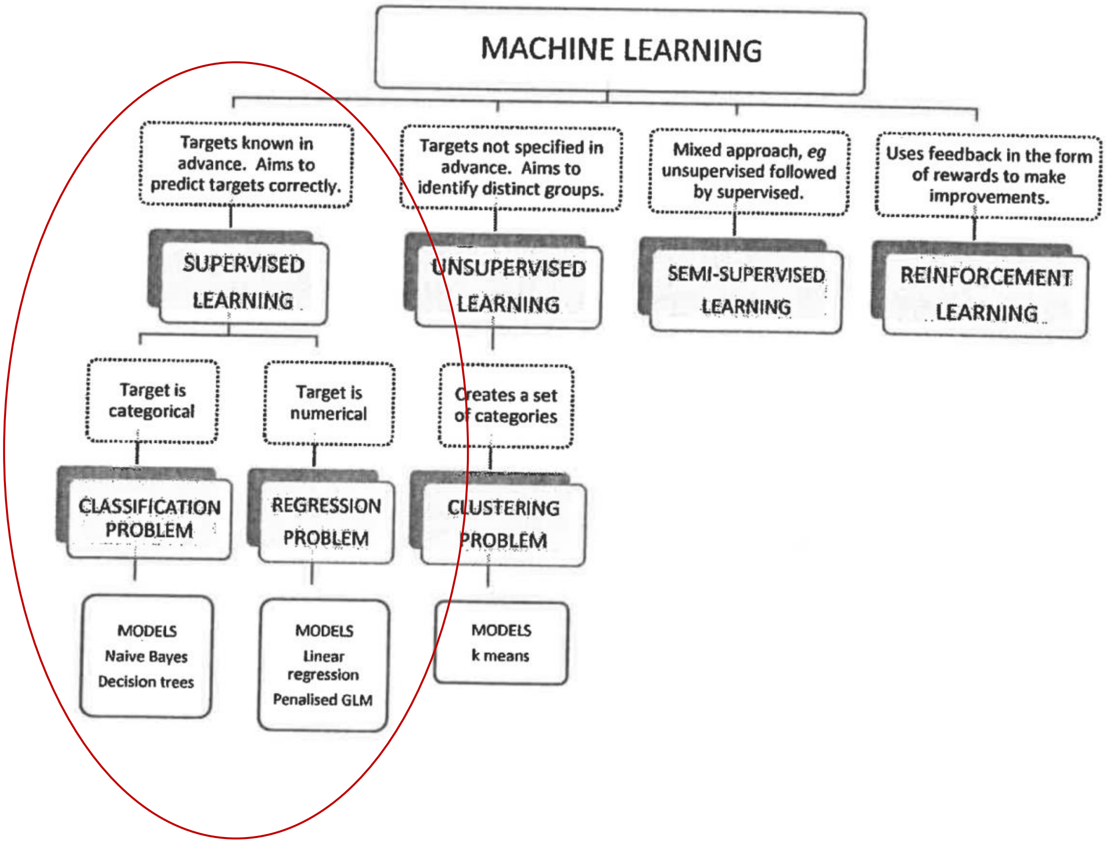
  f : target function
- approximation of unknown f
- mapping function which mimics f
- estimation of weights / coefficients / parameters via minimisation of error function
- iterative estimation procedure
- regression as a specific example
- machine learning on forecast performance
- statisticians on parameter significance

Training / Validation / Testing

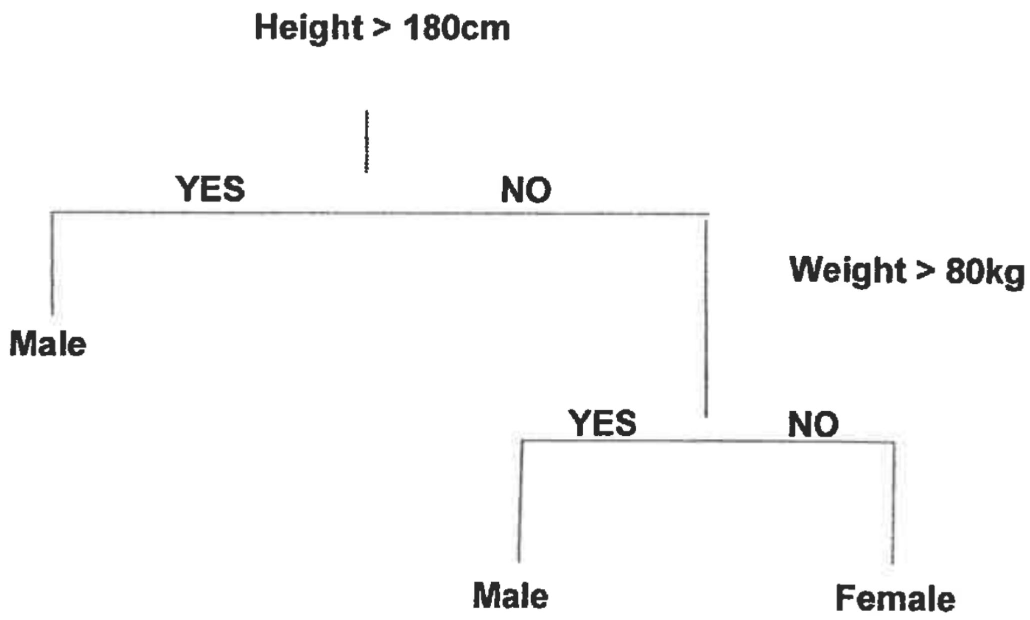
- splitting data set into *training set* and *testing set*
- further splitting training set into *fitting set* and *validation set*
- using fitting set to estimate parameters
- using validation set to adjust parameters and avoid overfitting
- AIC, BIC, L1 / L2 penalty as alternative
- using testing set to provide unbiased evaluation of forecast performance
- rule-of-thumb : 60% / 20% / 20%

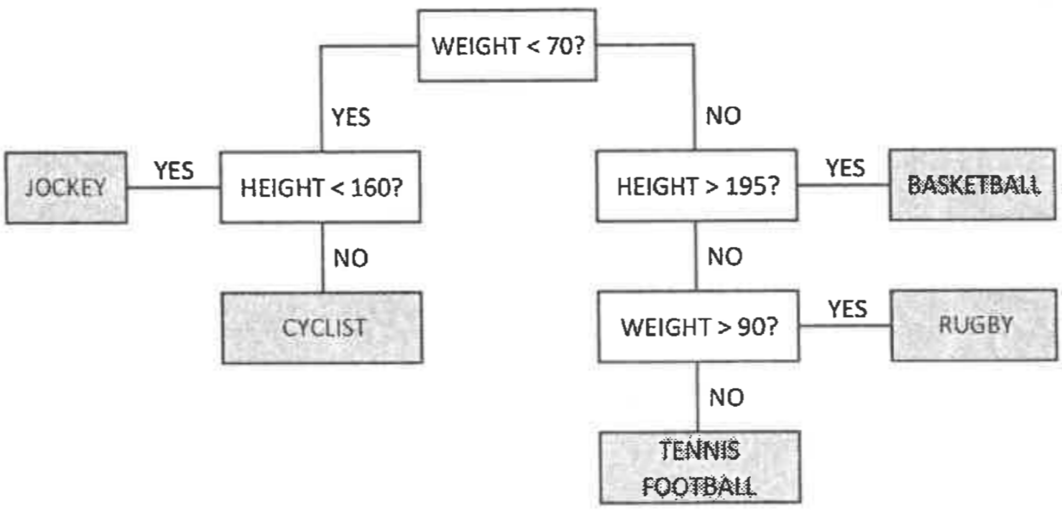


Types of Machine Learning

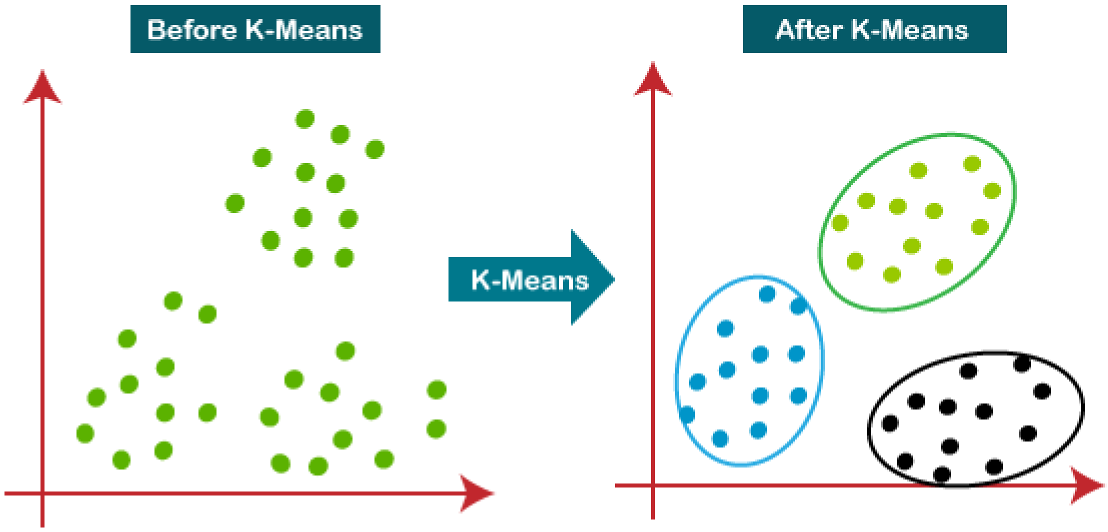


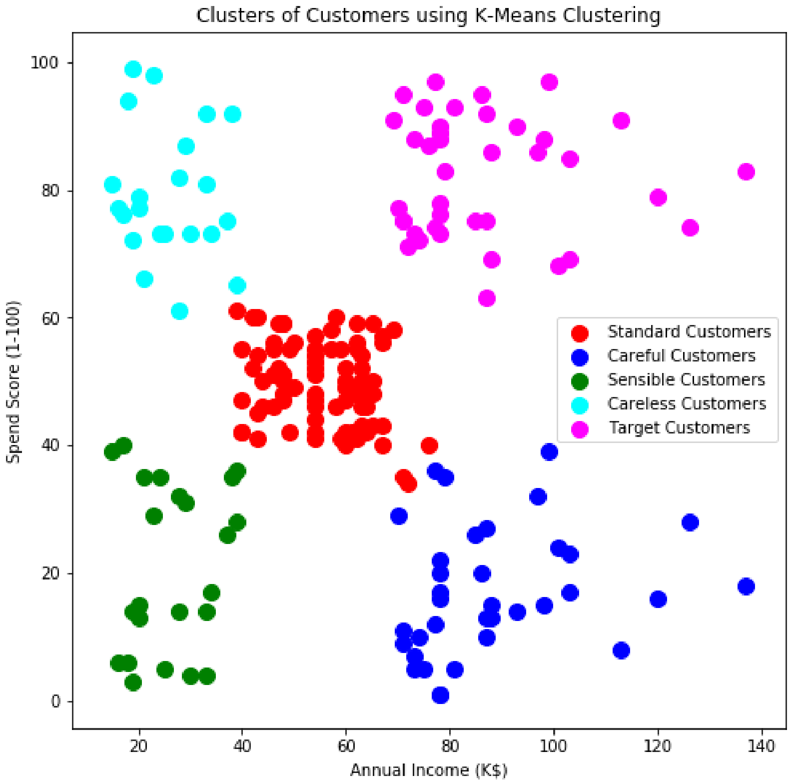
Decision Tree





K-Means Clustering





Types of Traditional Data

DATA TYPES				
NUMERICAL (*ie* numbers)		CATEGORICAL (*ie* not numbers)		
DISCRETE	CONTINUOUS	ATTRIBUTE (DICHOTOMOUS)	NOMINAL	ORDINAL
↓	↓	↓	↓	↓
Age last birthday Number of children Number of claims	Exact age Salary Claim amount	Alive / Dead Male / Female Claim / No claim Pass / Fail	Customer name Type of claim Occupation Marital status Country Colour of car	Date of birth (DD/MM/YY) Month (Jan, Feb, Mar, …) Exam grade (A, B, C, …) Size (S, M, L, XL) Agree/Don't know/Disagree

Data Preparation

– surveys, censuses, admin systems, specific databases, etc
– spreadsheet, Access, R, Python, etc
– data cleaning (errors, missing values, etc)
– exploratory data analysis
– scaling of features
– features engineering
– detailed documentation

Neural Network (NN)

– 3 layers of neurons in general
– *input* layer : features / covariates / explanatory variables
– *hidden* layer : most of learning
– *output* layer : output values
– each neuron receiving input from some neurons, and firing output to other neurons
– strengths of connections (weights / coefficients / parameters) between neurons changing epoch by epoch
– learning based on both architecture and weights of network

Neural Network (NN)

- universal approximators
- supervised learning
- FNN, RNN, LSTM, GRU, etc
- 1-layer feedforward neural network :

  $y_j = f\,(a_{0,j} + a_{1,j}\, x)$

  $z = f\,(b_0 + \Sigma_j\, b_j\, y_j)$

  x : feature value from input layer into hidden layer

  y_j : intermediate value from jth neuron in (single) hidden layer into output layer

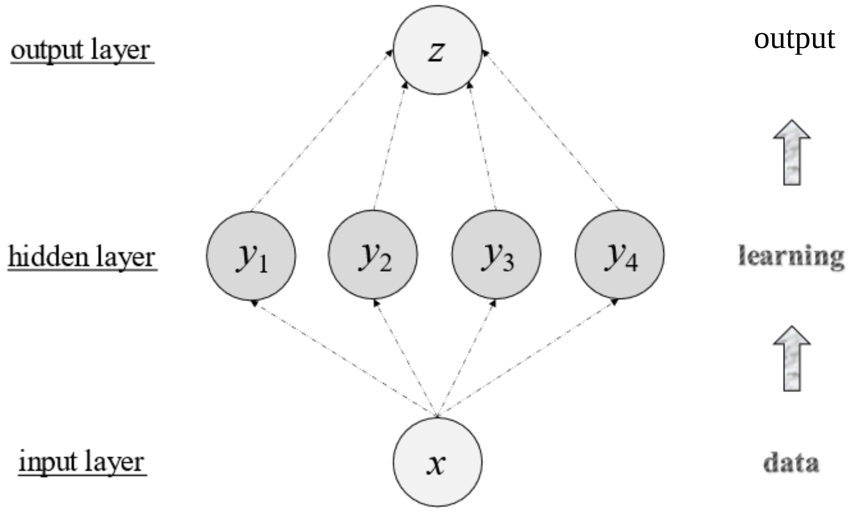
  z : output value from output layer

  f : activation function (e.g. logistic)

  $a_{0,j}$, $a_{1,j}$, b_0 , b_j : weights

Feedforward Neural Network (FNN)

– 1-layer FNN :



Feedforward Neural Network (FNN)

– network too small → inadequate modelling capacity
– network too large → overfitting and poor generalisation
– 1 or 2 hidden layers adequate for most commercial and financial applications
– rule of thumb : number of training samples around 10 times number of weights
– weights estimated by backpropagation

Backpropagation

– error function $e = 0.5\, \Sigma_k (z_k - t_k)^2 / n$

– x_k , t_k normalised to small values

– initial values of weights in hidden layer randomised as Uniform(–1, 1)

– initial values of weights in output layer randomised as –1 or 1 with equal chance

– in each epoch, for each weight :

$\varepsilon^* = \varepsilon + \lambda$ if $g \frac{\partial}{\partial w} e > 0$ [update

$\varepsilon^* = \varepsilon \, \phi$ otherwise learning rate]

$w^* = w - \varepsilon \frac{\partial}{\partial w} e$ [update weights]

$g^* = \theta g + (1-\theta) \frac{\partial}{\partial w} e$ [accumulate info]

$\lambda = 0.1$ $\phi = 0.5$ $\theta = 0.7$

Backpropagation

$$f(s)=\frac{\exp(s)}{1+\exp(s)}$$

$$\frac{\partial}{\partial s}f(s)=\frac{\exp(s)}{(1+\exp(s))^2}=f(s)(1-f(s))$$

$$\frac{\partial e}{\partial b_0}=\frac{1}{n}\sum_k(z_k-t_k)\frac{\partial z_k}{\partial b_0}=\frac{1}{n}\sum_k(z_k-t_k)z_k(1-z_k)$$

$$\frac{\partial e}{\partial b_j}=\frac{1}{n}\sum_k(z_k-t_k)\frac{\partial z_k}{\partial b_j}=\frac{1}{n}\sum_k(z_k-t_k)z_k(1-z_k)y_{j,k}$$

$$\frac{\partial e}{\partial a_{0,j}}=\frac{1}{n}\sum_k(z_k-t_k)\frac{\partial z_k}{\partial y_{j,k}}\frac{\partial y_{j,k}}{\partial a_{0,j}}$$

$$=\frac{1}{n}\sum_k(z_k-t_k)z_k(1-z_k)b_j y_{j,k}(1-y_{j,k})$$

$$\frac{\partial e}{\partial a_{1,j}}=\frac{1}{n}\sum_k(z_k-t_k)\frac{\partial z_k}{\partial y_{j,k}}\frac{\partial y_{j,k}}{\partial a_{1,j}}$$

$$=\frac{1}{n}\sum_k(z_k-t_k)z_k(1-z_k)b_j y_{j,k}(1-y_{j,k})x_k$$

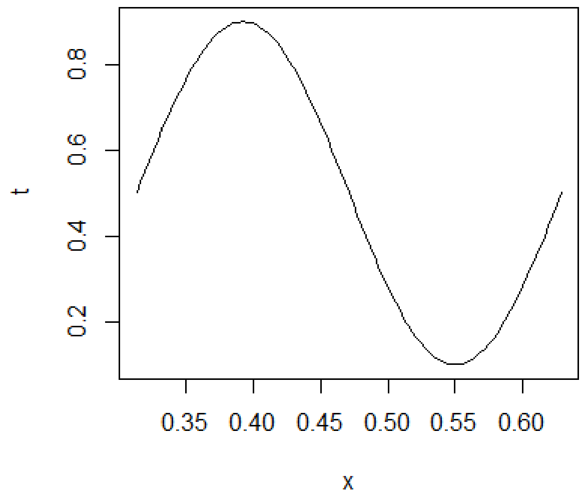
Example

x=seq(0.1,0.2,0.001)*pi

obs=sin(x*20)

t=(obs-(-1))/(1-(-1))*(0.9-0.1)+0.1

(Note: It is a toy problem without noises.)



n=101; hids=3; epoch=10000

y<-array(NA,c(hids,n))

z<-numeric()

Example

```
logistic<-function(s) {
if (s>700) { logistic=1 }
if (s<=700) { logistic=exp(s)/(1+exp(s)) }
logistic }

e<-numeric()
a<-array(NA,c(2,hids))
for (i in 1:2) { for (j in 1:hids) {
a[i,j]=runif(1,-1,1) } }
b=-1+rbinom(hids+1,1,0.5)*2
```

Example

```
epsilona<-array(0.5,c(2,hids))
epsilonb=rep(0.5,hids+1)
lambda=0.1; phi=0.5; theta=0.7

for (k in 1:n) {
for (j in 1:hids) {
y[j,k]=logistic(a[2,j]+a[1,j]*x[k]) }
z[k]=logistic(b[hids+1]+sum(b[1:hids]*y[1:
hids,k])) }

for (h in 1:epoch) {
da<-array(0,c(2,hids))
db<-rep(0,hids+1)
```

Example

```
for (k in 1:n) {

p=(z[k]-t[k])*z[k]*(1-z[k])/n
for (j in 1:hids) { db[j]=db[j]+p*y[j,k] }
db[hids+1]=db[hids+1]+p

for (j in 1:hids) {
q=(z[k]-t[k])*z[k]*(1-z[k])*b[j]*y[j,k]*(1-
y[j,k])/n
da[1,j]=da[1,j]+q*x[k]
da[2,j]=da[2,j]+q }

}
```

Example

```
if (h>1) {
for (i in 1:2) { for (j in 1:hids) {
if (ga[i,j]*da[i,j]>0) {
epsilona[i,j]=epsilona[i,j]+lambda }
if (ga[i,j]*da[i,j]<=0) {
epsilona[i,j]=epsilona[i,j]*phi }
}}
for (j in 1:(hids+1)) {
if (gb[j]*db[j]>0) {
epsilonb[j]=epsilonb[j]+lambda }
if (gb[j]*db[j]<=0) {
epsilonb[j]=epsilonb[j]*phi }
}}
```

Example

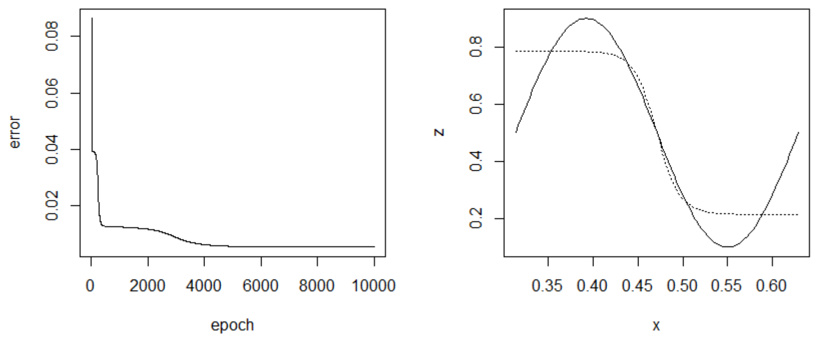
```
a=a-epsilona*da
b=b-epsilonb*db

if (h==1) { ga=da; gb=db }
if (h>1) {
ga=theta*ga+(1-theta)*da
gb=theta*gb+(1-theta)*db }

for (k in 1:n) {
for (j in 1:hids) {
y[j,k]=logistic(a[2,j]+a[1,j]*x[k]) }
z[k]=logistic(b[hids+1]+sum(b[1:hids]*y[1:
hids,k])) }
```

Example

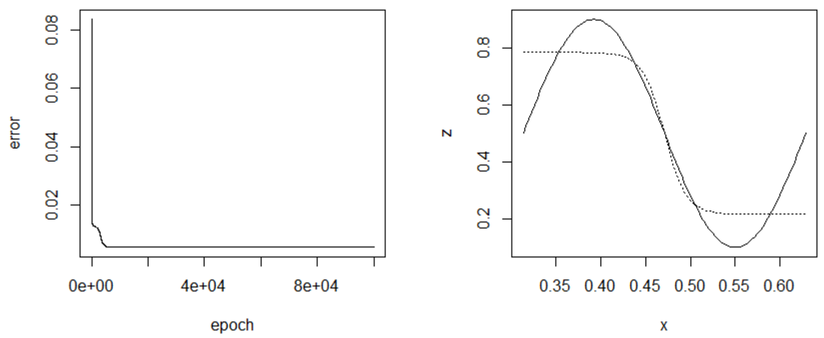
e[h]=0.5*sum((z-t)^2)/n

}

hids = 1; epoch = 10000; $t \in [0.1, 0.9]$

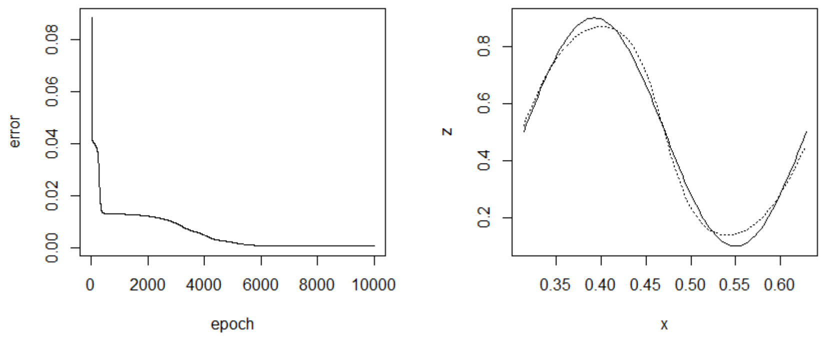


hids = 1; epoch = 100000; $t \in [0.1, 0.9]$

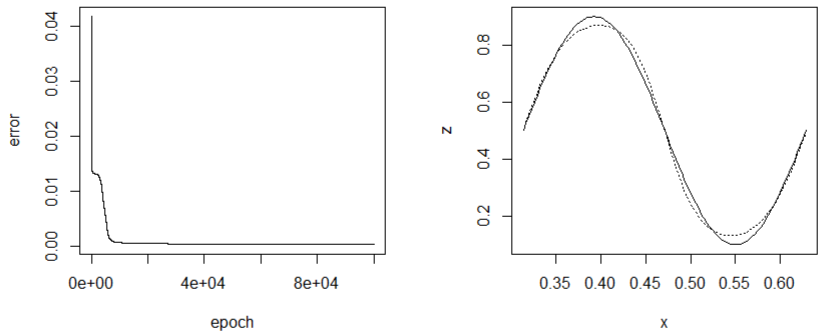


Example

hids = 3; epoch = 10000; $t \in [0.1, 0.9]$

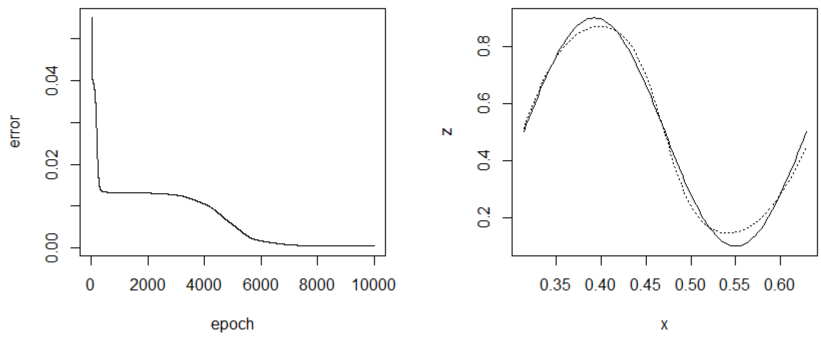


hids = 3; epoch = 100000; $t \in [0.1, 0.9]$

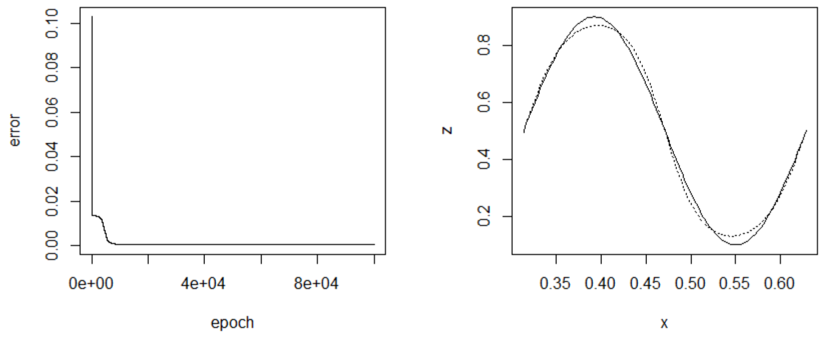


Example

hids = 5; epoch = 10000; $t \in [0.1, 0.9]$

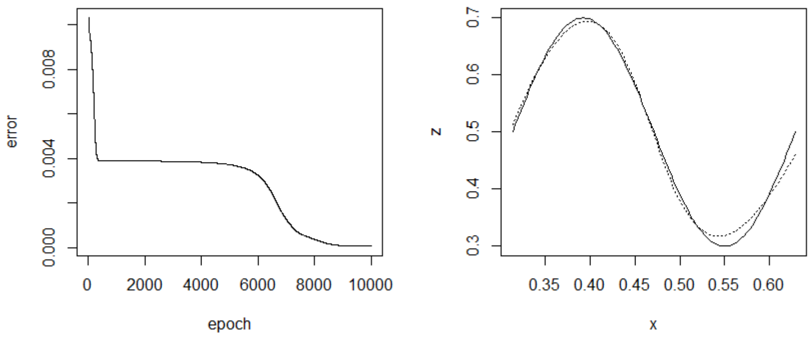


hids = 5; epoch = 100000; $t \in [0.1, 0.9]$



Example

hids = 5; epoch = 10000; $t \in [0.3, 0.7]$



hids = 5; epoch = 100000; $t \in [0.3, 0.7]$