Advanced Methods for Sequence Analysis in Bioinformatics

Gunnar Rätsch[1], Cheng Soon Ong[1,2], Petra Philips[1]

[1] Friedrich Miescher Laboratory, Tübingen
[2] MPI for Biological Cybernetics, Tübingen

Vorlesung WS 2007/2008
Eberhard-Karls-Universität Tübingen

16 October 2007

http://www.fml.mpg.de/raetsch/lectures/amsa07

Official Information

AMSA, WS 07/08 (Masterstudiengang Bioinformatik)

People

Gunnar Rätsch, Cheng Soon Ong, Petra Philips
Contact: amsa07@tuebingen.mpg.de

Course Website

http://www.fml.mpg.de/raetsch/lectures/amsa07

Practical Sessions

4 times X 3 hours: Irregular times and places!
→ see lecture flyer and webpage

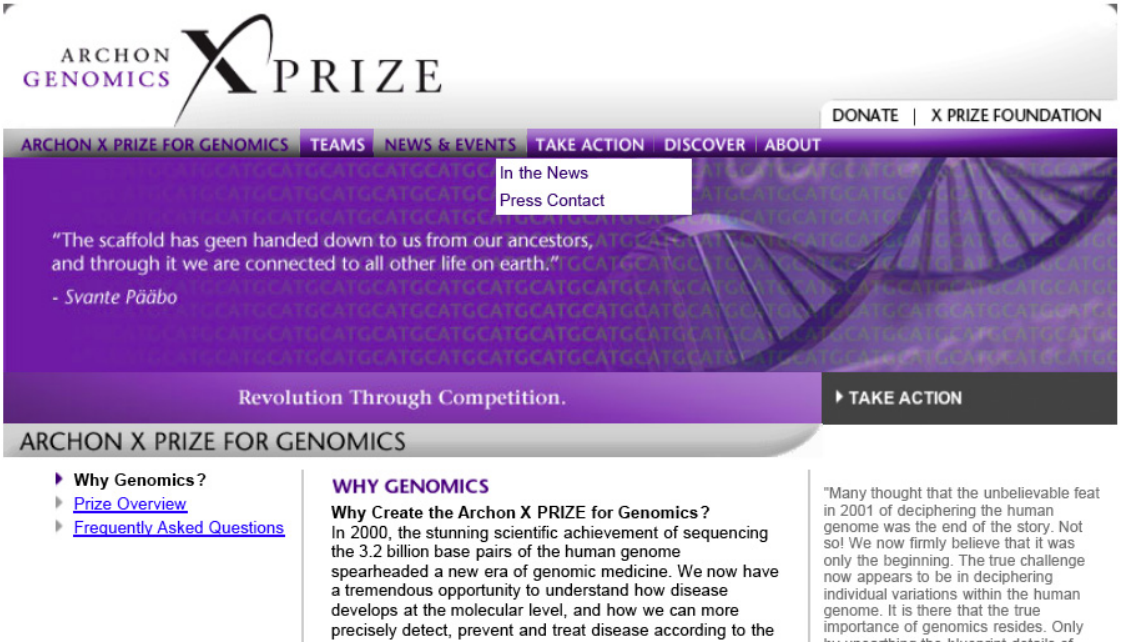
Exam

Oral exam in mid Feb 2008, register by 31 Jan 2007.

Why bioinformatics?

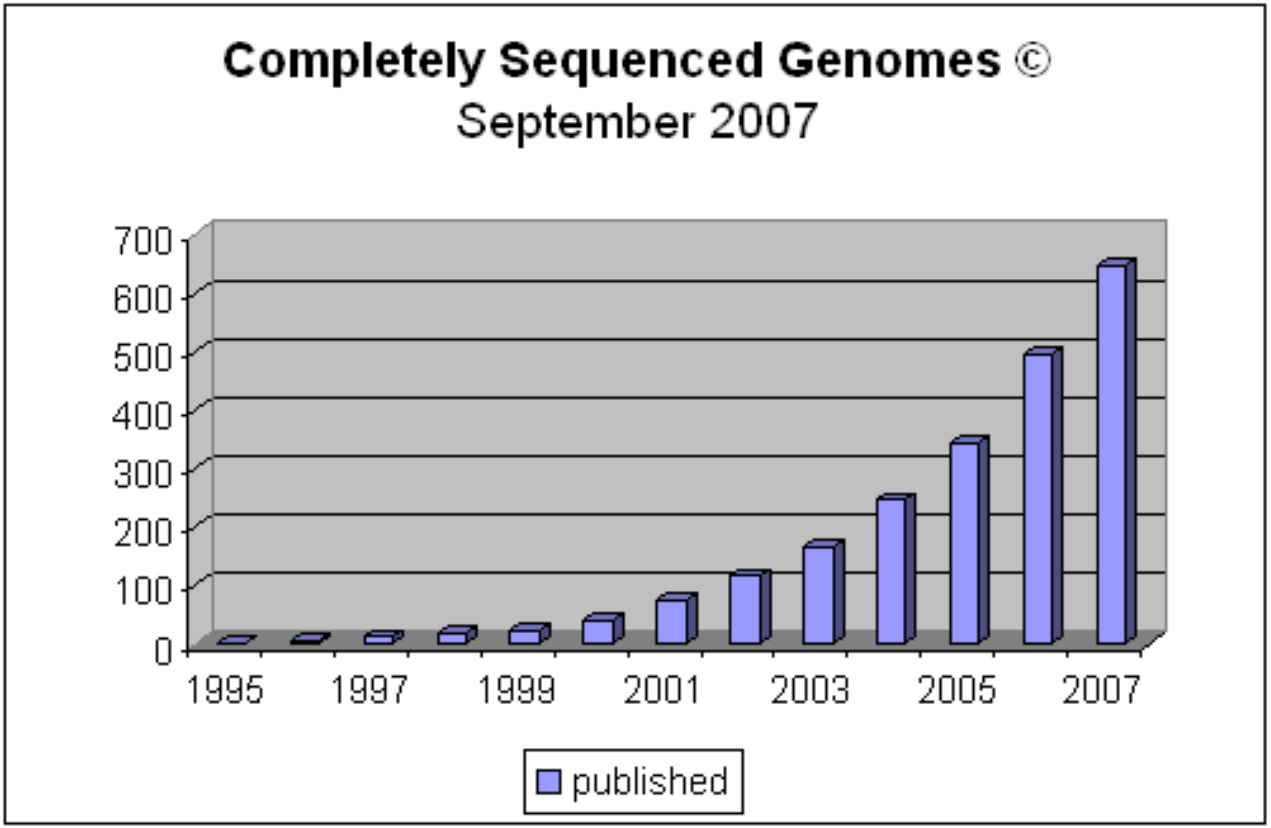


USD$10 Million

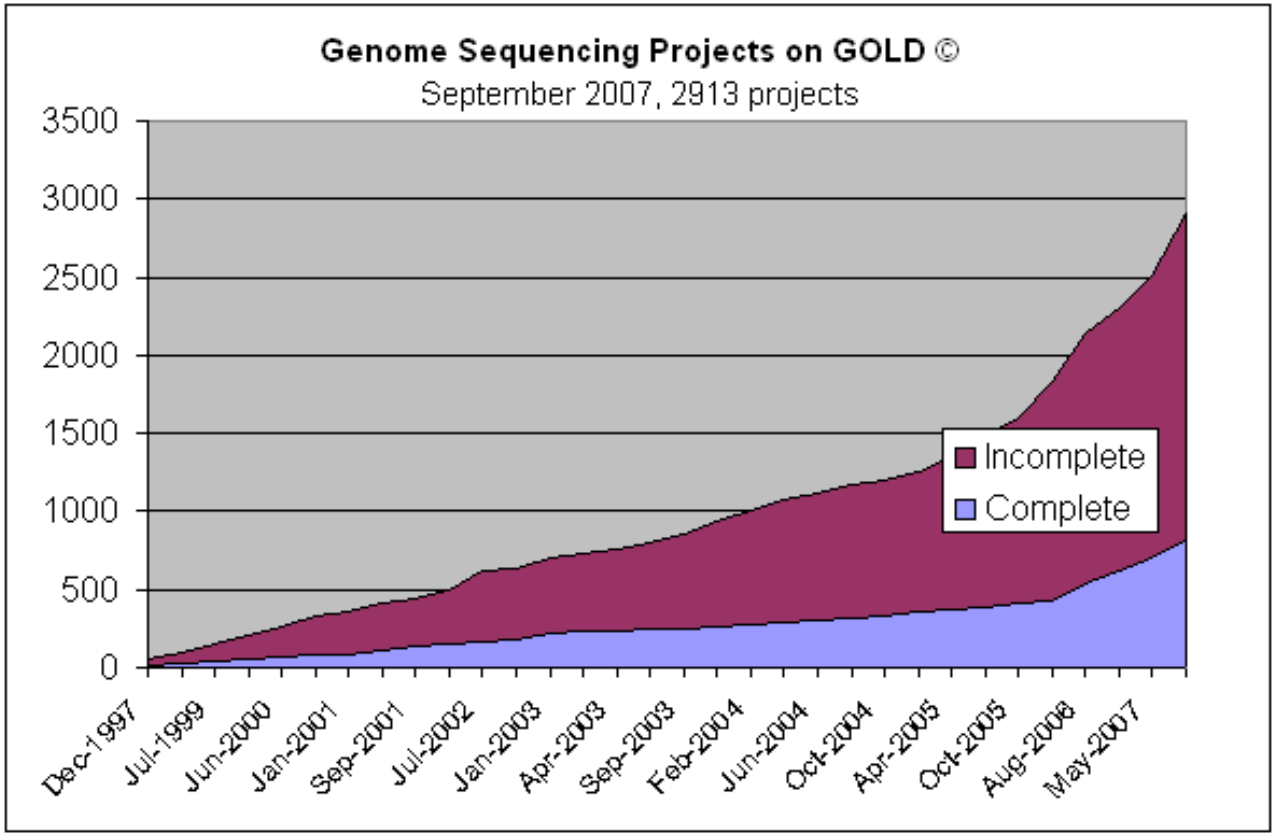


- Sequence 100 human genomes in 10 days.
- 3 billion base pairs per genome.
- 60GB of data.

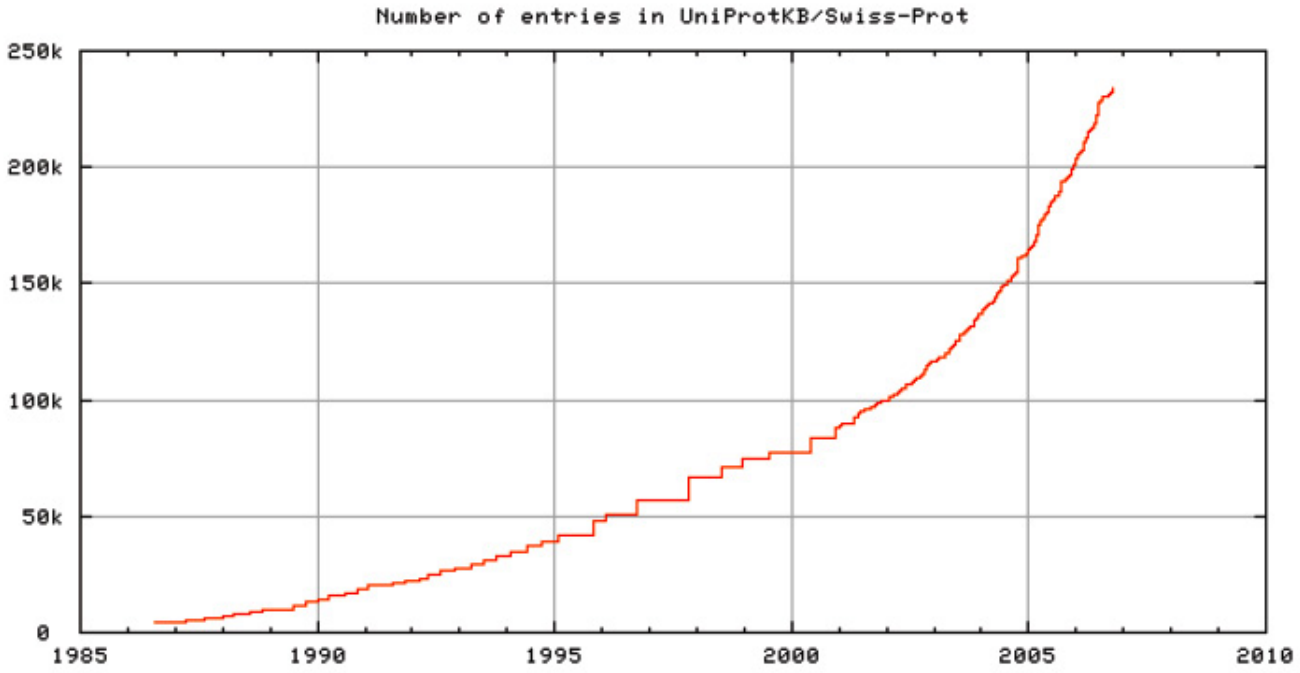
Current sequencing efforts



There's more to come



Protein Sequences



- Protein structures difficult to obtain
- Proteins interact with the cell

Genes

- Fragments of the genome sequence coding for one protein [Alberts et al., 2002]
- Estimated no. of human genes: $\approx$ 20,000-25,000
- Protein-coding DNA: 2% of the overall genome

But

- One gene can code for several protein products.
- Genes can overlap.
- Genes can code only for RNA. (80% of DNA expressed - see E. Pennisi, Science, June 2007)

Genetic code is complex and far away from being completely understood!

Why machine learning?

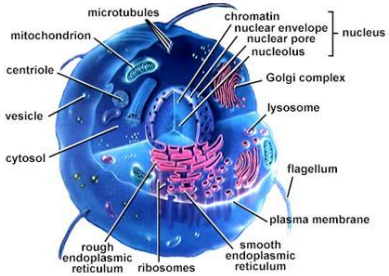


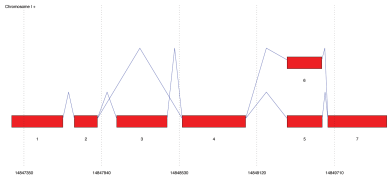
- A lot of data
- Data is noisy
- No clear biological theory
- Large number of features
- Complex relationships

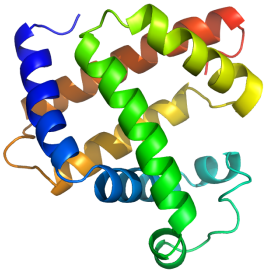
Let the data do the talking!

Sequence Analysis

- Annotate whole sequence
  - Protein Subcellular Localization
  - Motif finding
- Annotate positions
  - Gene finding
  - Splice form prediction
- Alignment
  - EST to DNA
  - Remote Homology
- Infer Structure
  - RNA secondary structure
  - Protein secondary structure



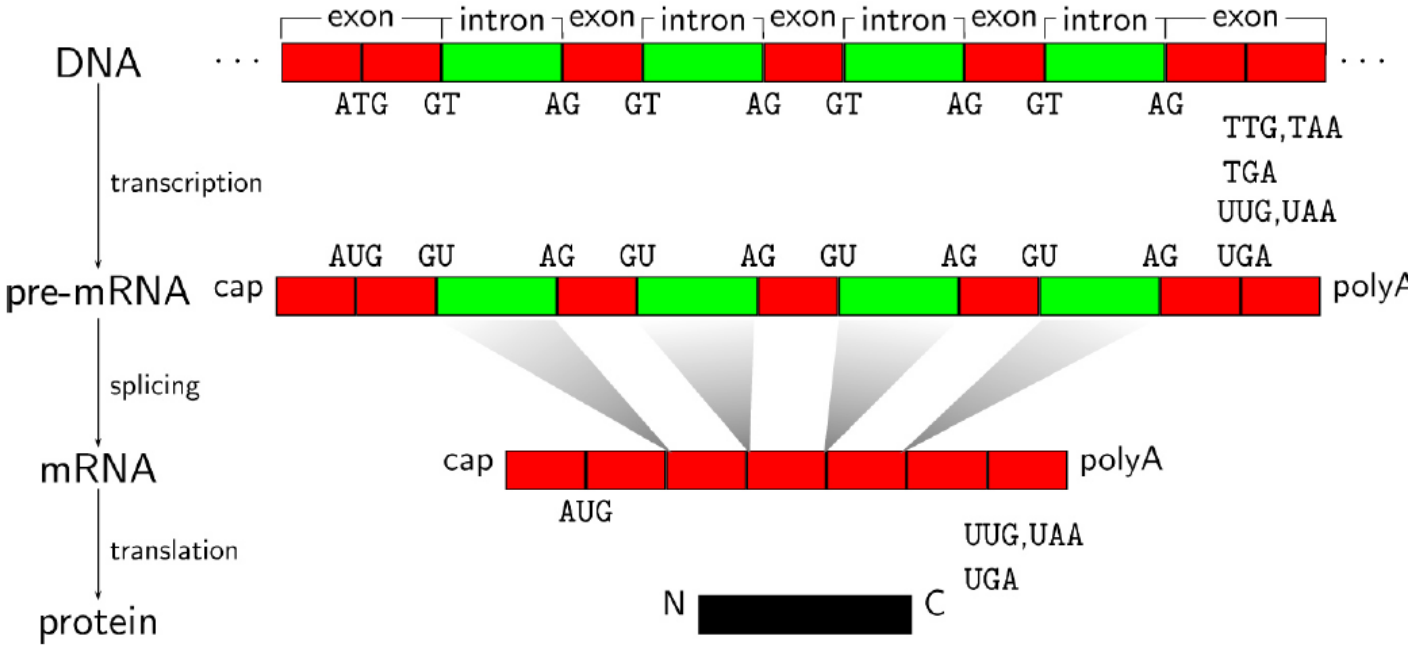




Course Outline

- Machine Learning
- Support Vector Machines and Kernel Methods
- Sequence and Graph Kernels
- Interpreting Classifiers
- Structured Output Learning
- State-of-the-art Applications

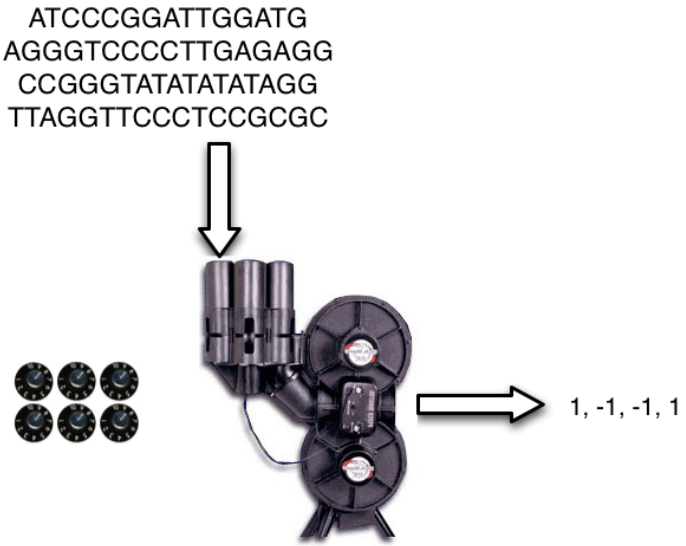
Running Example: Splicing



Classification of Sequences

Example: Recognition of splice sites

- Every 'AG' is a possible acceptor splice site
- Computer has to learn what splice sites look like
  - given some known genes/splice sites . . .
- Prediction on unknown DNA



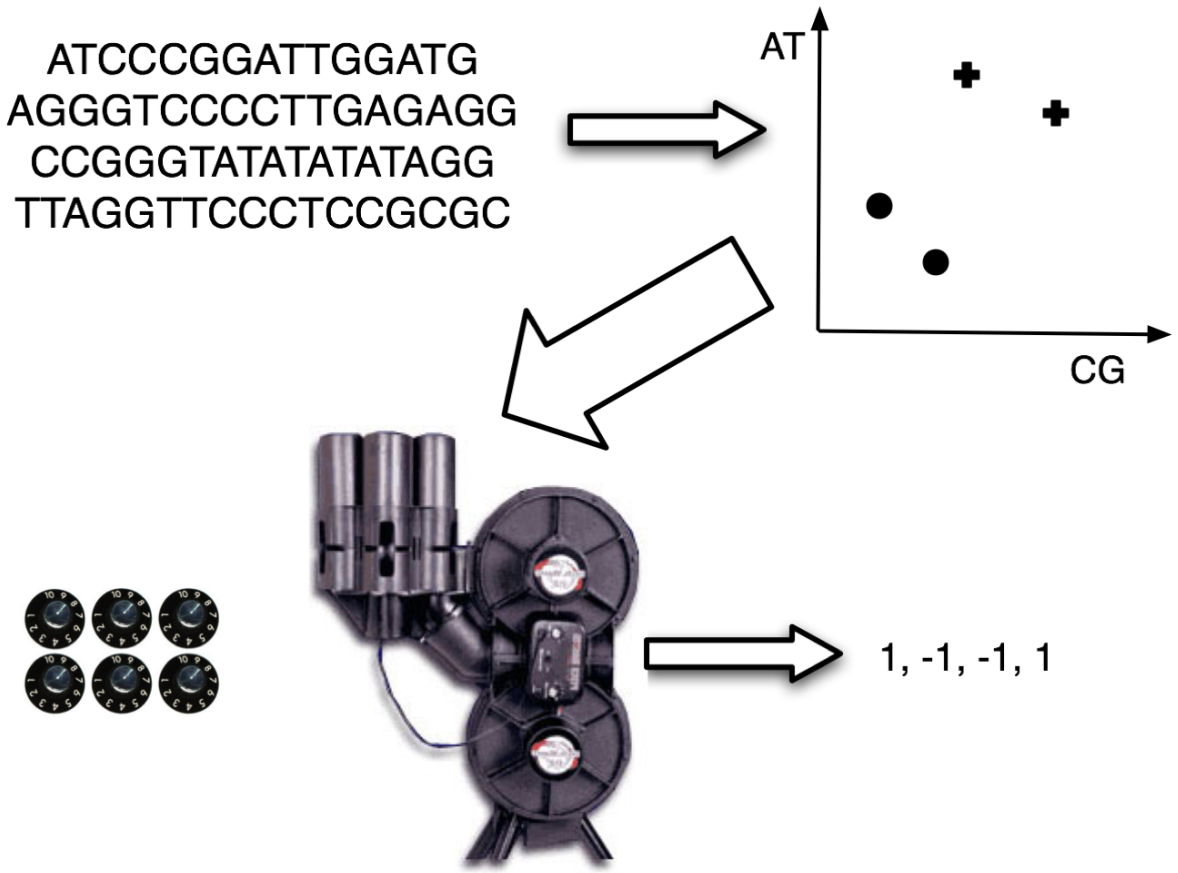
From Sequences to Features

- Many algorithms depend on numerical representations.
  - Each example is a vector of values (features).
- Use background knowledge to design good features.



	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$	$\mathbf{x}_5$	$\mathbf{x}_6$	$\mathbf{x}_7$	$\mathbf{x}_8$	...
GC before	0.6	0.2	0.4	0.3	0.2	0.4	0.5	0.5	...
GC after	0.7	0.7	0.3	0.6	0.3	0.4	0.7	0.6	...
AG**AG**AAG	0	0	0	1	1	0	0	1	...
TTT**AG**	1	1	1	0	0	1	0	0	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱
Label	+1	+1	+1	−1	−1	+1	−1	−1	...

Numerical Representation



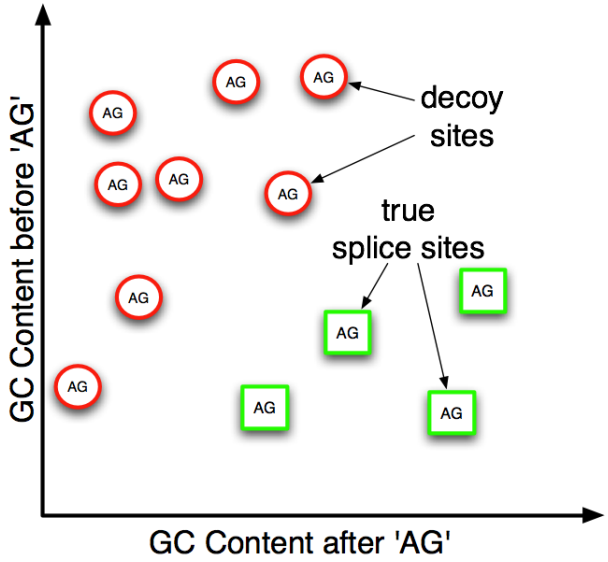
Recognition of Splice Sites

- Given: Potential acceptor splice sites

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
```

intron exon

- Goal: Rule that distinguishes true from false ones



e.g. exploit that exons have higher GC content

or

that certain motifs are located nearby

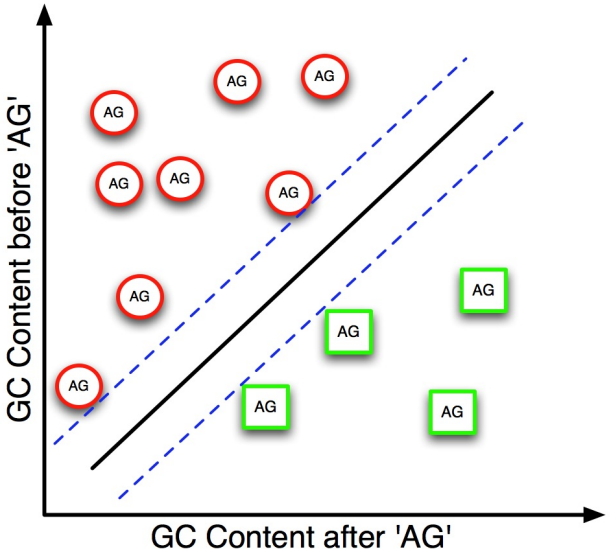
Recognition of Splice Sites

- Given: Potential acceptor splice sites

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
```

intron **exon**

- Goal: Rule that distinguishes true from false ones



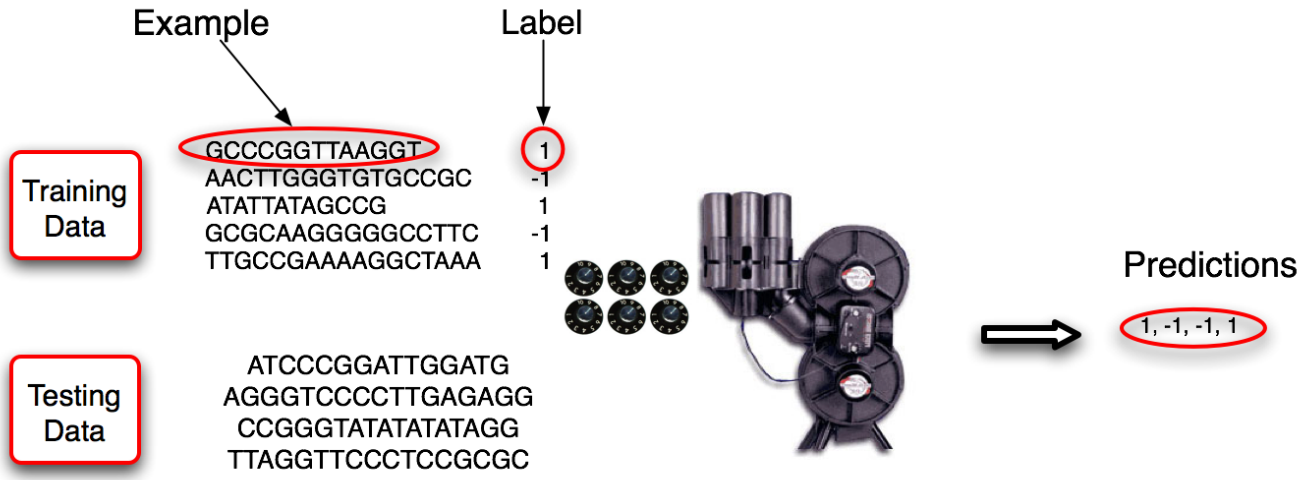
Linear classifiers
with large **margin**

Max-Margin Linear Classifiers

Advantage

- Simple.
- We can compute them.
- Good separation of **training** data.

Prediction



The machine utilizes information from training data to predict the outputs associated with a particular test example.

- Use **training** data to “train” the machine.
- Use trained machine to perform prediction on **test** data.

Generalization

Important not just to memorize the training examples!

How do we know that our estimator will perform well on **unseen** data?

Statistical Learning Theory

Assumption

Future examples are **similar** to seen labeled examples.

Max-Margin Linear Classifiers

Good separation of **training** data implies provably good separation of **new** data.

All these are in a probabilistic sense...

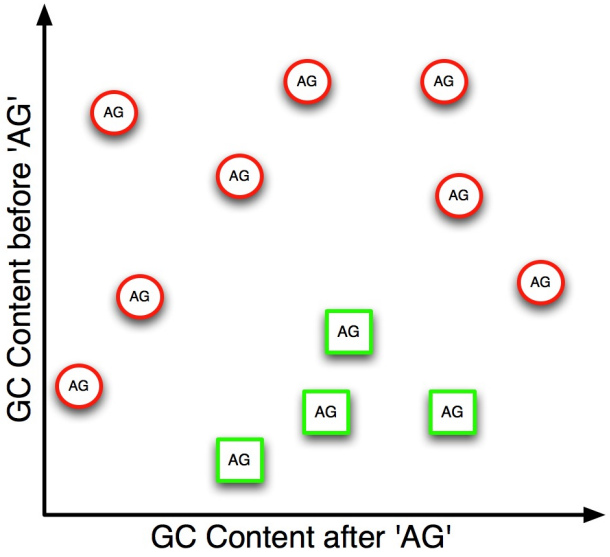
But...

- Given: Potential acceptor splice sites

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
```

intron **exon**

- Goal: Rule that distinguishes true from false ones



More realistic problem!?

- Not linearly separable!
- Need nonlinear separation!?
- Need more features!?

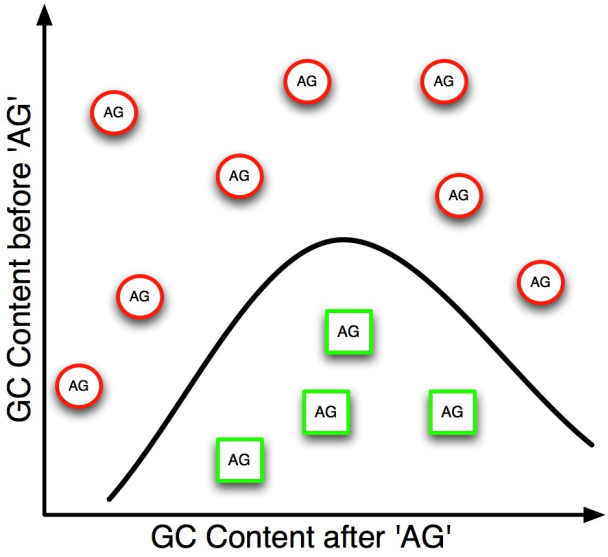
Nonlinear Separation

- Given: Potential acceptor splice sites

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
```

intron **exon**

- Goal: Rule that distinguishes true from false ones



More realistic problem!?

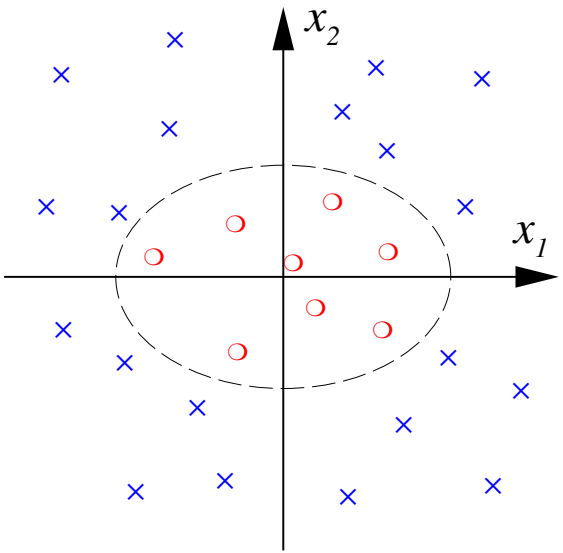
- Not linearly separable!
- Need nonlinear separation!?
- Need more features!?

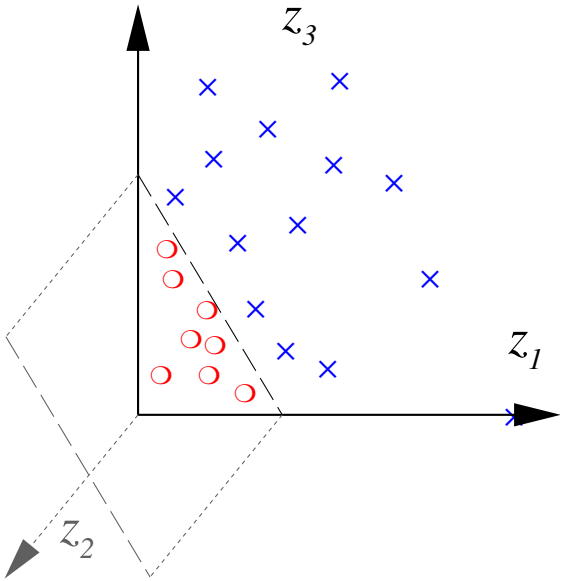
Nonlinear Algorithms in Feature Space

Idea: Map into a higher dimensional feature space where one can separate linearly.

Example: all second order monomials

$$\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$





Kernel "Trick"

- For special maps Φ (**kernels**) and special feature spaces (with **scalar products**) the scalar products in feature space (here $\mathbb{R}^3$) can be computed directly in the input space (here $\mathbb{R}^2$)!
- This also works for higher orders and dimensions
  $\Rightarrow$ relatively low dimensional input spaces
  $\Rightarrow$ very high dimensional feature spaces

We can still think of linear classifiers!

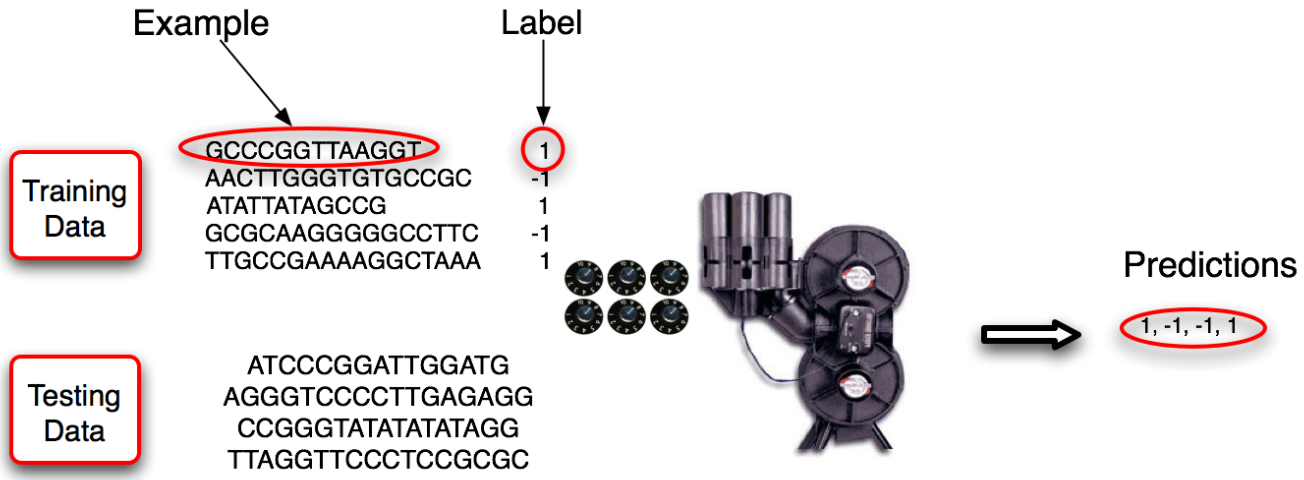
The art consists in designing the 'right' kernels!

Max-Margin Linear Classifiers

Advantage

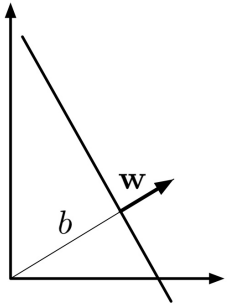
- Simple.
- We can compute them.
- Good separation of **training** data.
- Good separation of **unseen** data.

Computing Linear Classifiers

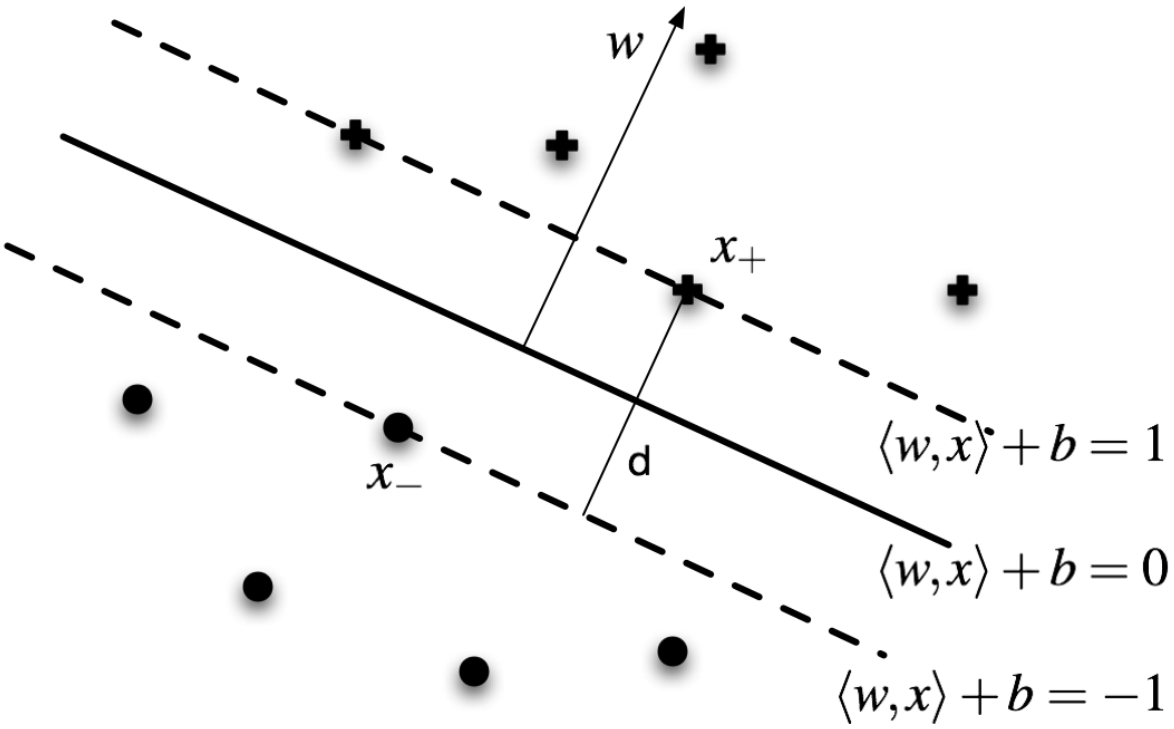


Consider linear classifiers with parameters $\mathbf{w}, b$:

$$f(\mathbf{x}) = \sum_{j=1}^{d} w_j x_j + b = \langle \mathbf{w}, \mathbf{x} \rangle + b$$

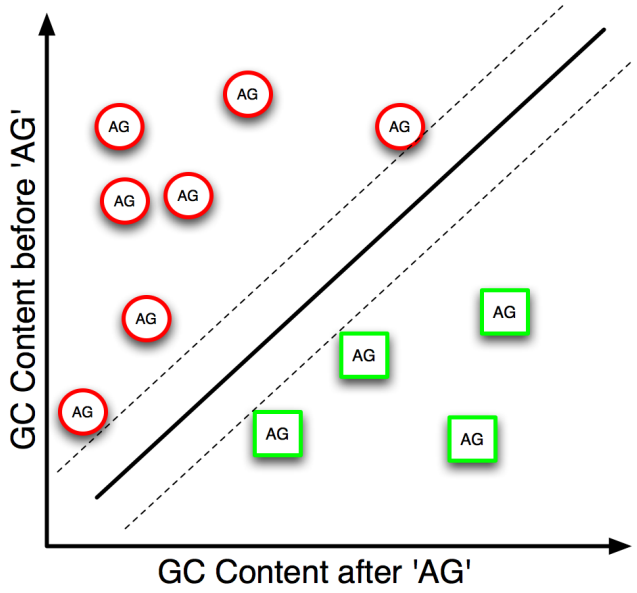


Margin Maximization



Margin maximization is equivalent to minimizing $\|\mathbf{w}\|$.

Classifier: SVM



Minimize

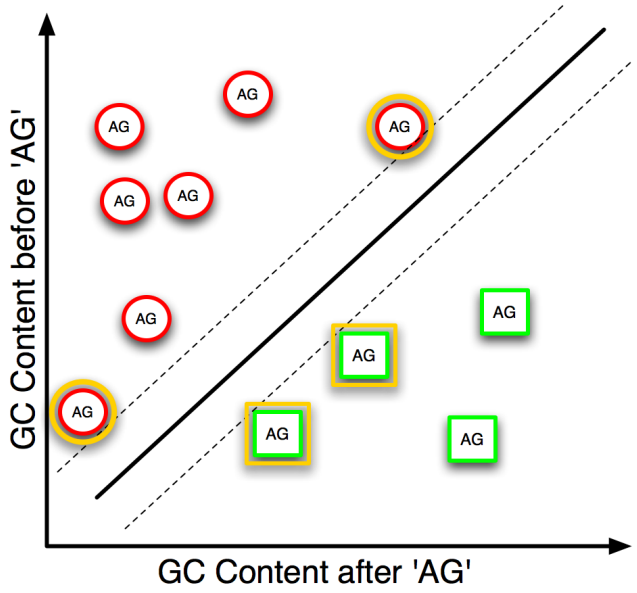
$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

- The examples on the margin are called support vectors [Vapnik, 1995]
- Called the soft margin SVM or the C-SVM [Cortes and Vapnik, 1995]

Classifier: SVM



Minimize

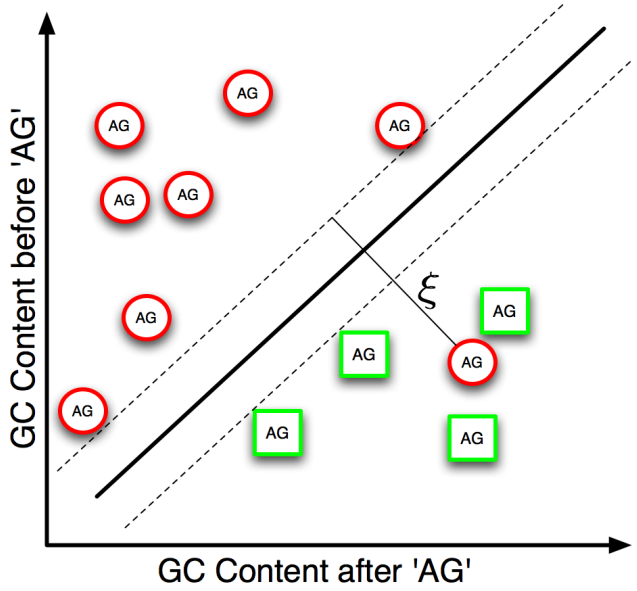
$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

- The examples on the margin are called support vectors [Vapnik, 1995]
- Called the soft margin SVM or the C-SVM [Cortes and Vapnik, 1995]

Classifier: SVM



Minimize

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

- The examples on the margin are called support vectors [Vapnik, 1995]
- Called the soft margin SVM or the C-SVM [Cortes and Vapnik, 1995]

SVM is dependent on training data

Minimize

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

Representer Theorem

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i\mathbf{x}_i$$

Minimize

$$\frac{1}{2}\sum_{i,j}^{N}\alpha_i\alpha_j\langle \mathbf{x}_i, \mathbf{x}_j\rangle + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i(\langle \sum_{j=1}^{N}\alpha_j\mathbf{x}_j, \mathbf{x}_i\rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

SVM solution only depends on scalar products between examples ($\rightsquigarrow$ kernel trick)

SVM is dependent on training data

Minimize

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

Representer Theorem

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i\mathbf{x}_i$$

Minimize

$$\frac{1}{2}\sum_{i,j}^{N}\alpha_i\alpha_j\langle \mathbf{x}_i, \mathbf{x}_j\rangle + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i(\langle \sum_{j=1}^{N}\alpha_j\mathbf{x}_j, \mathbf{x}_i\rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

SVM solution only depends on scalar products between examples ($\rightsquigarrow$ kernel trick)

SVM is dependent on training data

Minimize

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

Representer Theorem

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i\mathbf{x}_i$$

Minimize

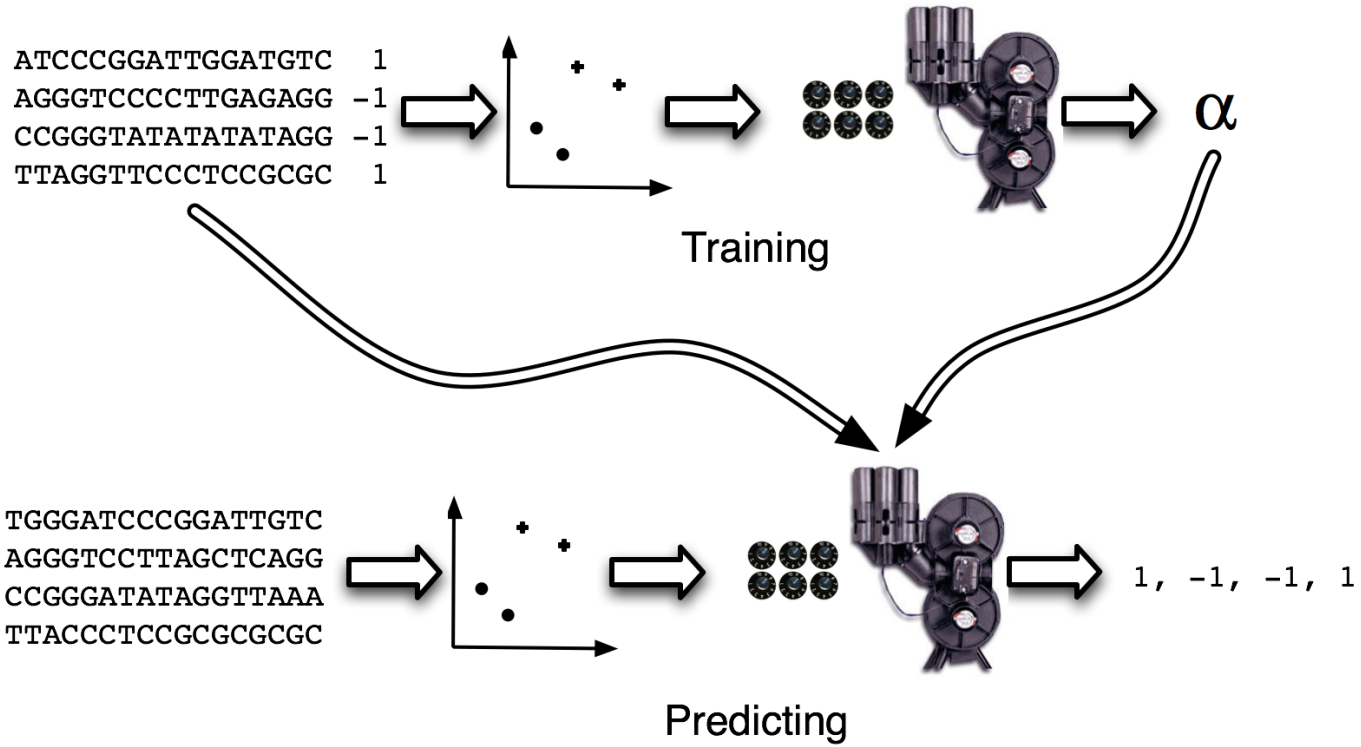
$$\frac{1}{2}\sum_{i,j}^{N}\alpha_i\alpha_j\langle \mathbf{x}_i, \mathbf{x}_j\rangle + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i(\langle \sum_{j=1}^{N}\alpha_j\mathbf{x}_j, \mathbf{x}_i\rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

SVM solution only depends on scalar products between examples ($\rightsquigarrow$ kernel trick)

Summary: Prediction



Duda et al. [2001], Schölkopf and Smola [2002], Shawe-Taylor and Cristianini [2004]

Useful to Review

Basics of

- Linear algebra: dot product, metric (euclidean distance)
- Optimization: gradient descent, convex functions
- Probability: joint, conditional, marginal probability

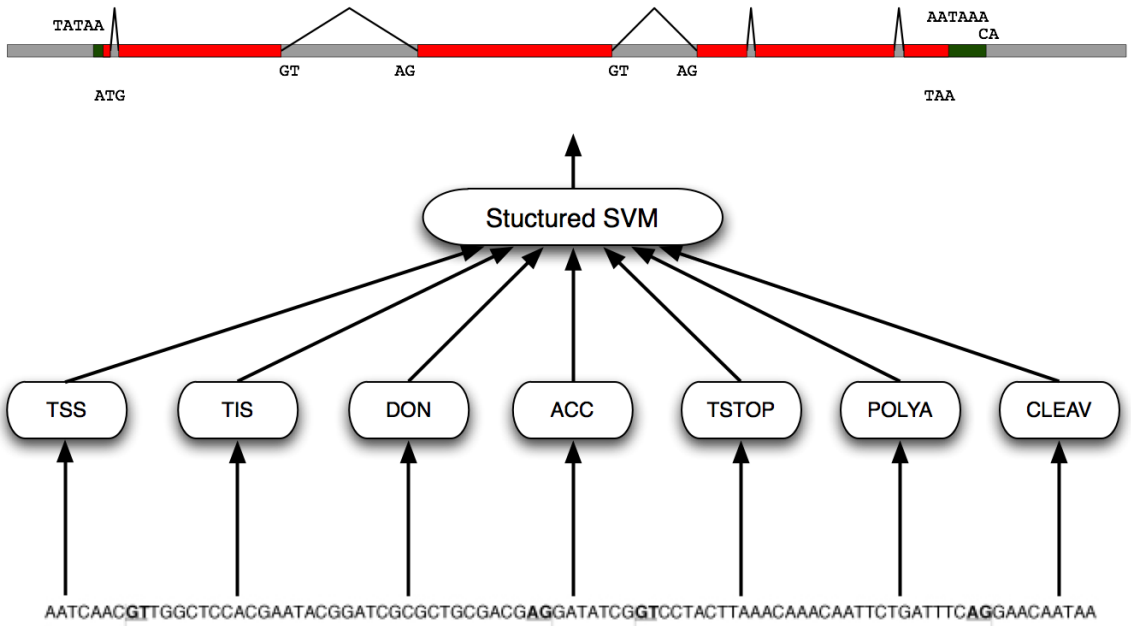
At the end of the semester

you will understand

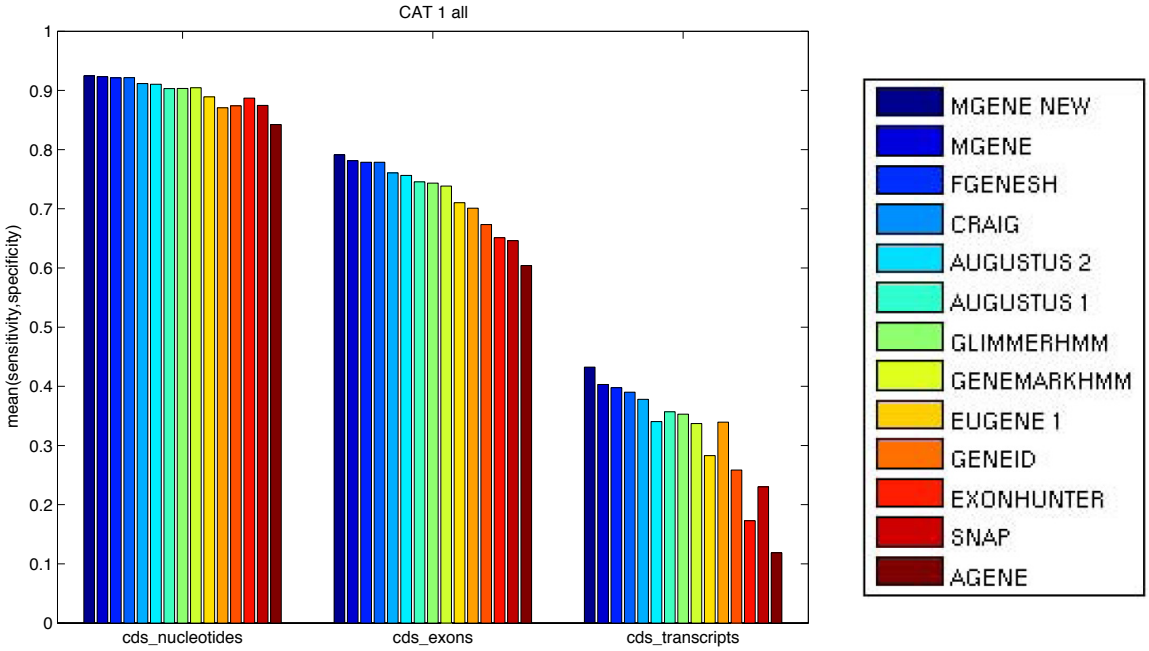
- what an SVM and kernel methods are
- how they work
- why they work
- how to employ them for sequence analyses
- how they compare to classical sequence analysis algorithms
- and what cutting edge applications they are used for

Outlook

mGene: An ab initio gene finding system



nGASP Competition



Sensitivity and specificity on coding nucleotide, exon and transcript level for *ab initio* gene finding.

References

C. Cortes and V.N. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

R.O. Duda, P.E.Hart, and D.G.Stork. *Pattern classification*. John Wiley & Sons, second edition, 2001.

W.S. Noble. What is a support vector machine? *Nature Biotechnology*, 2006.

E. Pennisi. Dna sudy forces rethink what it means to be a gene. *Science*, 316, June 2007.

B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

V.N. Vapnik. *The nature of statistical learning theory*. Springer Verlag, New York, 1995.