Patterns and machine learning in Bioinformatics

Arne Elofsson

To read:

http://www.ploscollections.org/article/info%3Adoi%2F10.1371%2Fjournal.pcbi.0030116
http://en.wikipedia.org/wiki/Machine_learning

Extra:
http://www.ploscollections.org/article/info%3Adoi%2F10.1371%2Fjournal.pcbi.1000173

Detecting local motifs in proteins

- Patterns/Motifs method
- Machine learning method
  - Artificial Neural Networks
  - Support Vector Machines

Summary of earlier lectures

- Multiple alignments can be used to:
  - Improve detection of homologus proteins
  - Create Phylogenetic trees
- Homology detections can be improved using:
  - Profiles
  - HMMs
- PSIBLAST is a fast an accurate methods using iterative Blast searches
- Phylogenetic trees can be created using three methods
  - Parsimony
  - Distance based methods
  - Maximum likelihood
- Domains are the structural and evolutionary unit of proteins
  - Domain databases include Pfam, SCOP, CATH

Today

- **Patterns in proteins**
  - What is a pattern ?
  - What can patterns be used for.
- **How can we detect patterns**
  - Motifs
  - Profiles
  - Machine learning methods
    - ANN
    - SVM

Patterns vs Sequence search

- Domains are conserved

- In contrast patterns are short sequence motifs that are functionally important but not necessary homologous
- Example of patterns
  - Ser-proteases active site

Patterns and motifs

- Expression matching a MSA
- Example motif
  - A-[GC]-{0-1}-T
- DE Signal peptidases I Serine active site (PS00501).
  - PA [GS]-x-S-M-x-P-[AT]-[LF]

```
ACGT
AC-T
AG-T
```

Patterns and motifs

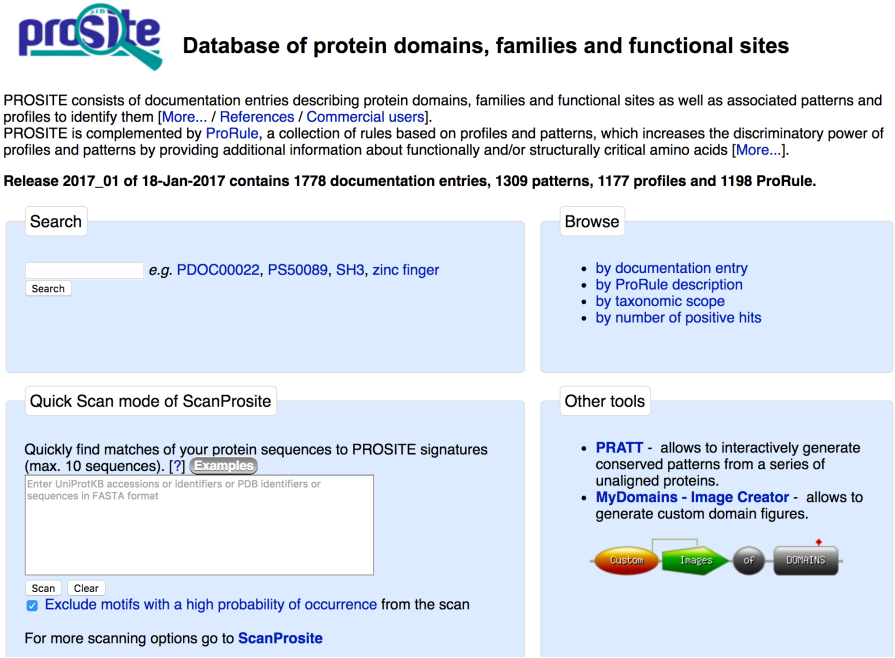
- Patterns origin in the BLOCKS database
  - The origin of BLOSUM matrices
- The prosite database
- Patterns limited in length
- Patterns binary Yes/No
  - Can be extended with "Gibbs-sampling", see Mount for details

```
ACGT
AC-T
AG-T
```

Profiles for local searches

- Same method as in profile searches
- More flexible than patterns
  - Not only exact matching
  - Different weights in different positions
  - Can include variable gap penalties
- Slower
- HMMs can also be used

The prosite database



- Contains many motifs for pattern
- Trypsin
  - TRYPSIN_HIS, PS00134; Serine proteases, trypsin family, histidine active site (PATTERN)
  - Consensus pattern: [LIVM] - [ST] - A - [STAG] - H - C
  - H is the active site residue
  - Sequences known to belong to this class detected by the pattern: ALL, except for complement components C1r and C1s, pig plasminogen, bovine protein C, rodent urokinase, ancrod, gyroxin and two insect trypsins

Prosite gives also motifs for some domains

PTS EIIB domains cysteine phosphorylation site signature.
N-[LIVMFY]-x(5)-C-x-T-R-[LIVMF]-x-[LIVMF]-x-[LIVM]-x-[DQ]

TOP PAGE | QUERY | RESULTS | SESSIONS | VIEWS | DATABANKS | HELP

Reset

View * Complete entries *

This entry is from:

PROSITE

Save

Link

Printer Friendly

PROSITE:PTS_EIIB_CYS

```
ID   PTS_EIIB_CYS; PATTERN.
AC   PS01035;
DT   JUN-1994 (CREATED); NOV-1997 (DATA UPDATE); JUL-1998 (INFO UPDATE).
DE   PTS EIIB domains cysteine phosphorylation site signature.
PA   N-[LIVMFY]-x(5)-C-x-T-R-[LIVMF]-x-[LIVMF]-x-[LIVM]-x-[DQ].
NR   /RELEASE=38,80000;
NR   /TOTAL=27(27); /POSITIVE=27(27); /UNKNOWN=0(0); /FALSE_POS=0(0);
NR   /FALSE_NEG=0; /PARTIAL=0;
CC   /TAXO-RANGE=???P?; /MAX-REPEAT=1;
CC   /SITE=4,phosphorylation;
DR   P09323, PTAA_ECOLI, T; P45604, PTAA_KLEPN, T; P40739, PTBA_BACSU, T;
DR   P08722, PTBA_ECOLI, T; P26207, PTBA_ERWCH, T; P24241, PTDA_ECOLI, T;
DR   P42015, PTGA_BACST, T; P20166, PTGA_BACSU, T; Q45298, PTGA_BRELA, T;
DR   P35595, PTGA_STRPN, T; P05053, PTGB_ECOLI, T; P37439, PTGB_SALTY, T;
DR   P54715, PTIB_BACSU, T; P31451, PTIB_ECOLI, T; P19642, PTOA_ECOLI, T;
DR   Q04938, PTSA_LACLA, T; P43470, PTSA_PEDPE, T; P12655, PTSA_STRMU, T;
DR   P05306, PTSB_BACSU, T; P27219, PTSB_KLEPN, T; P08470, PTSB_SALTY, T;
DR   P51184, PTSB_STAXY, T; P22825, PTSB_VIBAL, T; P39794, PTTB_BACSU, T;
DR   P36672, PTTB_ECOLI, T; P15400, SACX_BACSU, T; P39816, YBFS_BACSU, T;
3D   1GPR; 1AX3; 1IBA;
DO   PDOC00795;
//
```

Sequence Logos

- One way to look at a particular PSSM is to view it visually. Sequence logos are one way to do so, by illustrating the information in each column of a motif.
- Such a graph can indicate which residues and which columns are the most important as far as sequence conservation is concerned.
- The height of the logo is calculated as the amount by which uncertainty has been decreased
- If the frequency in the column is less than the frequency in the background, then a negative relative entropy can be computed, which can be shown by an inverted character in the logo.

What we want to achieve

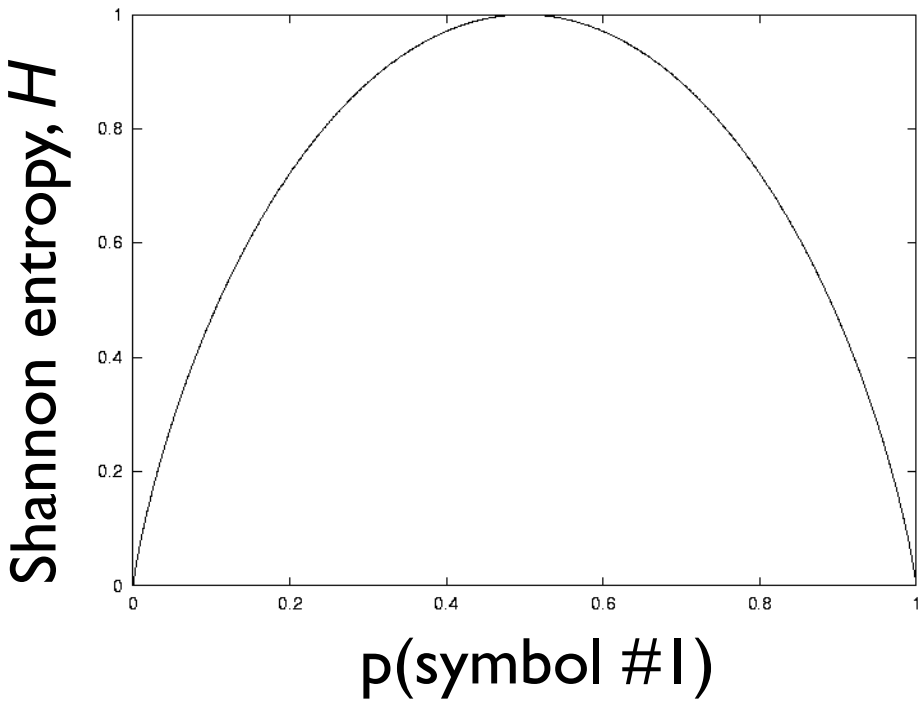
"He can compress the most words into the smallest ideas of any man I ever met."

Shannon's noiseless channel coding theorem: The minimal achievable value of R is given by the Shannon entropy of the source distribution,

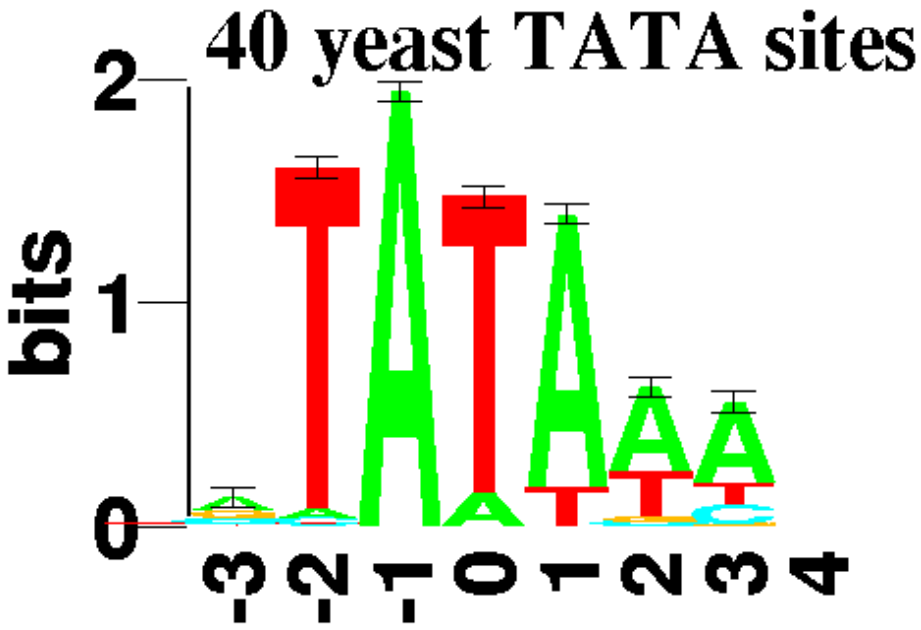
$$H(X) \equiv H(p_x) \equiv -\sum_x p_x \log(p_x),$$

where logarithms are taken to base two.

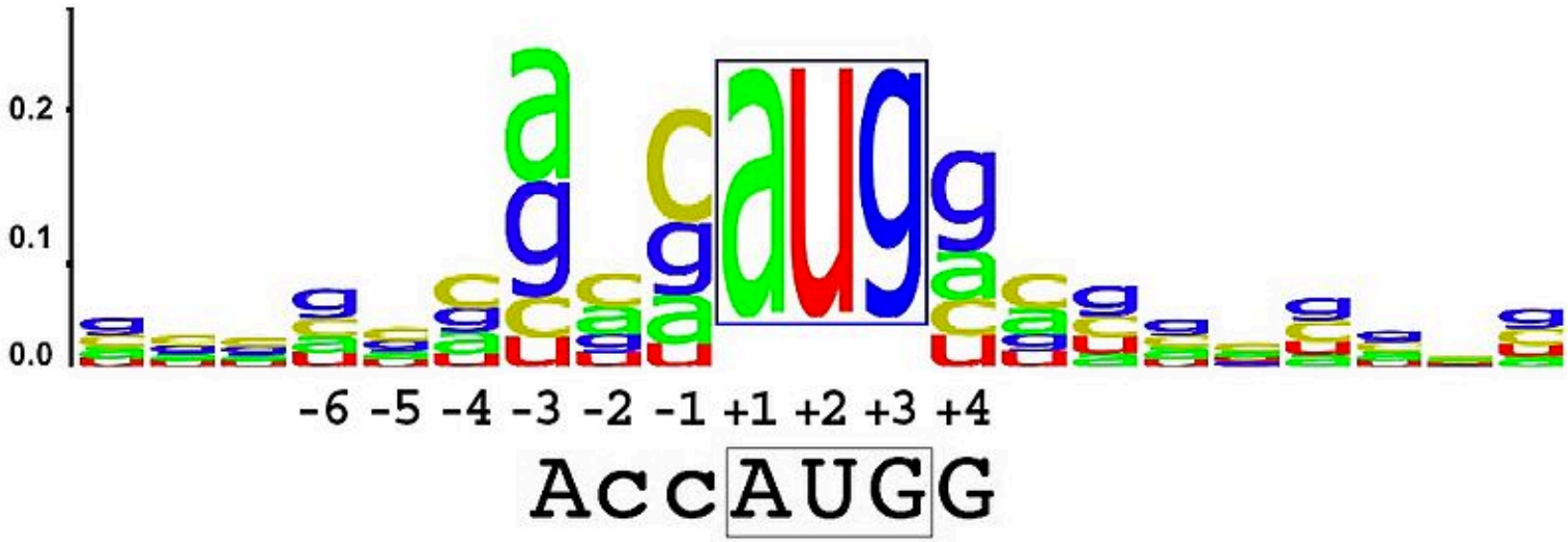
Entropy in the case of 2 symbols



Sequence Logos



A sequence logo showing the most conserved bases around the initiation codon from all human mRNAs (Kozak consensus sequence).

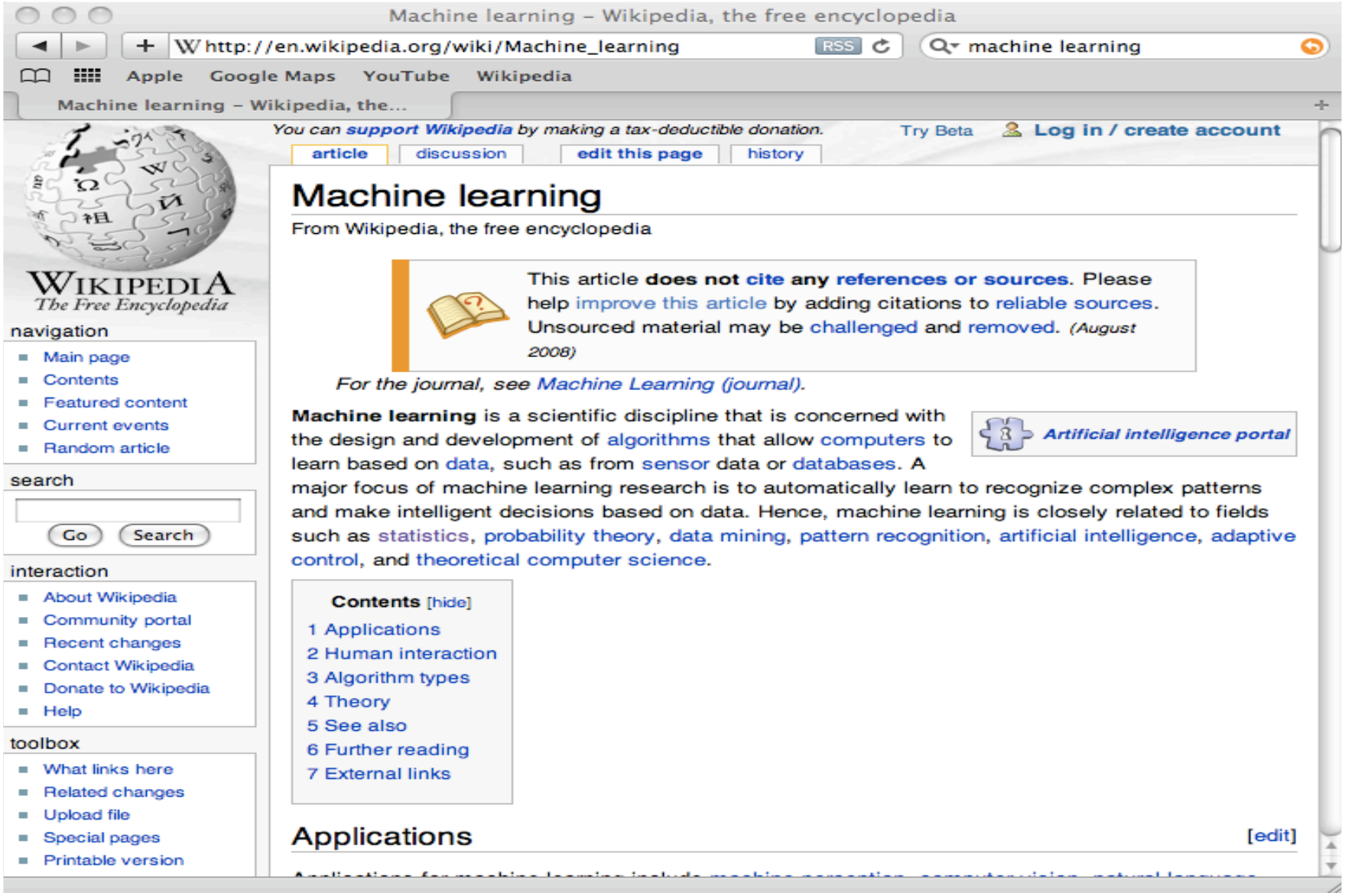


https://en.wikipedia.org/wiki/Sequence_logo

Machine learning

- Patterns and profiles do not include negative examples
- Machine learning are computer programs that "learn" the best way to solve a problems
- Input is positive and negative examples

Machine learning in bioinformatics



Applications of ML within Bioinformatics

<u>"Classical" Bioinformatics</u>

- Gene prediction
- Protein family classification
- Protein structure prediction
- Secondary structure prediction
- Transmembrane topology prediction
- Protein compartment prediction
- Sequence alignment

<u>Interpretation of high throughput experiments</u>

- Chromatin Structure prediction by DNA hypersensitive site assays
- Copy Number Variation
- Transcription factor analysis by ChIP-Seq
- Peptide/protein inference from shotgun proteomics
- Clustering analysis of transcriptomics/ proteomics data

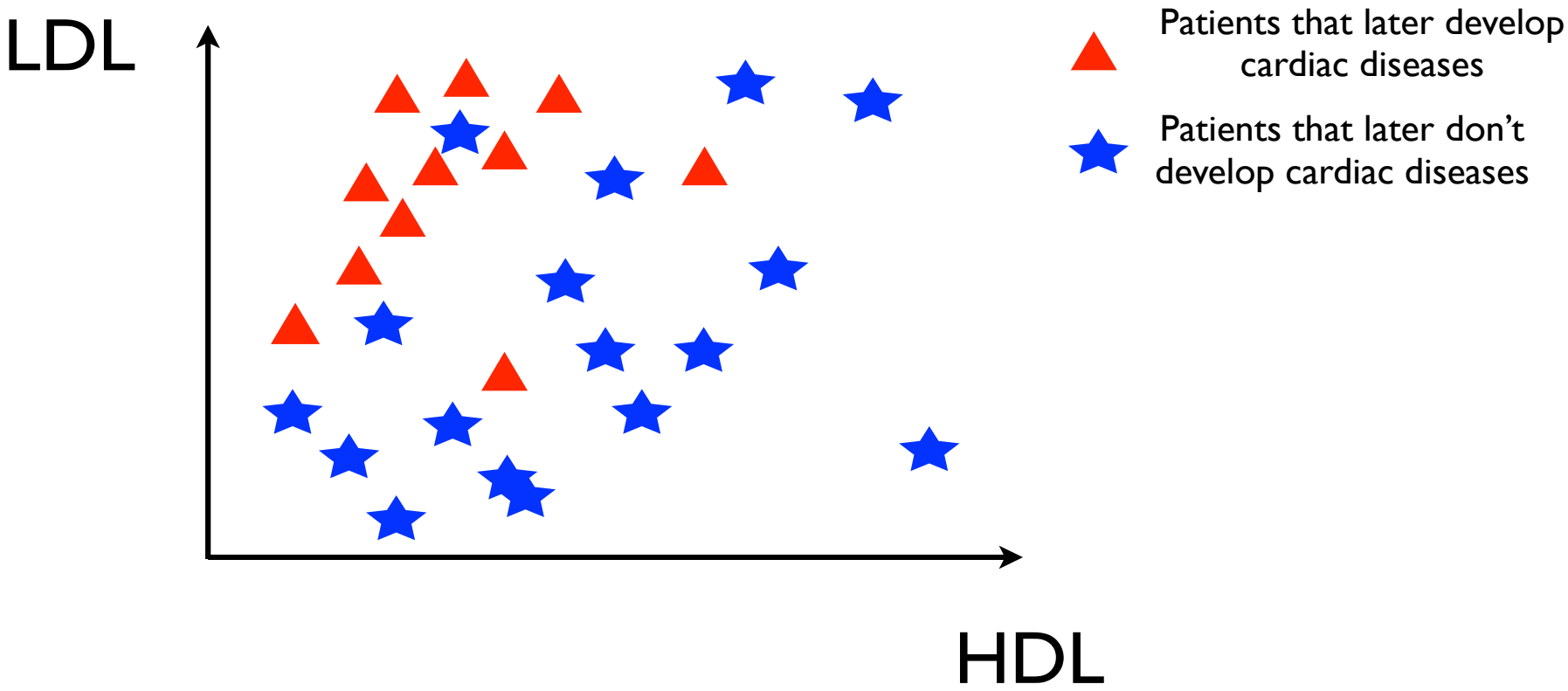
Terms

- Regression: Given a set of features of an example predict one (or more) variables (dependent variables)
- Classification: Given a set of features of an example predict which class the example belongs to

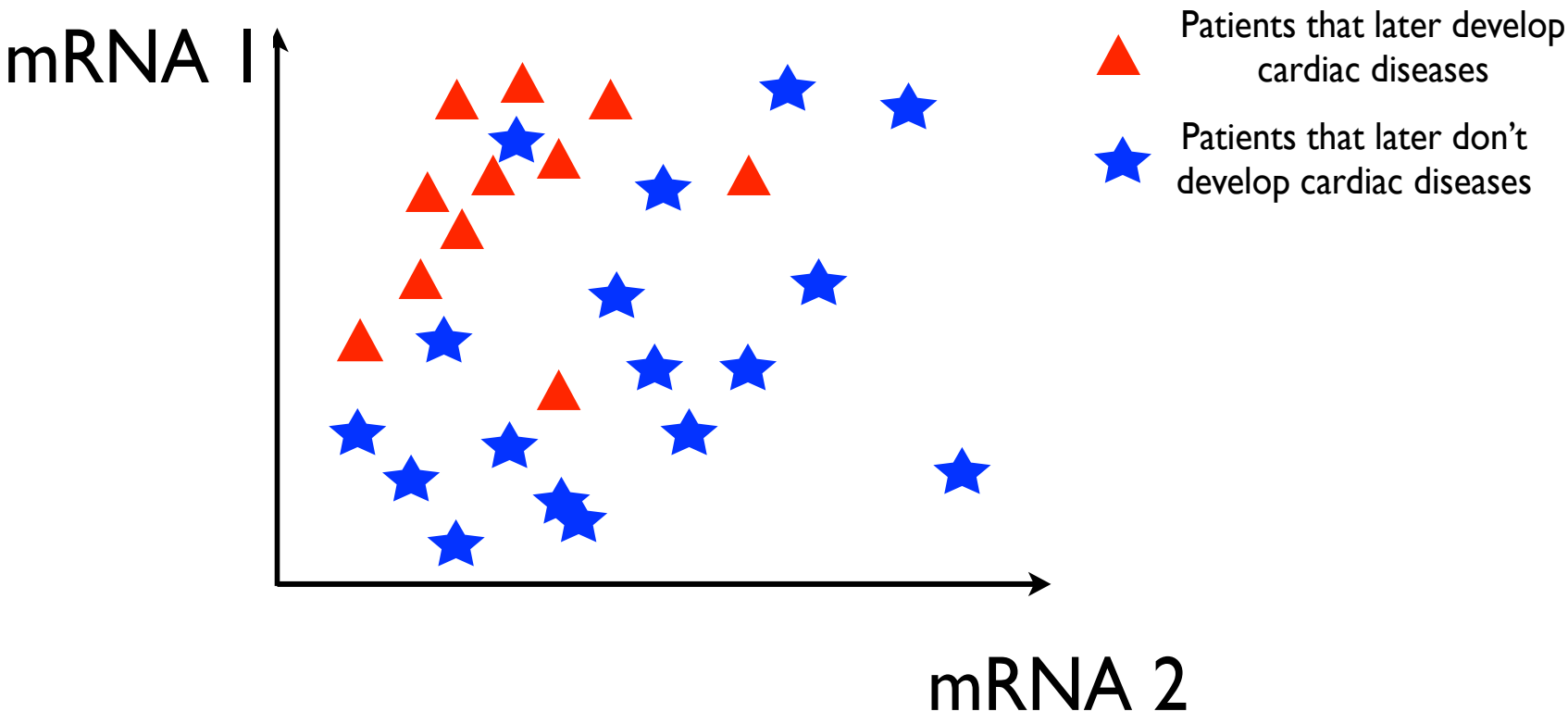
More Terms

- Supervised learning: All training examples are labeled
- Unsupervised learning: No examples are labeled (Clustering)
- Semi-supervised learning a few examples are labeled, but the bulk of our set is unlabeled

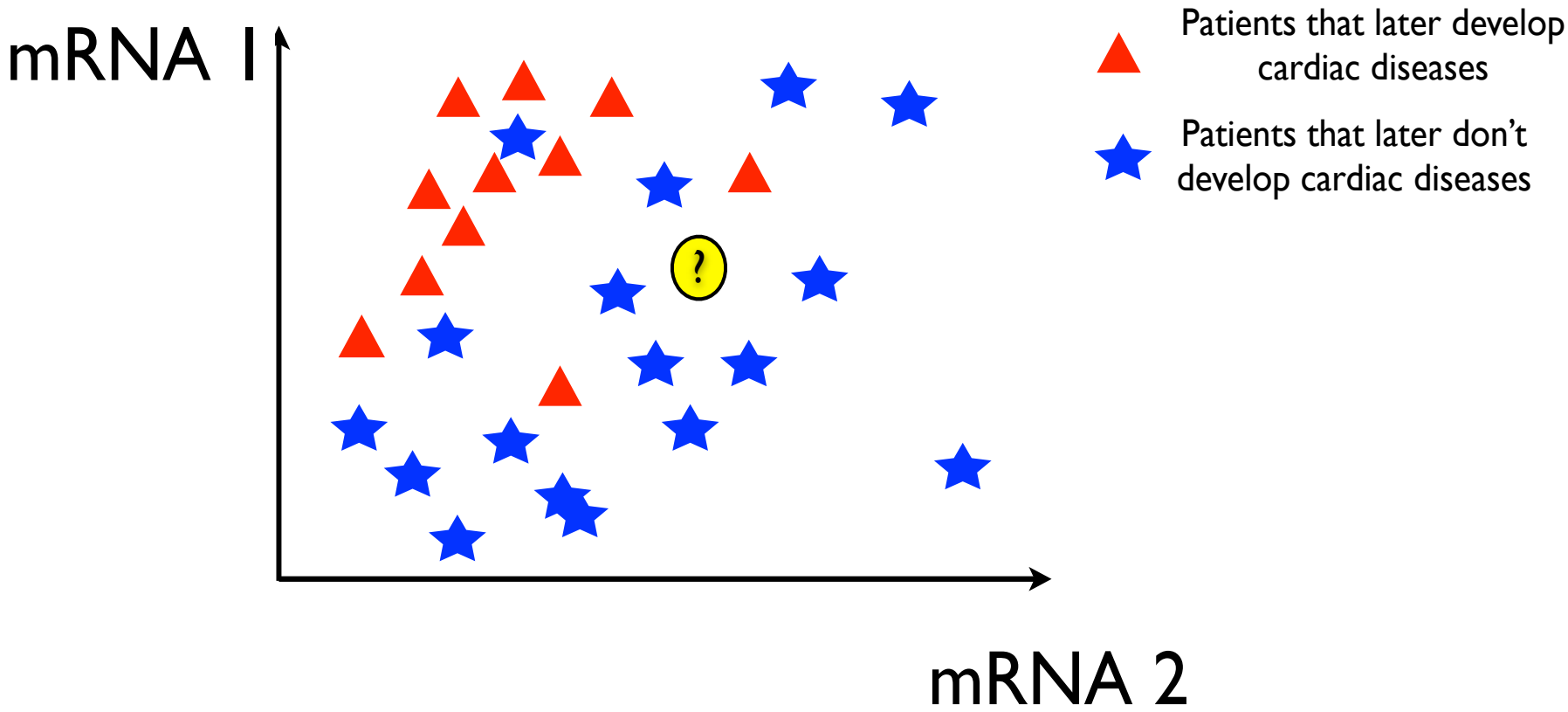
Classification - supervised learning



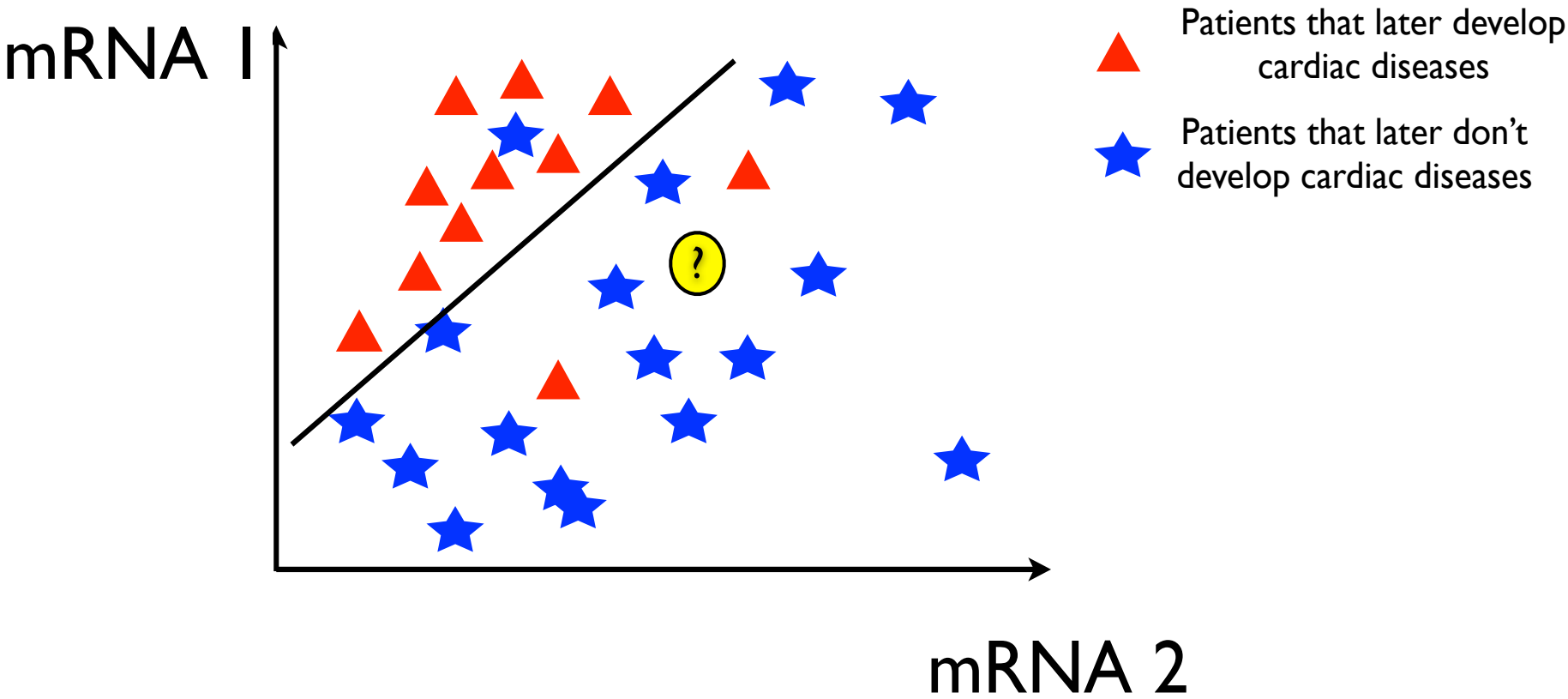
Classification - supervised learning



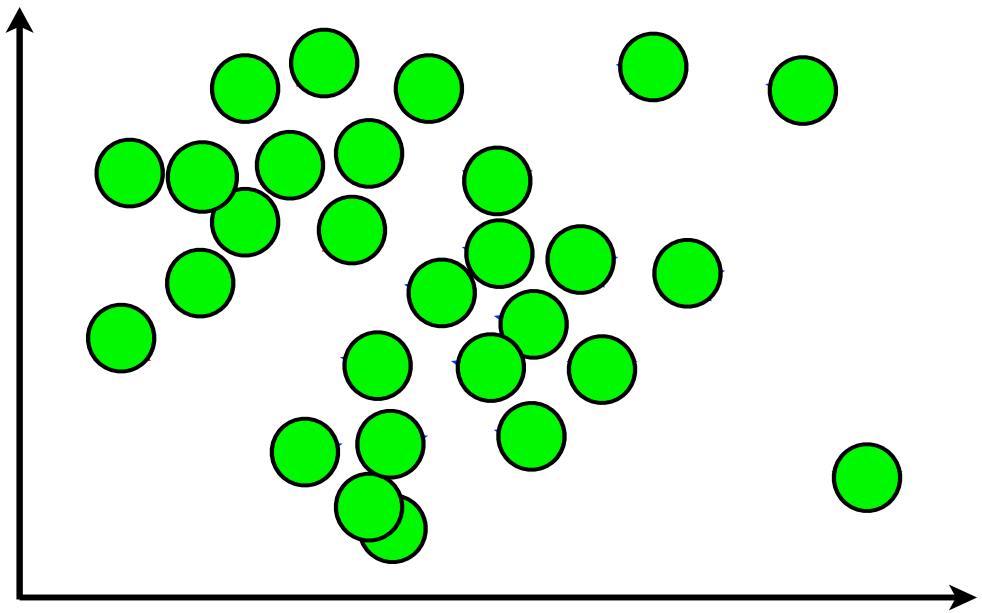
Classification - supervised learning



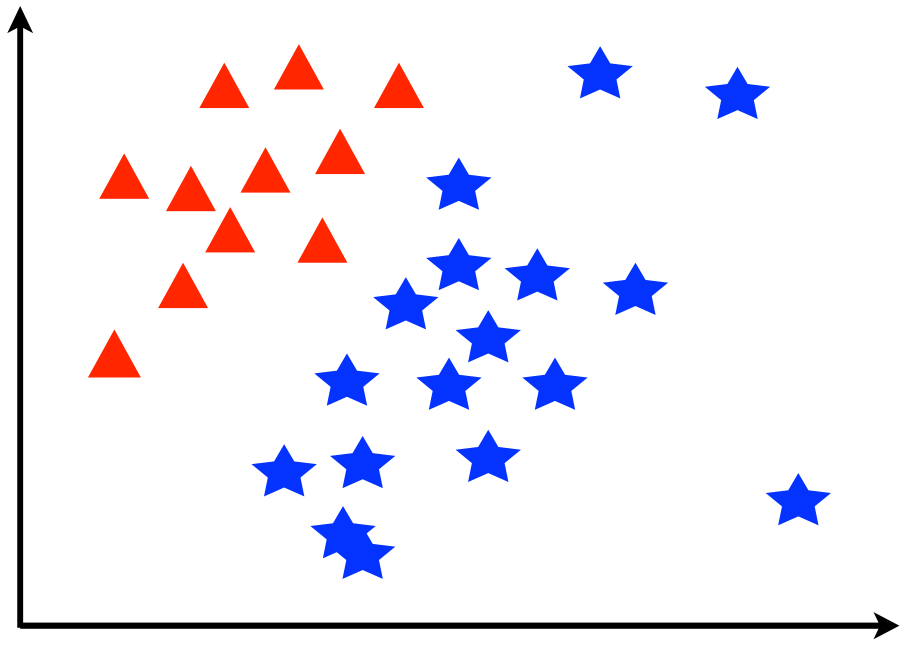
Classification - supervised learning



Classification - unsupervised learning



Classification - unsupervised learning



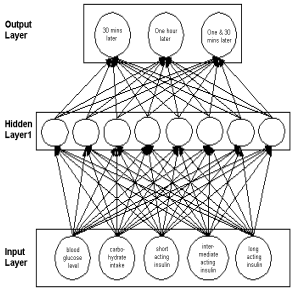
ANN Overview

(slides from Olof Emanuelsson):

1. Introduction to artificial Neural Networks (NN or ANN)
   - the idea of "artificial neurons"
   - important concepts of artificial neural networks
   - two different types: supervised vs. unsupervised

2. One type of NN covered in more detail: *feed-forward* NN
   - the over-all principle
   - the components (nodes, weights)
   - the training procedure

3. An example application: *TargetP*
   - the biological problem
   - creating training sets
   - the neural networks



The idea behind artificial neural networks

The brain of a vertebrate is (in general) capable of *learning* things

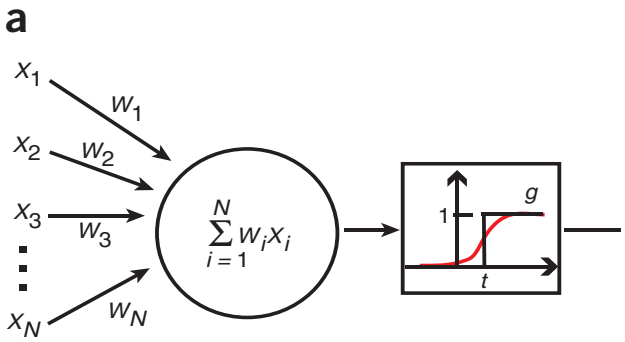
Example: having seen a number of trees, a normally gifted person will be able to recognise almost all types of trees

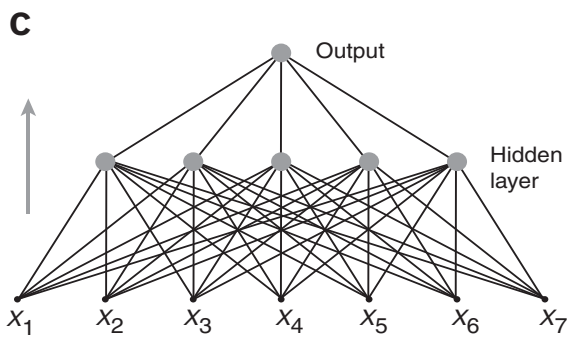
The idea: to construct networks of *artificial* neurons and make them *learn* and *generalize* in a way similar to how the physiological neural networks do that

As an example, the artificial neural networks may learn to recognize a particular type of sequence motif, e.g.:

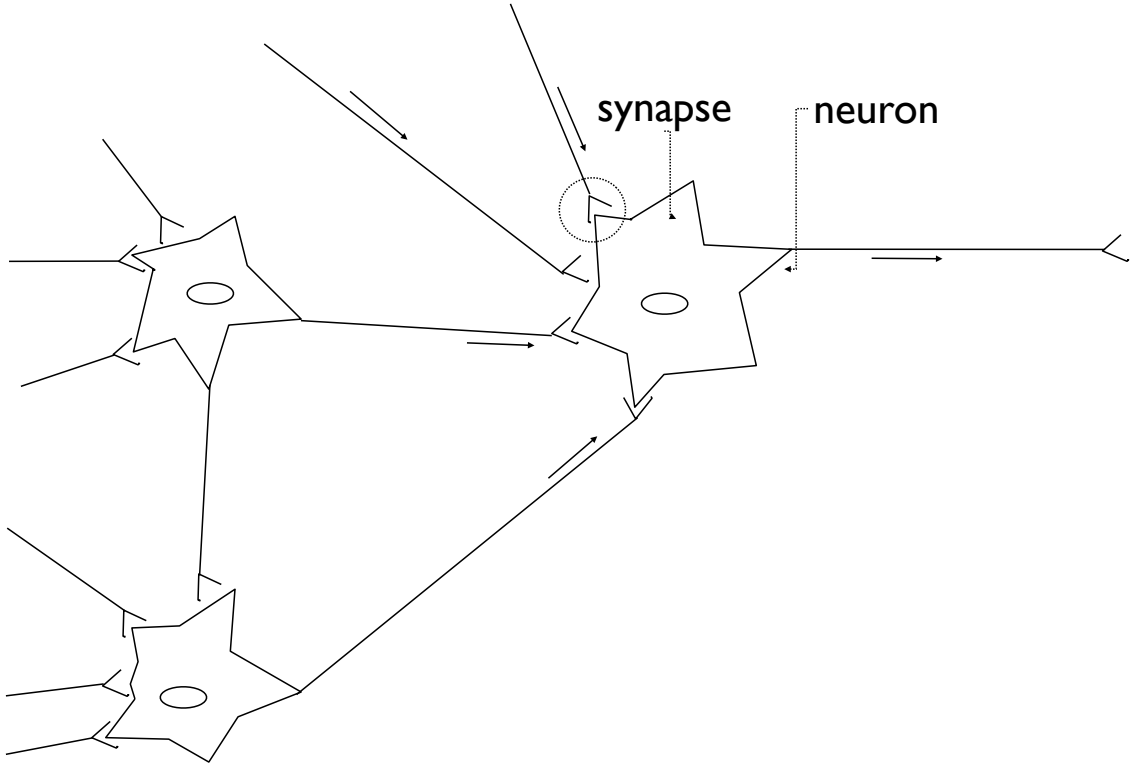
- the local sequence environment around a residue that is part of a secondary structure element
- a subcellular localization signal ("address label")

Neural Nework





A "physiological" neural network:



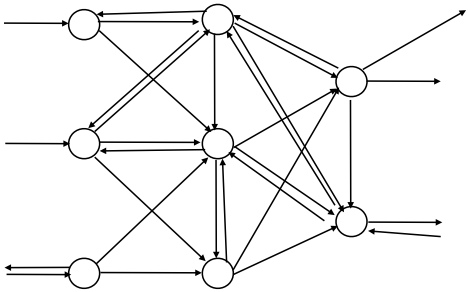
The neuron sums the input potentials and <u>fires</u> if the sum is above some threshold

Different synapses have different impact on the firing (+ or -)

This is formalised into an *artificial* neural network

The neurons are computing nodes: summing the input, outputting something based on that sum

The connections have different strengths (even negative)

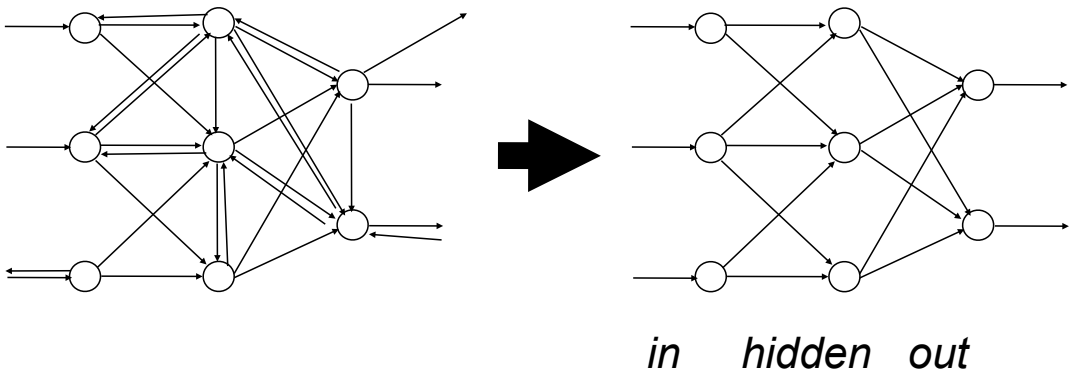


An example (not fully connected)

The feed-forward neural network
- *architecture*

Allow connections only in one direction (feed-*forward* NN)

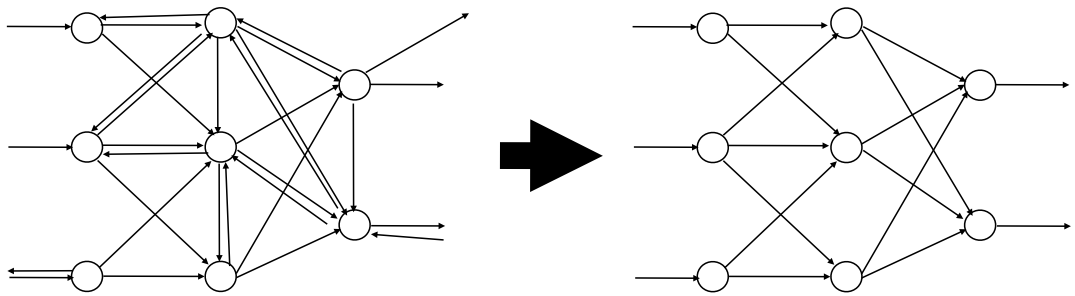
The neurons will be ordered in *layers*:



The feed-forward neural network
- *architecture*

Allow connections only in one direction (feed-*forward* NN)

The neurons will be ordered in *layers*:



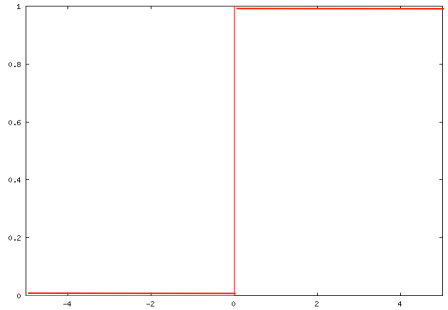
in *hidden* *out*

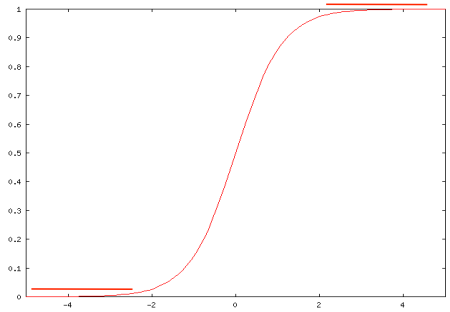
The feed-forward neural network
- *the transfer function f(v)*

logistic neuron

f(v)

$$f(v)=\begin{cases}1, if\ v\geqslant 0\\ 0, if\ v<0\end{cases}$$





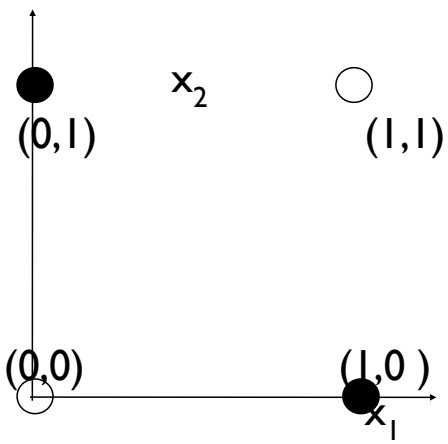
$$f(v)=\frac{1}{1+\exp(-av)}$$

(where *a* is the slope parameter)

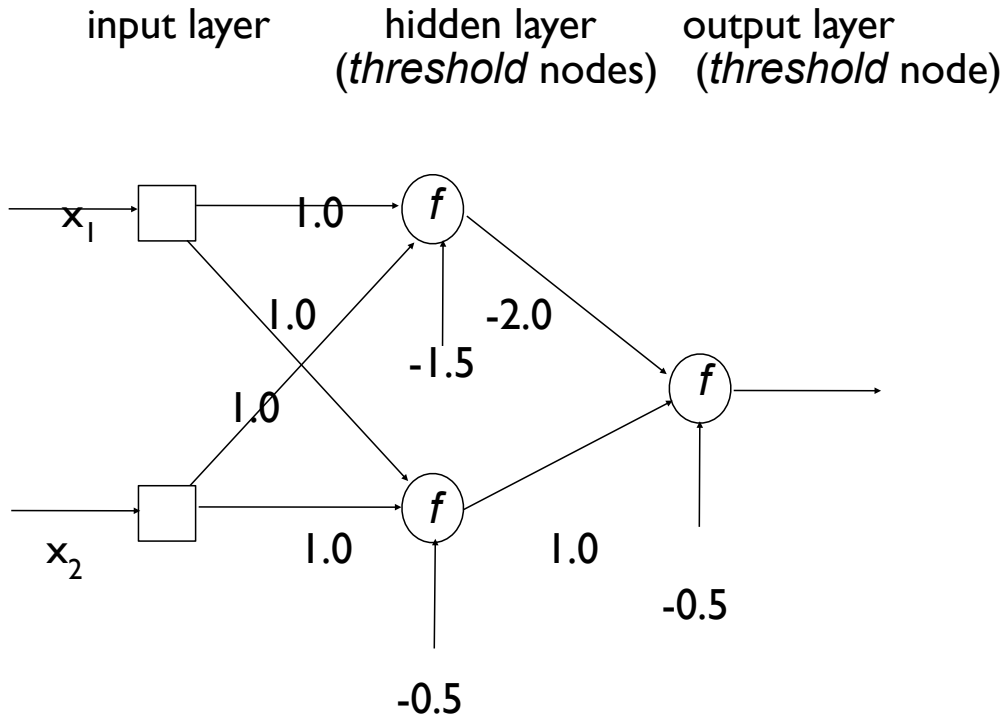
The feed-forward neural network
- *the XOR problem*

A non-linear *classification* problem

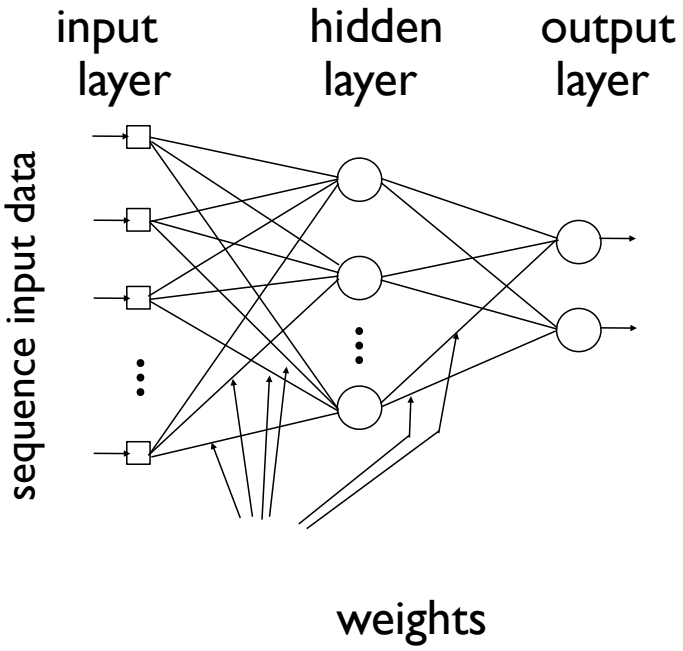
0 xor 0 = 0
1 xor 0 = 1
0 xor 1 = 1
1 xor 1 = 0



The feed-forward neural network
- *solving the XOR problem*

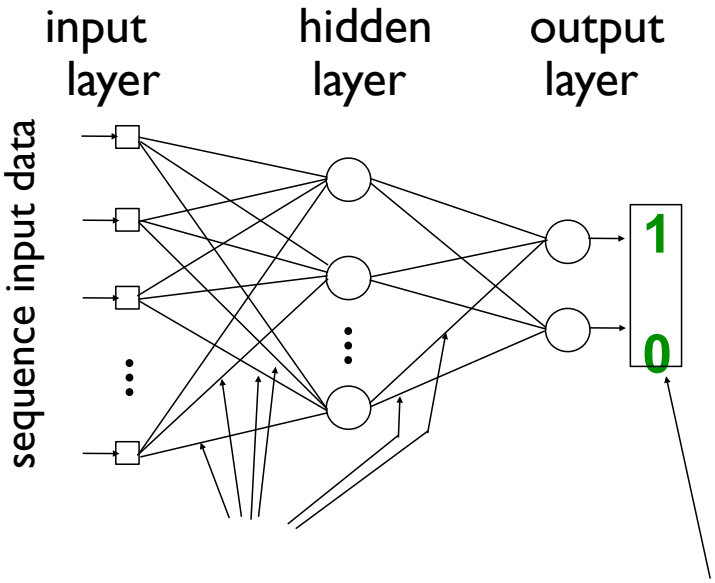


The feed-forward neural network
- the training principle



- (sequence) data presented at input
- correct answer fixed at output
- difference between correct and actual output used for weight adjusting (*training*)
- present both "positive" and "negative" training examples

The feed-forward neural network
- the training principle



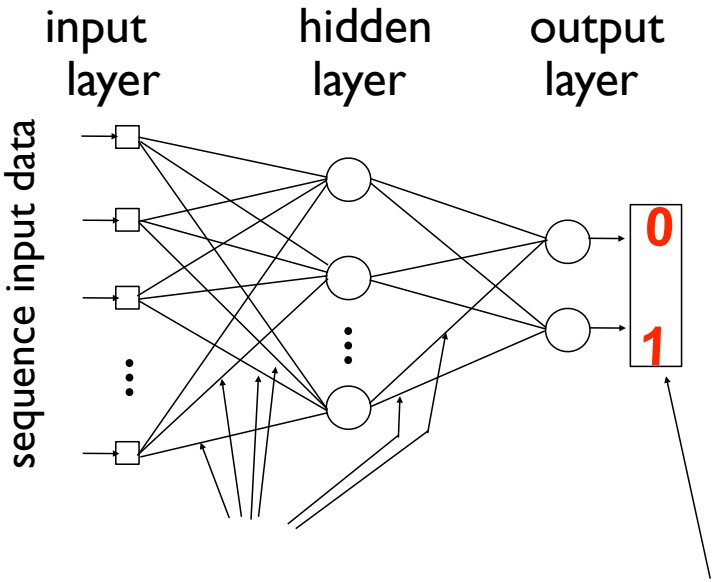
- (sequence) data presented at input

- correct answer fixed at output

- difference between correct and actual output used for weight adjusting (*training*)

weightsif motif *is present* in the sequence ("positive" training example)

The feed-forward neural network
- the training principle



- (sequence) data presented at input

- correct answer fixed at output

- difference between correct and actual output used for weight adjusting (*training*)

weightsif motif *not present* in the sequence ("negative" training example)

The feed-forward neural network
- *the training algorithm*

1. Start with random weights

2. Show one input example and calculate output

3. Calculate output errors (difference between observed and desired output)

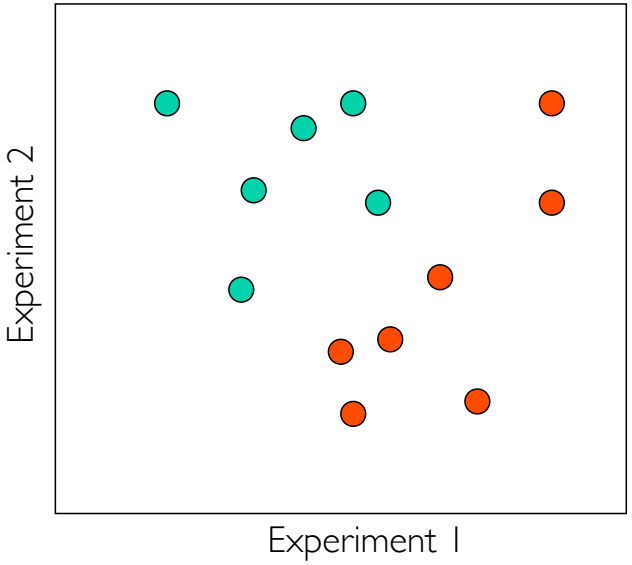
4. Adjust weights to decrease the error (using error backprop. alg.)

5. Repeat (2)-(4) for all input examples

6. Repeat (2)-(5) until error minimum is found (each repeat is called an *epoch*) (typically 50-1000 epochs)

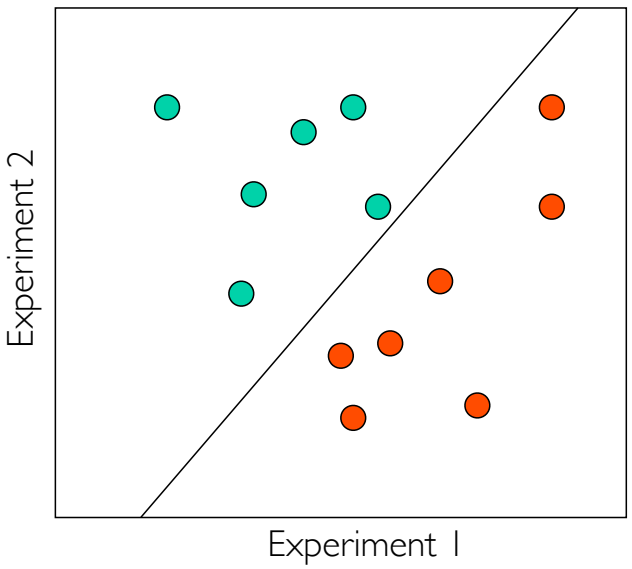
Support Vector Machines

Another machine learning method



- Consider *m* points in an n-dimensional space (*m* genes, *n* experiments)

Support Vector Machines



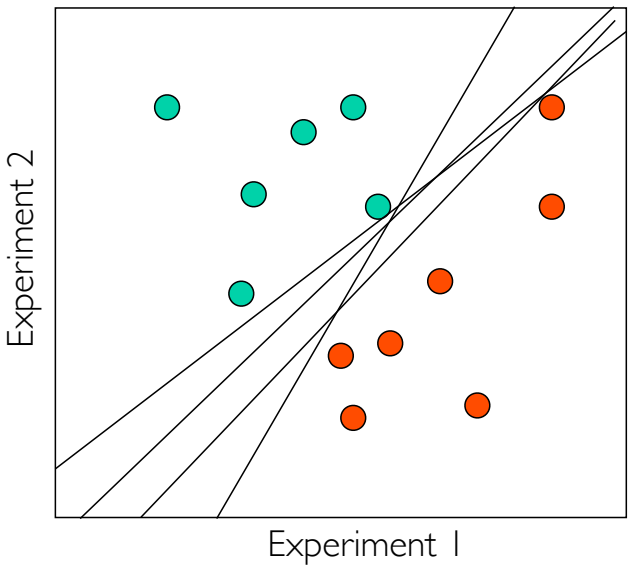
- Learning in SVMs involves finding a hyperplane (decision surface) that separates the examples of one class from another.

Support Vector

- For the ith example, let x_i be the vector of expression measurements, and y_i be +1, if the example is in the class of interest; and –1, otherwise
- The hyperplane is given by:

$$w \cdot x + b = 0$$

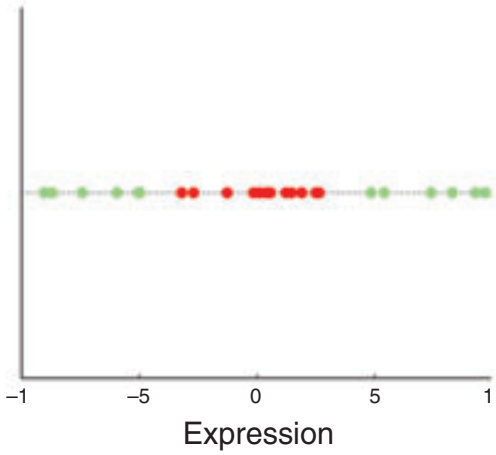
Support Vector Machines

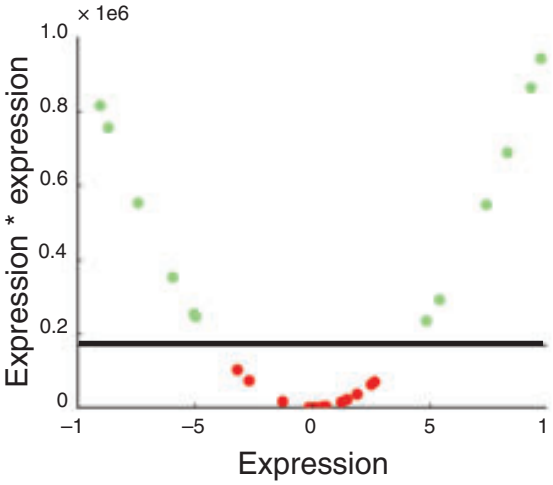


- There may be many such hyperplanes..
- Which one should we choose?

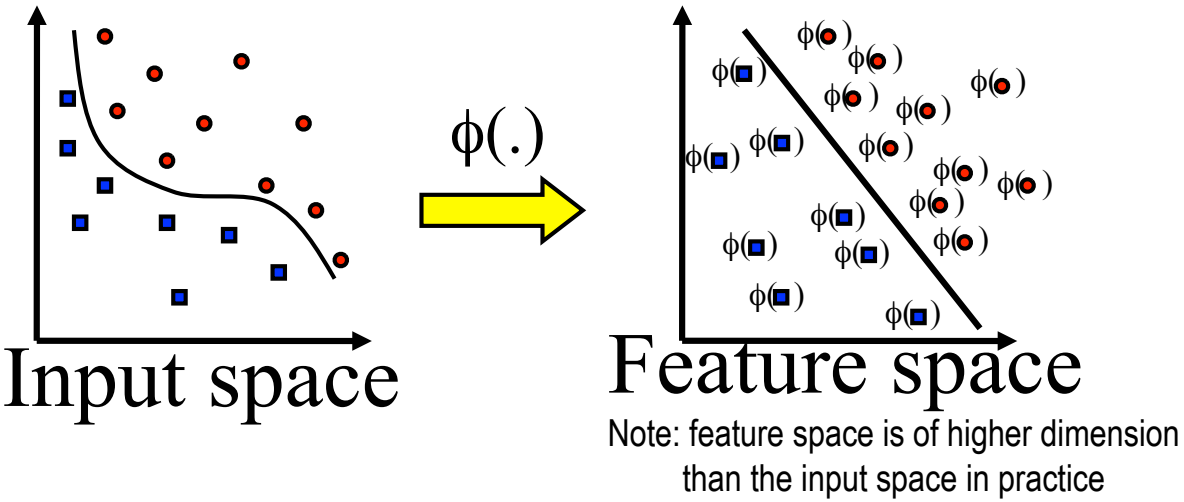
Support Vector Machine (2)

- Kernels: Any non-linear problem may be transformed into a linear problem if we select the right kernel





Transforming the Data

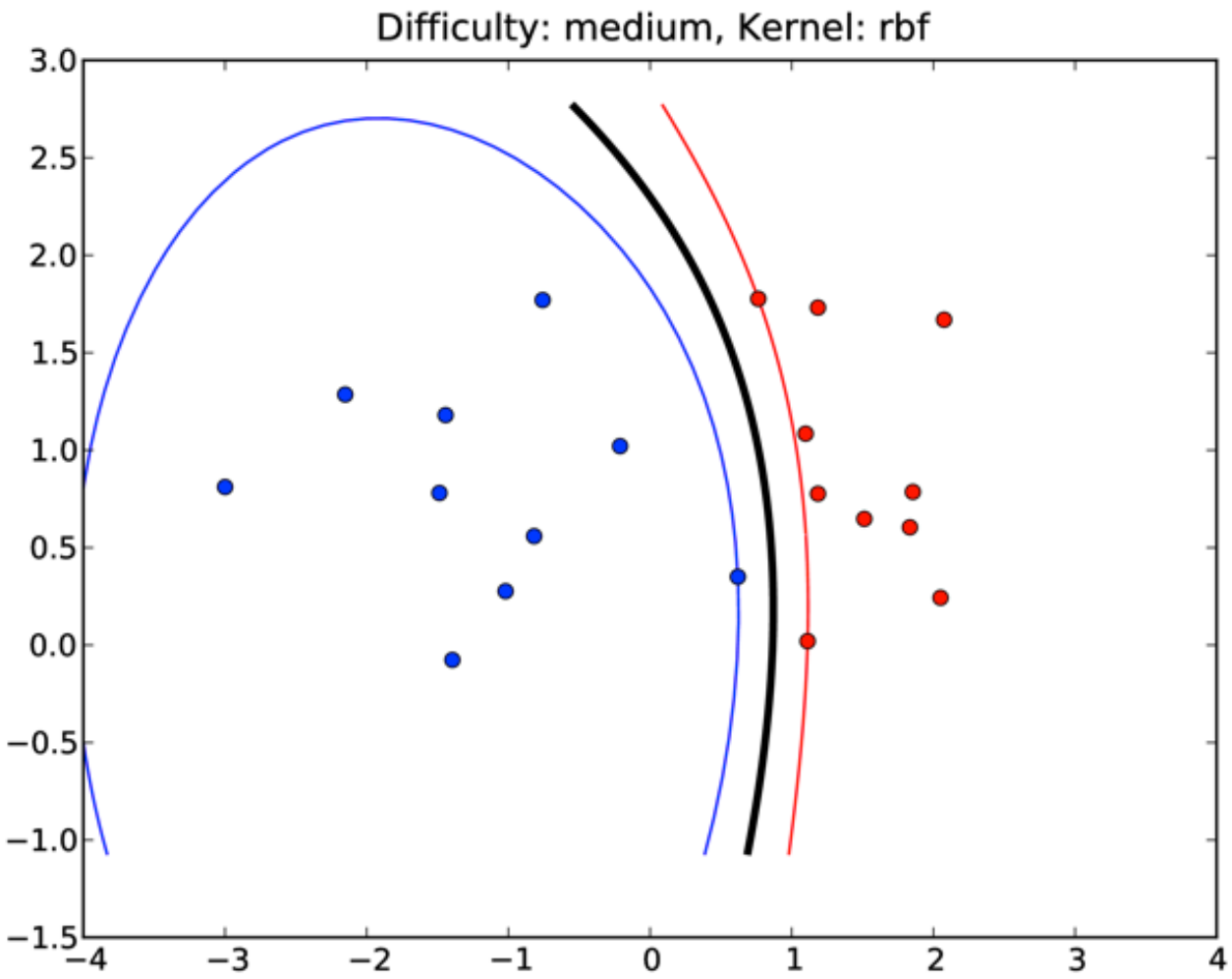


- Computation in the feature space can be costly because it is high dimensional
  - The feature space is typically infinite-dimensional!
- The kernel trick comes to rescue

Kernel Functions

- In practical use of SVM, the user specifies the kernel function; the transformation $\phi(.)$ is not explicitly stated
- Given a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$, the transformation $\phi(.)$ is given by its eigenfunctions (a concept in functional analysis)
  - Eigenfunctions can be difficult to construct explicitly
  - This is why people only specify the kernel function without worrying about the exact transformation
- Another view: kernel function, being an inner product, is really a similarity measure between the objects

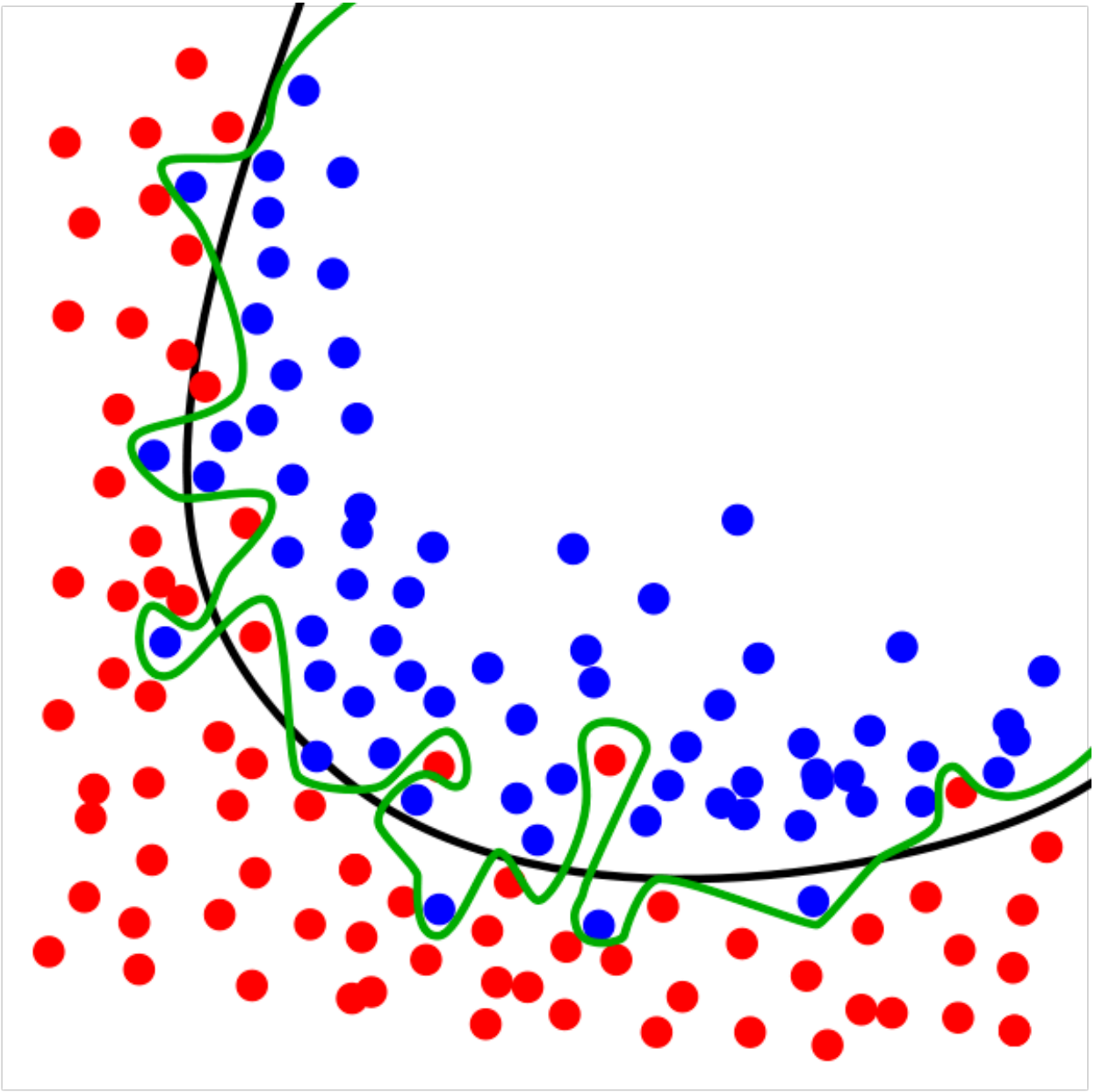
Which kernel



SVM vs. Neural Networks

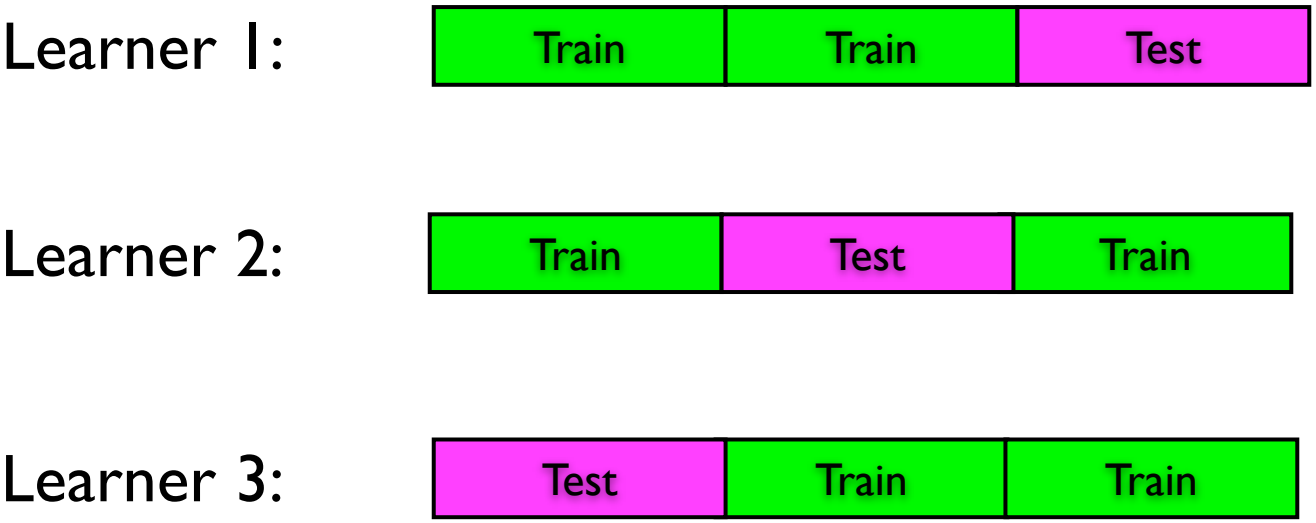
- SVM
  - Represents linear or nonlinear separating surface
- Neural Network
  - Represents linear or nonlinear separating surface
  - Weights determined by optimization method (optimizing sum of squared error—or a related objective function)

Overfitting



Cross Validation

3-fold cross validation



Validation strategies

- If we want to be able to detect over-fitting we need to train our method examples in a training set that is separate from the examples that we test our method with the test set.
- If we need to select hyper-parameters we need to yet another separate test set to find an optimal value.

The training set

- large enough

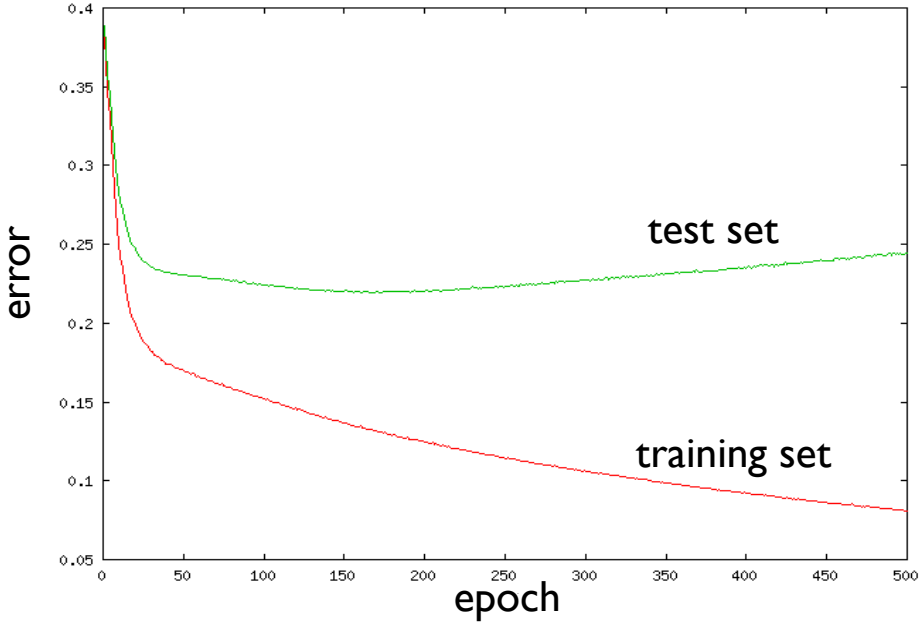
- contain all possible classes in approximately equal amounts

- unbiased, i.e. no particular type *within* a class should be overrepresented --- this is important for two reasons:
  - if training set is biased towards a particular type, so will the ANN be
  - if training and test set contain too similar examples, the performance will be over-estimated

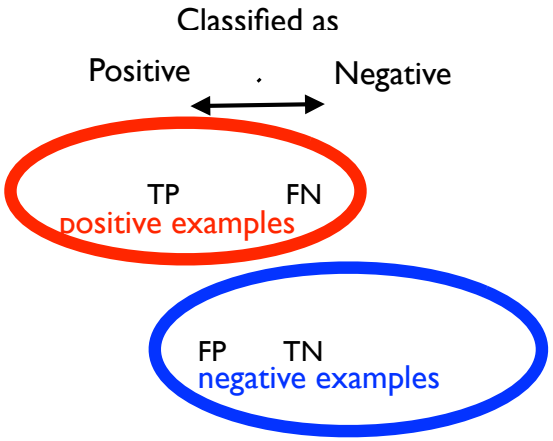
in short: the training set should be *representative*

When to stop training

We want to get a good *generalization* performance *and* to avoid over-fitting of the parameters to the *training* set (over-training)



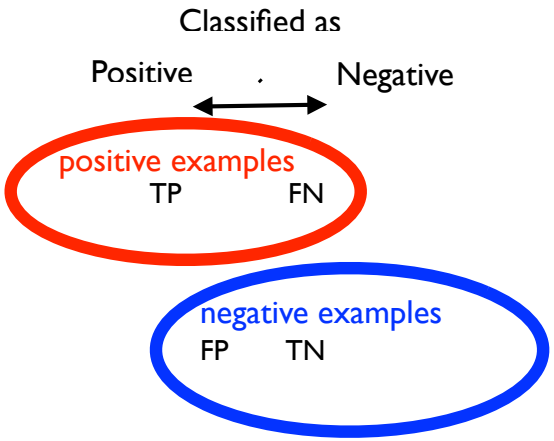
Metrics



	Classified as positive	Classified as negative
Positive example	TP	FN
Negative example	FP	TN

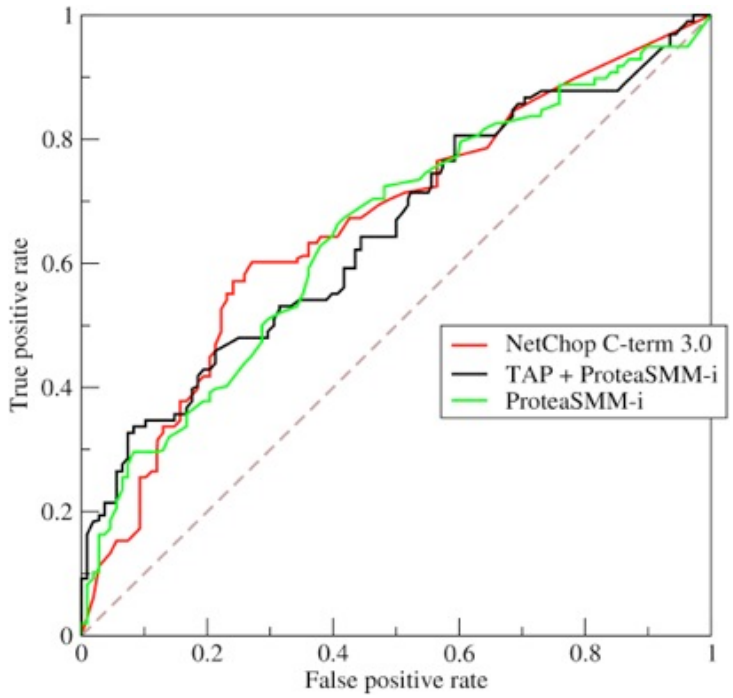
- TP = True positive = Correctly classified as positive example
- FP = False positive = Incorrectly classified as positive example
- FN = False negative = Incorrectly classified as negative example
- TN = True negative = Correctly classified as negative example

Metrics



- precision=accuracy=TP/(TP+FP)
- recall=sensitivity=TP/(TP+FN)
- True Positive Rate=TPR=TP/(TP+FN)
- False Positive Rate=FPR=FP/(FP+TN)
- False Discovery Rate=FDR=FP/(FP+TP)
  - accuracy=1-FDR
- Matthews correlation coefficient=
  MCC=(TP*TN-FP*TN)/sqrt(TP+FP)(TP+FN)(TN+FP)(TN+FN)
  MCC is in the range +1 to -1.
  MCC of +1 represents a perfect prediction, 0 an average random prediction and −1 an inverse prediction.

Receiver operating characteristic (ROC) plot



ROC score =
area under the ROC curve

ROC_{50} score =
area under the ROC curve
up to 50 negative examples

How to measure performance for classifications

Matthew's correlation coefficient for a 2-category predictor:

$$MCC = \frac{tp * tn - fp * fn}{\sqrt{(tn + fn)(tn + fp)(tp + fn)(tp + fp)}}$$

Sensitivity= tp/(tp+fn)
the fraction of the sequences that *belong* to a certain class
that really are *predicted* to that class

Specificity= tp/(tp+fp)
the fraction of the sequences that are *predicted* to a certain class

Sequence analysis using ANNs

Features

- Normalize features so that they have a uniform spread
- Sparse representation of amino acids - each position in a string

A	C	D	E	F	...	Y
0	1	0	0	0	...	0

Input encoding - going from amino acid sequence to numbers

- sparse encoding [20 numbers per residue]

MASL... =>

M A S L

00000000001000000000 10000000000000000000 00000000000000010000 0000000000 10...

- physicochemical encoding

e.g. [3 numbers per residue]

1. residue side-chain volume
2. hydrophobicity
3. charge

Input encoding - going from amino acid sequence to numbers

- sparse encoding [20 numbers per residue]

PROBLEM:

what if the motif we are looking for has different lengths in different proteins!?

SOLUTION:

==> use a "sliding window technique"

i.e. looking at a *piece* of the sequence at a time

Input encoding - going from amino acid sequence to numbers

- sliding input window

sequence: `MASLVLVRSLAVAFLDAGRSVRAASAVEGPA...`

if window size is 7:

1st window	MASLVLV
2nd window	ASLVLVR
3rd window	SLVLVRS
...	...
11th window	AVAFLDA
...	...

Input encoding - going from amino acid sequence to numbers

- sliding input window

sequence: `MASLVLVRSLAVAFLDAGRSVRAASAVEGPA...`

if window size is 7:

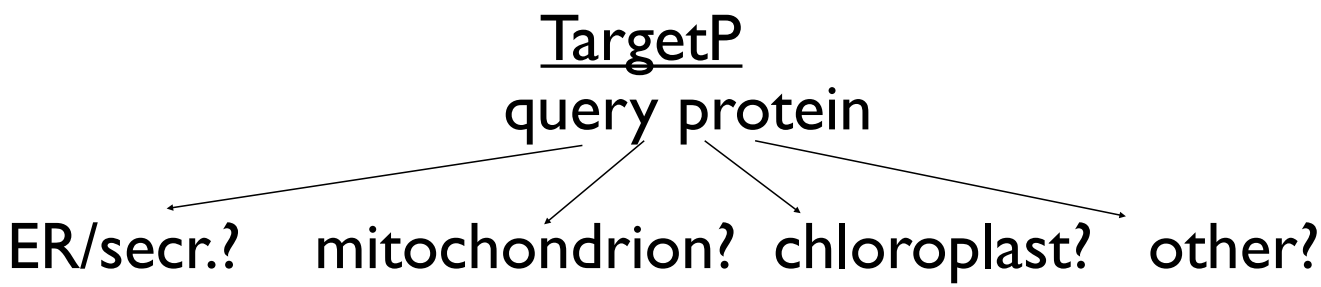
1st window	MASLVLV
2nd window	ASLVLVR
3rd window	SLVLVRS
...	...
11th window	AVAFLDA

1st window translated using "sparse encoding":

0000000000010000000000 1000000000000000000000 0000000000000000010000 0000000000100000000000 ...

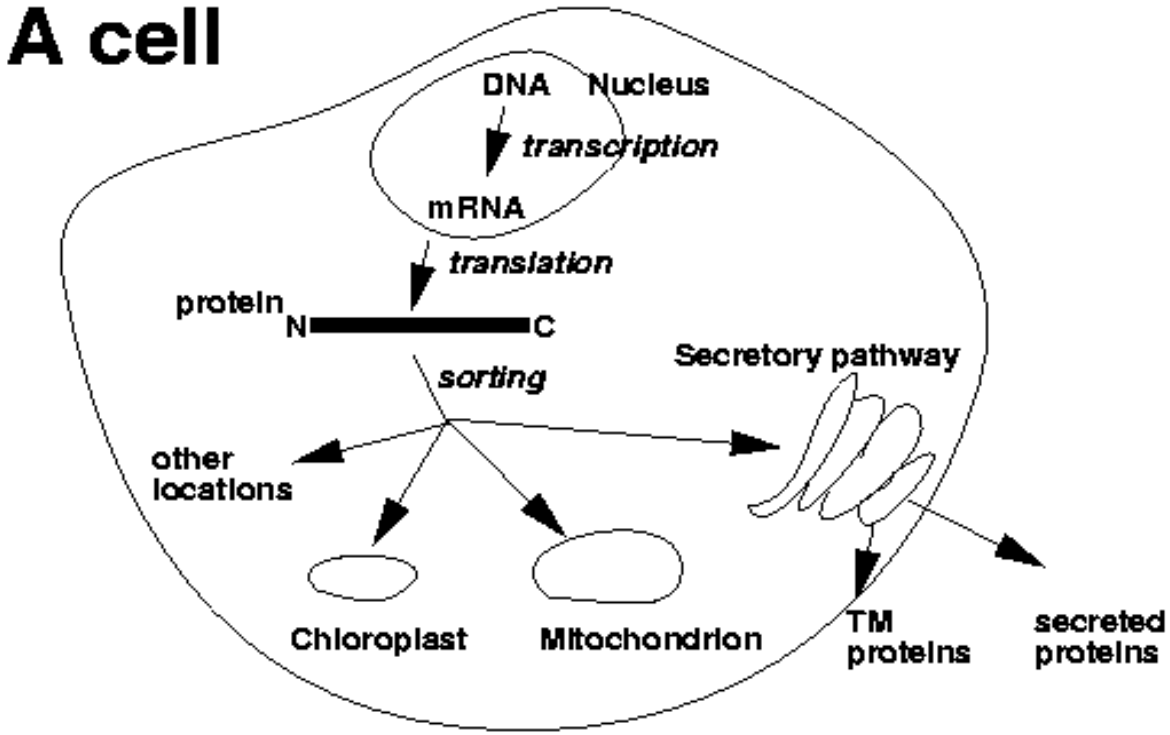
TargetP: An example application of neural networks in molecular biology

TargetP - predicts the subcellular localization of proteins based on their amino acid sequence

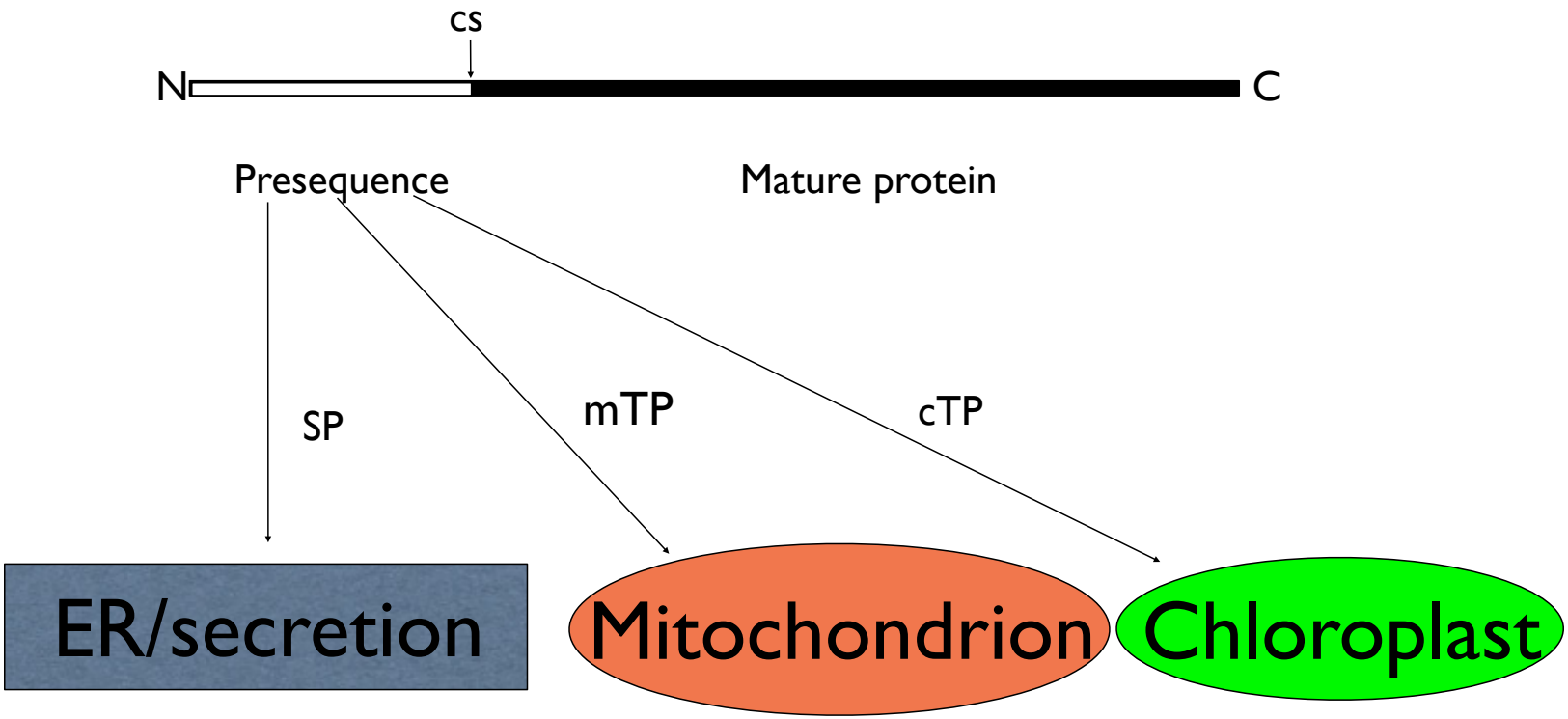


What is the problem?

- Proteins are synthezised in the cytosol
- They need to be sorted to their proper location

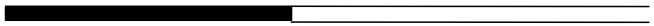


Three N-terminal presequences



The presequences; weak consensus

[I,V]-x-[A,C]
A



Signal Peptide, SP
~25 aa's

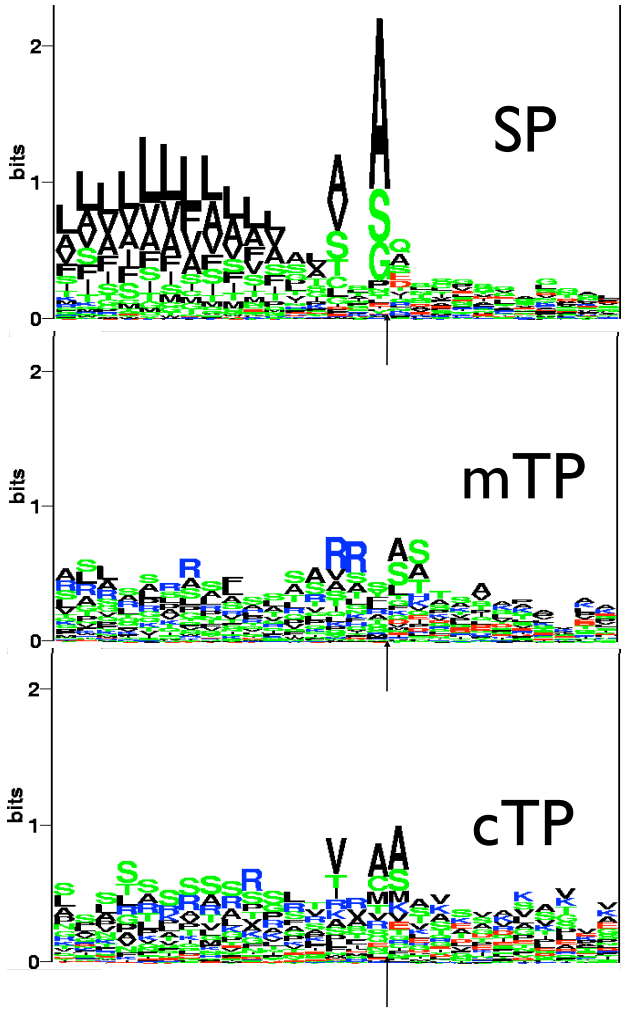
R-10 / R-3/ R-2

Mitochondrial Targeting
Peptide, mTP
~35 aa's

VRA AA

Chloroplast Transit
Peptide, cTP
~50 aa's

Sequence logos



A way of visualising multiple sequence alignments, and the degree of conservation at the positions

Here: Plant sequences aligned around their annotated cleavage site

Data sets

1. Extract sequences from SWISS-PROT

2. Check in literature (for cleavage site reliability)

3. Remove too similar sequences
(redundancy/homology reduction)

Swiss-Prot rel. 48.5; 199 607 entries

Extensively annotated. Data checked by human experts. Kept well up-to-date. Free for academic use.

```
ID   ATPG_TOBAC     STANDARD;      PRT;   377 AA.
AC   P29790;
DT   01-APR-1993 (Rel. 25, Created)
DT   01-APR-1993 (Rel. 25, Last sequence update)
DT   01-OCT-1994 (Rel. 30, Last annotation update)
DE   ATP SYNTHASE GAMMA CHAIN, CHLOROPLAST PRECURSOR (EC 3.6.1.34).
GN   ATPC.
OS   Nicotiana tabacum (Common tobacco).
OC   Eukaryota; Viridiplantae; Embryophyta; Tracheophyta; Spermatophyta;
OC   Magnoliophyta; eudicotyledons; core eudicots; Asteridae; euasterids I;
OC   Solanales; Solanaceae; Nicotiana.
/.../
RX   MEDLINE=92322965 [NCBI, ExPASy, Israel, Japan]; PubMed=1535803;
RA   Larsson K.H., Napier J.A., Gray J.C.;
RT   "Import and processing of the precursor form of the gamma subunit of
RT   the chloroplast ATP synthase from tobacco.";
RL   Plant Mol. Biol. 19:343-349(1992).
/.../
FT   TRANSIT       1     55       CHLOROPLAST.
FT   CHAIN        56    377       ATP SYNTHASE GAMMA CHAIN.
FT   ACT_SITE    143    143       BY SIMILARITY.
FT   DISULFID    253    259       BY SIMILARITY.
SQ   SEQUENCE   377 AA;  41446 MW;  60A262F08013F3E0 CRC64;
MSCSNLTMLV SSKPSLSDSS ALSFRSSVSP FQLPNHNTSG PSNPSRSSSV TPVHCGLRDL
RDRIESVKNT QKITEAMKLV AAAKVRRAQE AVVGARPFSE TLVEVLYNIN EQLQTDDIDV
PLTKVRPVKK VALVVVTGDR GLCGGFNNYL IKKAEARIRD LKALGIDYTI ISVGKKGNSY
FIRRPYIPVD KFLEGSNLPT AKDAQAIADD VFSLFVSEEV DKVELLYTKF VSLVKSEPVI
HTLLPLSPKG EICDINGNCV DAANDEFFRL TTKEGKLTVE RDIIRTKTTD FSPILQFEQD
PVQILDALLP LYLNSQILRA LQESLASELA ARMSAMSSAT DNATELKKNL SRVYNRQRQA
KITGEILEIV AGADALV
```

Removing too similar sequences

1. Pairwise alignment (Smith-Waterman, PAM250)

2. Compare distribution of scores to extreme value distribution to get a similarity cutoff

3. Count the "neighbours" of each protein (*i.e.* pair-wise alignment score > cutoff)

4. Remove the protein with the highest number of neighbours

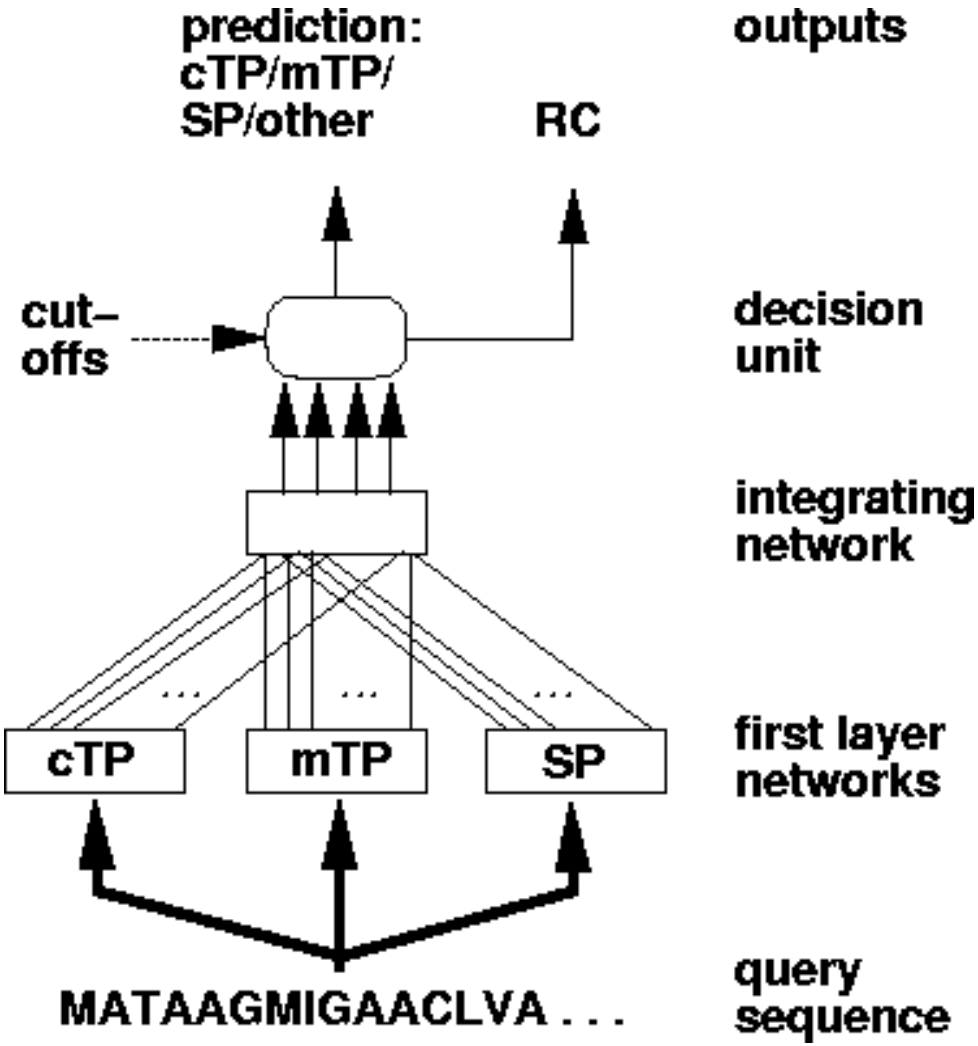
5. Repeat (3) and (4) until no neighbours left

TargetP data sets

(number of proteins before/after redundancy reduction)

Plant sets			Non-plant sets		
cTP	432	141		-	
mTP	658	368	mTP	702	371
SP	648	269	SP	2292	715
other	751	162	other	6311	1652

The neural networks of TargetP



<u>Some technical details:</u>

Logistic neurons

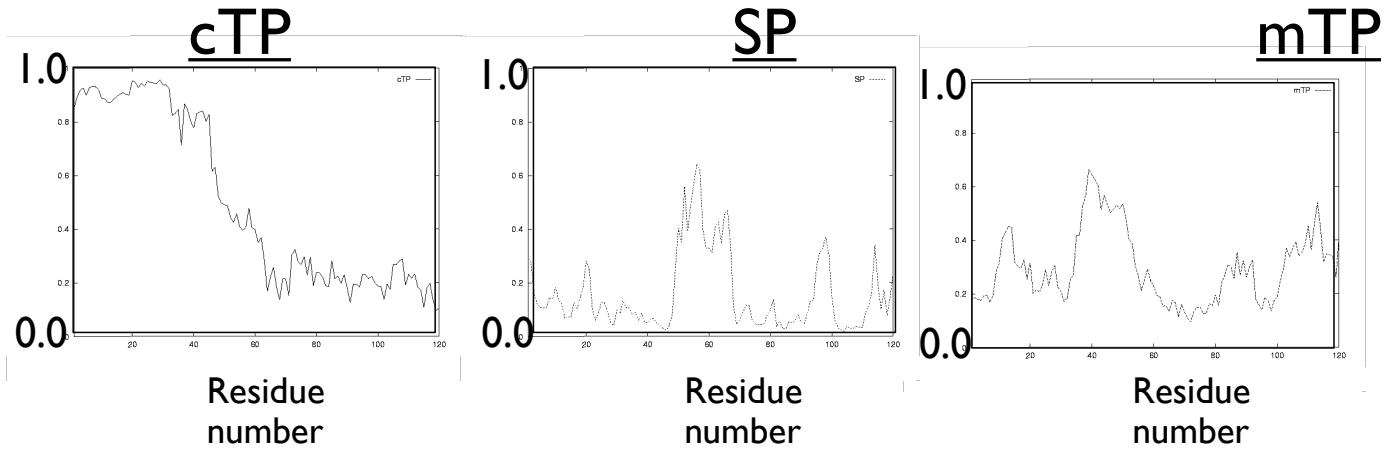
Feedforward connections

Trained using error back-propagation and 5-fold cross-validation

An "integrating network" used for post-processing

Neural network output

-The output of the first layer networks looks like this:



- This is the input to the integrating network.

TargetP output

```
### ### ###  T A R G E T P  1.1  prediction results  ### ### ###

# Number of input sequences:   4
# Cleavage site predictions included.
# Using PLANT networks.

#                Name        Length      cTP   mTP   SP   other    Loc  RC  TPlen
#-------------------------------------------------------
               P11043         516      0.873  0.012  0.004  0.320        C    3     65
               P07505         222      0.977  0.015  0.002  0.046        C    1     67
               P12352          97      0.397  0.555  0.014  0.150        M    5     40
               P48786         688      0.199  0.070  0.067  0.822        _    2     -
#-------------------------------------------------------
# cutoff                                 0.00       0.00       0.00     0.00
```

The training of TargetP

For each of the NNs, several parameters were tested:

- learning rate: 0.001, 0.05, 0.01

- sliding window size: 7-55 residues

- number of epochs: 200-1000

- number of nodes in hidden layer (0-10)

... and the best performing parameter combination was chosen

TargetP results

Plant test set (redundancy reduced):
940 proteins.
In total 85.3% correct.

MCC:

cTP	0.72
mTP	0.77
SP	0.90
other	0.77

	predicted				
	cTP	mTP	SP	other	*sens.*
cTP	120	14	2	5	*0.85*
mTP	41	300	9	18	*0.82*
SP	2	7	245	15	*0.91*
other	10	13	2	137	*0.85*
spec.	*0.69*	*0.90*	*0.96*	*0.78*	

Summary

- Pattern recognition is important to:
  - Detect non-homologous patterns in sequences
  - Analyze complex data
- Machine learning methods
- Automatic methods to learn from examples
- Important with good training and test sets
  - Jack-knifing, cross-validations
  - The problem of overtraining

Implementations

- General Toolboxes
  - Weka
  - R
  - Matlab
- SVM
  - SVMlight
  - linSVN
  - PyML
- Random Forests
  - C4.5
- DBNs
  - Graphical Models Toolkit
- HMM
  - HMMer
  - SAM