Machine Learning & Application in Biology

HyunJung (Helen) Shin

shin@tuebingen.mpg.de

Friedrich Miescher Laboratory
Max-Planck-Society, Tuebingen, Germany

European School of Genetic Medicine, 6th Courses in Bioinformatics for Molecular Biologist,
Bertinoro di Romagna, GSF-National Research Center for Environment and Health

Goal of Lecture

Machine Learning can alleviate the burden of solving many biological problems,

- saving the time and cost required for experiments
- providing predictions that guide new experiments.

Goal of Lecture

Machine Learning can alleviate the burden of solving many biological problems,

- saving the time and cost required for experiments

- providing predictions that guide new experiments.

The goal of this tutorial is to raise awareness and comprehension of machine learning

so that biologists can properly match the task at hand to the corresponding analytical approach

Abstract

We explore representative models, from traditional statistical models to recent machine learning models,

presenting several up-to-date research projects in bioinfomatics to exemplify how biological questions can benefit from a machine learning approach.

Content

1. Basics
2. Tasks
3. Learning
4. Models with Examples
5. ~~Evaluation and Statistical Tests~~

Content

1. Basics
2. Tasks
3. Learning
4. Models with Examples

Basics

Data Representation

Data Table (Data Base)

20 records,
20 samples,
20 observations,
20 objects,
20 data points,
20 individuals,
20 experimental units,
etc.

or

	A_1	A_2	A_3	…	A_{10}	y	y
x_1	*10*	*5*	*red*	…	*1000*	*class1*	*1*
x_2	*6*	*6*	*blue*	…	*3500*	*class2*	*20*
x_3	*7*	*7*	*yellow*	…	*400*	*class1*	*45*
…	…	…	…	…	…	…	…
x_{18}	*3*	*56*	*red*	…	*0*	*class2*	*30*
x_{19}	*15*	*62*	*red*	…	*500*	*class1*	*100*
x_{20}	*3*	*88*	*blue*	…	*700*	*class2*	*3*

Data Representation

A_j

attribute,
feature,
descriptor,
input variable,
predictor variable,
independent variable,
exogeneous variable,
etc,

or

	A_1	A_2	A_3	…	A_{10}	y	y
x_1	10	5	red	…	1000	class1	1
x_2	6	6	blue	…	3500	class2	20
x_3	7	7	yellow	…	400	class1	45
…	…	…	…	…	…	…	…
x_{18}	3	56	red	…	0	class2	30
x_{19}	15	62	red	…	500	class1	100
x_{20}	3	88	blue	…	700	class2	3

Data Representation

x_i

input,
predictor,
etc.

or

	A_1	A_2	A_3	…	A_{10}	y	y
x_1	10	5	red	…	1000	class1	1
x_2	6	6	blue	…	3500	class2	20
x_3	7	7	yellow	…	400	class1	45
…	…	…	…	…	…	…	…
x_{18}	3	56	red	…	0	class2	30
x_{19}	15	62	red	…	500	class1	100
x_{20}	3	88	blue	…	700	class2	3

* *Input set:* $X = \{x_1, x_2, \ldots, x_{20}\}$

Data Representation

y_i

output variable,
response,
target variable,
endogeneous variable,
label,
etc.

or

	A_1	A_2	A_3	…	A_{10}	*y*	*y*
x_1	*10*	*5*	*red*	…	*1000*	*class1*	*1*
x_2	*6*	*6*	*blue*	…	*3500*	*class2*	*20*
x_3	*7*	*7*	*yellow*	…	*400*	*class1*	*45*
…	…	…	…	…	…	…	…
x_{18}	*3*	*56*	*red*	…	*0*	*class2*	*30*
x_{19}	*15*	*62*	*red*	…	*500*	*class1*	*100*
x_{20}	*3*	*88*	*blue*	…	*700*	*class2*	*3*

* *Output(Target) Set:* $Y = \{y_1, y_2, \ldots, y_{20}\}$

Content

1. Basics
2. Tasks
3. Learning
4. Models with Examples

Tasks

Tasks

- Prediction
  - Classification
  - Regression
- Description
  - Clustering
  - Feature Description
- Dimensionality Reduction
  - Feature Selection
  - Feature Extraction
- Data Reduction (Sample Selection)
- Data Integration

Classification

Classification is concerned with the problem of **separating** distinct sets of data points and **allocating** new (test or unknown) data points to previously defined group (class)

Classification

Input Target

	A_1	A_2	A_3	…	A_{10}	y	y
x_1	10	5	red	…	1000	class1	1
x_2	6	6	blue	…	3500	class2	20
x_3	7	7	yellow	…	400	class1	45
…	…	…	…	…	…	…	…
x_{20}	3	88	blue	…	700	class2	3

Target variable (y) is categorical

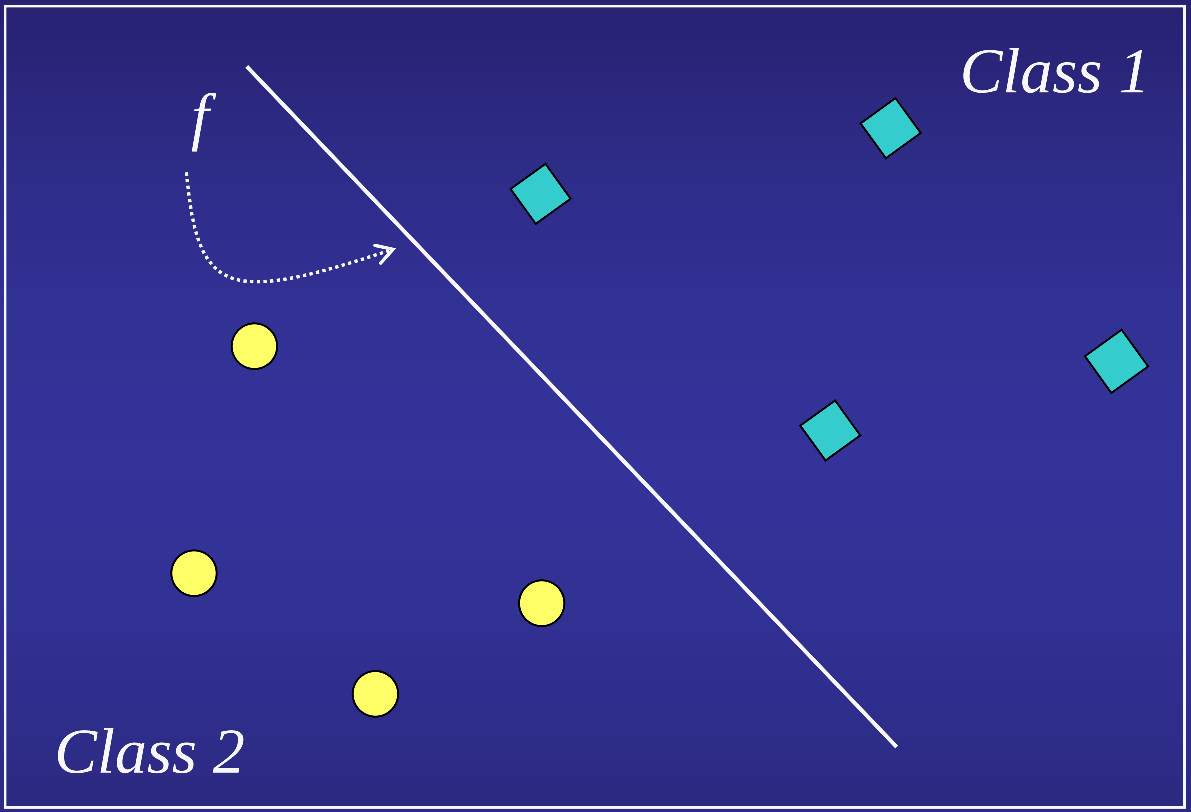
Nominal : (ex) yes or no, 1 or -1
: (ex) blue, red, yellow…
Ordinal : (ex) age groups
(10-20, 20-30, 30-40, …)

* Predicted Value

$$f \left\{ \begin{array}{l} \textit{model,} \\ \textit{function} \end{array} \right.$$

$$f(x) \left\{ \begin{array}{l} \textit{predicted value (output),} \\ \textit{output,} \\ \textit{score,} \\ \textit{etc} \end{array} \right.$$

Classification



$f(x) \approx y$

(Ex) *Class 1: y = 1*
Class 2: y = -1

$$f(x) = \begin{cases} +1 \ \text{(class1)} & \text{if } f(x) \geq 0 \\ -1 \ \text{(class2)} & \text{if } f(x) < 0 \end{cases}$$

Regression

Regression is concerned with the problem of **predicting** the value of continuous target variable.

Regression

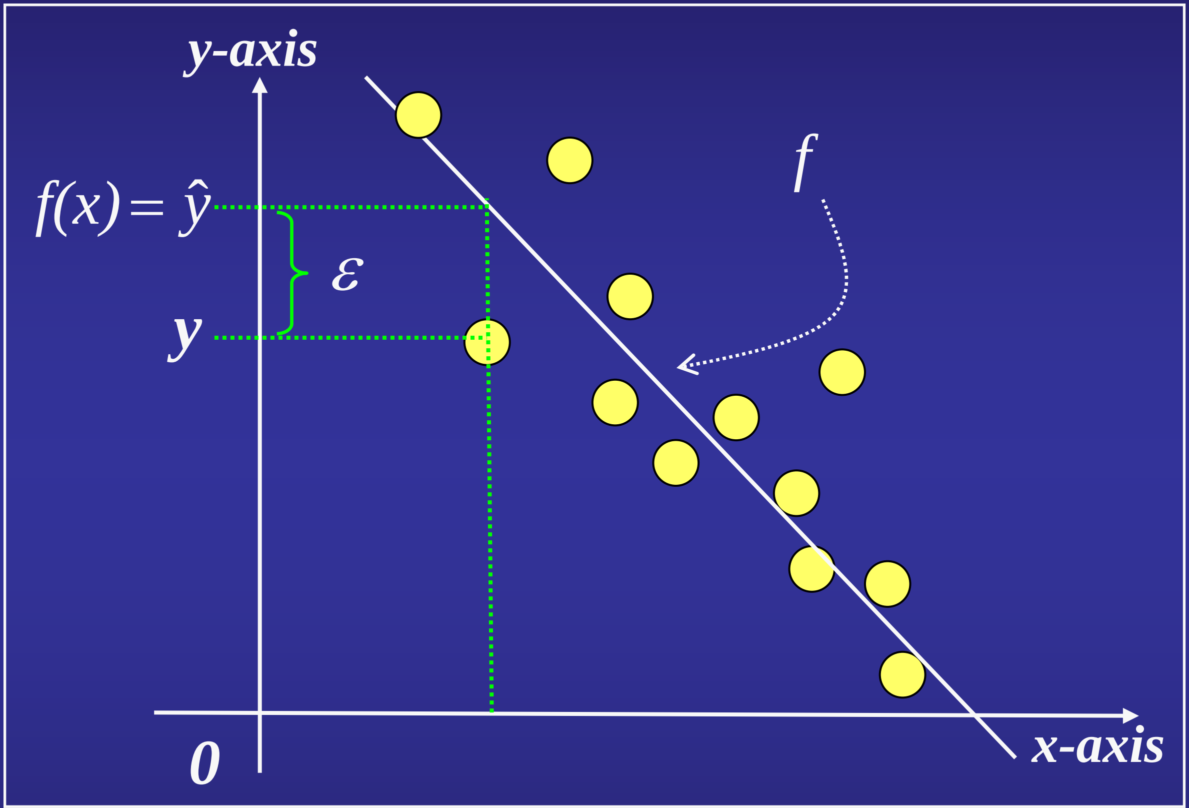
Input

Target

	A_1	A_2	A_3	…	A_{10}	y	y
x_1	10	5	red	…	1000	class1	1
x_2	6	6	blue	…	3500	class2	20
x_3	7	7	yellow	…	400	class1	45
…	…	…	…	…	…	…	…
x_{20}	3	88	blue	…	700	class2	3

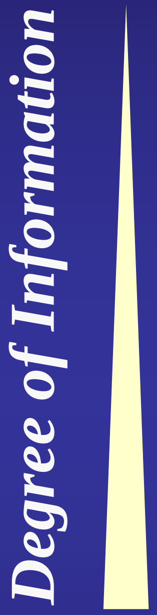
Target variable (y) is continuous

Regression



$$f(x) \approx y$$

** Scales of Variable*



- Nominal: sex, blood type, color,…
- Ordinal: rank, educational degree,…
  ≥/=/≤,

Categorical

- Interval: temperature, …
  ≥/=/≤, +/–
- Ratio: income, age, weight…
  ≥/=/≤, +/–, ×/÷

Continuous

** *Classification can be regarded as a subset of Regression in viewpoint of modeling (not task).*

Clustering

Clustering is concerned with the identification of groups of similar data points based on similarity measures.

Clustering

Clustering is concerned with the identification of groups of similar data points based on similarity measures.

Clustering is distinct from classification in that

- Classification pertains to a known number of groups and its operational objective is to assign new data points to one of these groups

- Clustering makes no assumption concerning the number of groups

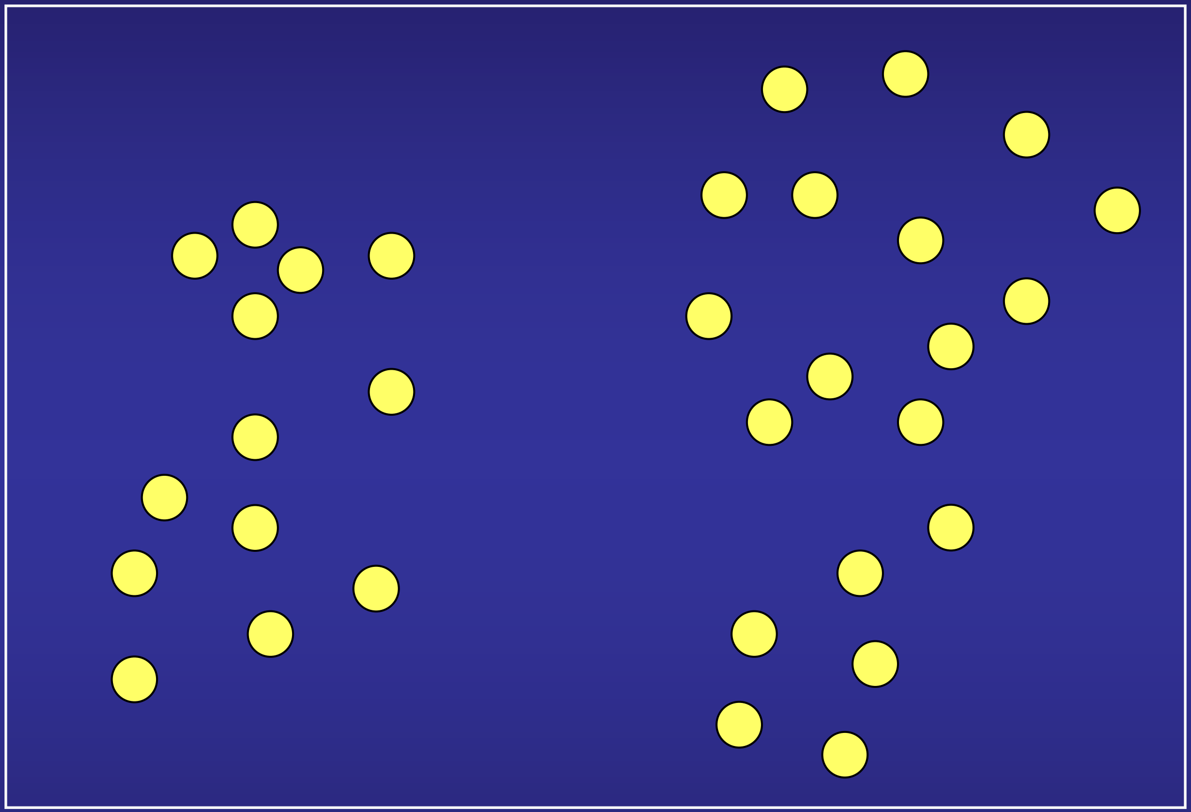
Clustering

Input **No Target !!**

	A_1	A_2	A_3	…	A_{10}
x_1	10	5	red	…	1000
x_2	6	6	blue	…	3500
x_3	7	7	yellow	…	400
…	…	…	…	…	…
x_{20}	3	88	blue	…	700

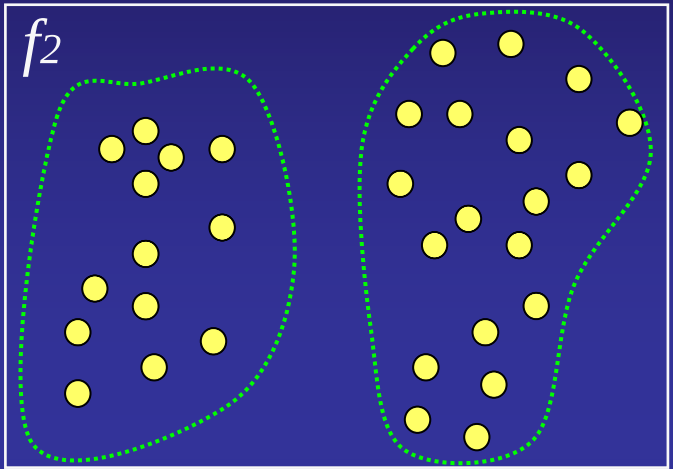
No target variable (y)

Clustering

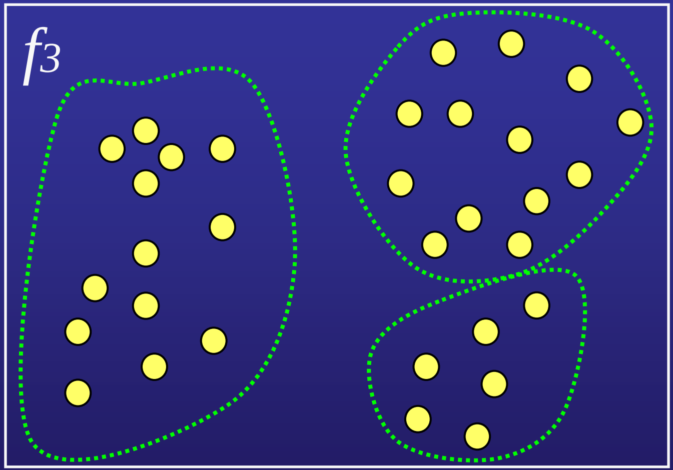


Clustering

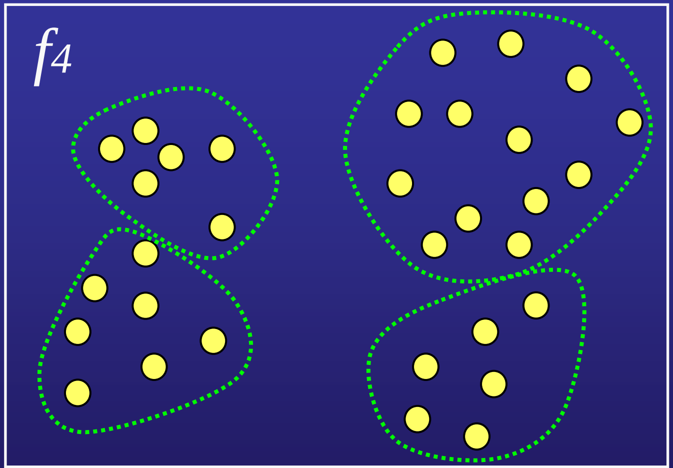
2 clusters



3 clusters



4 clusters



Dimensionality Reduction

Dimensionality reduction is concerned with the process which **removes irrelevant or redundant features (attributes)** from the original feature set, in order to avoid "curse of dimensionality"—

complication of learning process, erroneous results, computational burden.

* note: irrelevant or redundant for learning or modeling

Dimensionality Reduction

- Feature Selection

A process of finding a subset of relevant features (attributes) from the original set of features.

(Ex) Selected Features: A_1, A_{1000}

	A_1	A_2	A_3	…	A_{1000}	y
x_1	10	5	red	…	1000	1
x_2	6	6	blue	…	3500	20
x_3	7	7	yellow	…	400	45
…	…	…	…	…	…	…
x_{20}	3	88	blue	…	700	3

$$\xrightarrow{f}$$

	A_1	A_{1000}	y
x_1	10	1000	1
x_2	6	3500	20
x_3	7	400	45
…	…	…	…
x_{20}	3	700	3

Dimensionality Reduction

- Feature Extraction

A process of defining new descriptors (features) condensed via transformations of the raw features. The descriptors are represented as the features in the new feature space

(Ex) Extracted Features:
$$\begin{cases} P_1 = \beta_1 A_1 + \beta_1 A_2 + \beta_1 A_1 A_{350} \\ P_2 = \Phi(A_1, A_2, \ldots, A_{1000}) \end{cases}$$

	A_1	A_2	A_3	…	A_{1000}	y
x_1	10	5	red	…	1000	1
x_2	6	6	blue	…	3500	20
x_3	7	7	yellow	…	400	45
…	…	…	…	…	…	…
x_{20}	3	88	blue	…	700	3

$\xrightarrow{f}$

	P_1	P_2	y
x_1	-2	-0.5	1
x_2	0	0.01	20
x_3	25	0.9	45
…	…	…	…
x_{20}	10	-0.7	3

Data Reduction (Sample Selection)

Data Reduction is concerned with the process which removes irrelevant or redundant "data points" from the original data set,

in order to avoid complication of learning process or computational burden.

* note: irrelevant or redundant for learning or modeling

Data Reduction (Sample Selection)

(Ex) Selected Data Points: x_2, x_3, x_{100}, x_{9999}

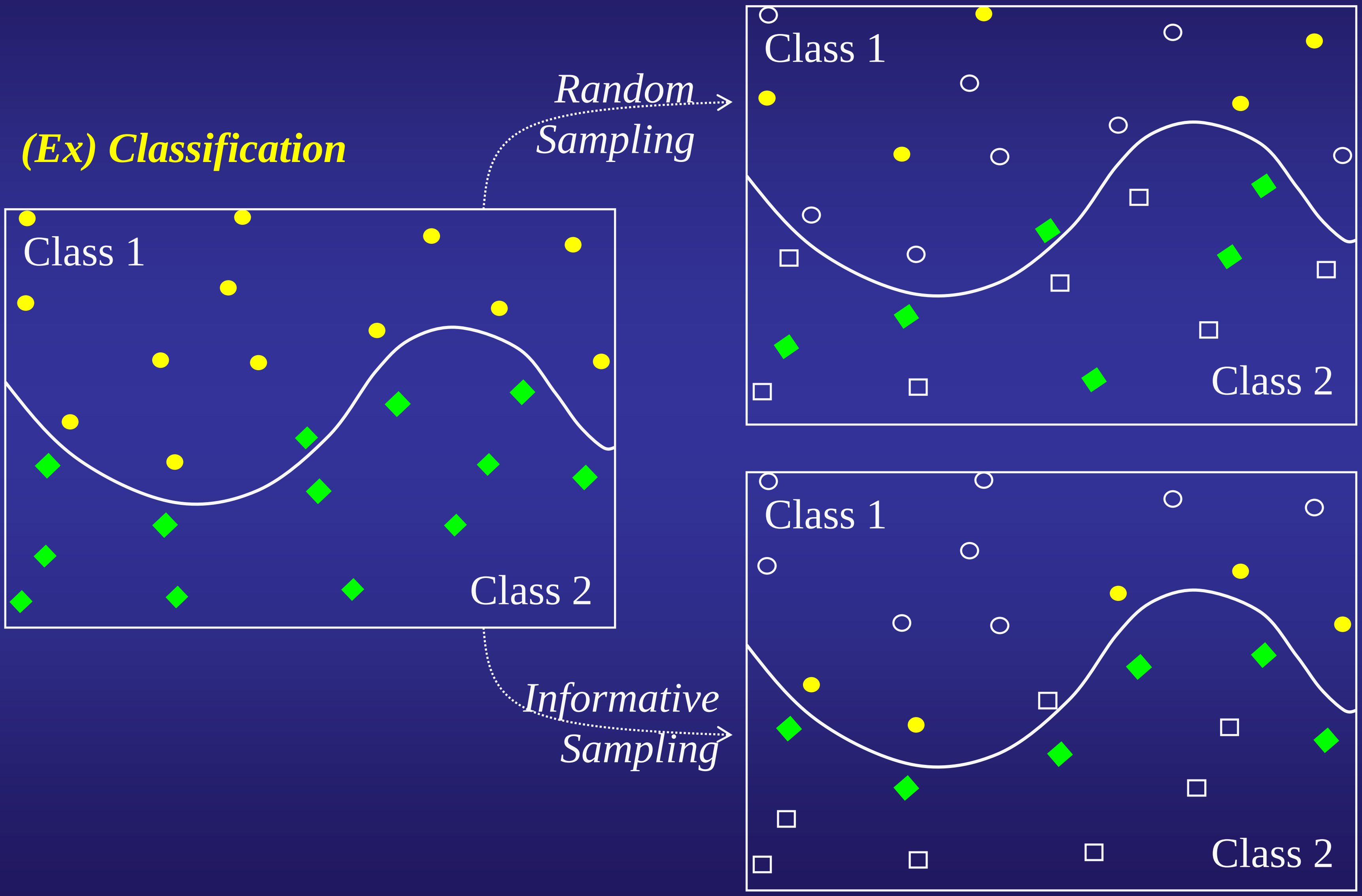
	A_1	A_2	A_3	…	A_{10}	y
x_1	10	5	red	…	1000	class1
x_2	6	6	blue	…	3500	class2
x_3	7	7	yellow	…	400	class1
…	…	…	…	…	…	…
x_{100}	3	88	blue	…	700	class1
…	…	…	…	…	…	…
x_{500}	60	68	red	…	1700	class2
…	…	…	…	…	…	…
x_{9999}	3	85	green	…	2500	class2
x_{10000}	3	1	blue	…	5700	class1

$\xrightarrow{f}$

	A_1	A_2	A_3	…	A_{10}	y
x_2	6	6	blue	…	3500	class2
x_3	7	7	yellow	…	400	class1
x_{100}	3	88	blue	…	700	class1
x_{9999}	3	85	green	…	2500	class2

Data Reduction (Sample Selection)

(Ex) Classification



Data Integration

Data Integration is concerned with the **integration of different or heterogeneous data sources** (sets) in order to enhance the total information about the problem at hand.

Each data source contains partly independent and partly complementary pieces of information about the problem.

Data Integration

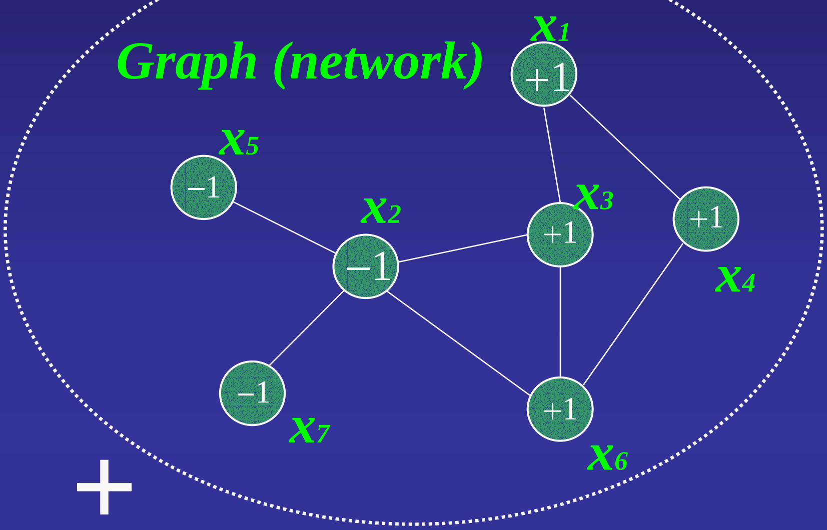
Ex) Heterogeneous Representation
of Multiple Data Sources

Vectorial Data

	A_1	A_2	…	A_{10}	y
x_1	10	5	…	1000	1
x_2	6	6	…	3500	-1
x_3	7	7	…	400	1
…	…	…	…	…	…
x_7	3	88	…	700	-1

+

Graph (network)



+

Sequence (string)

x_1	agctgtttagctatatgcgtatagggct	1
x_2	cagtgtcgaatagccgctcgaaaaaa	-1
…	…	…
x_7	catgctgtatgcccgatagcgtgatcg	-1

Content

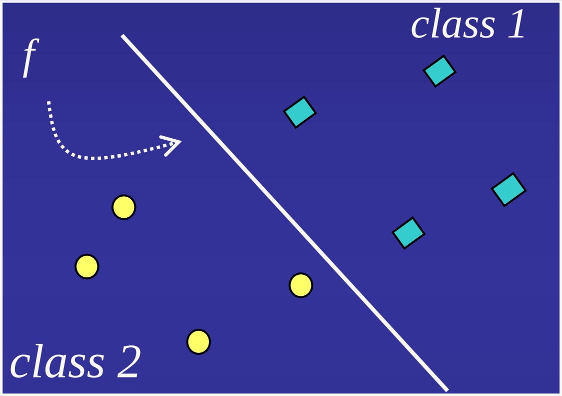
1. *Basics*
2. *Tasks*
3. *Learning*
4. *Models with Examples*

Learning

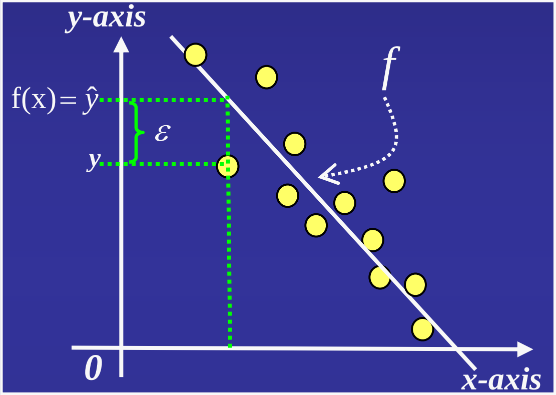
Learning

Building a model *f* given dataset {X,Y} is called “Learning” or “Training”

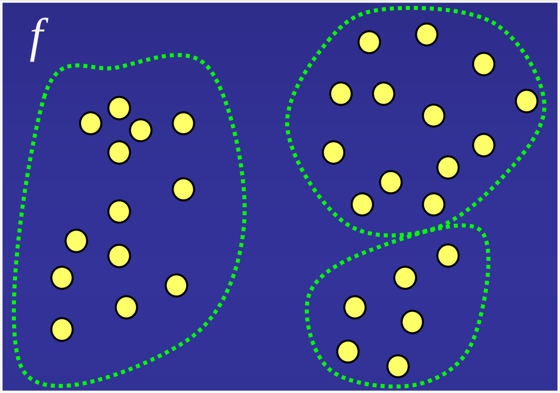
Classification



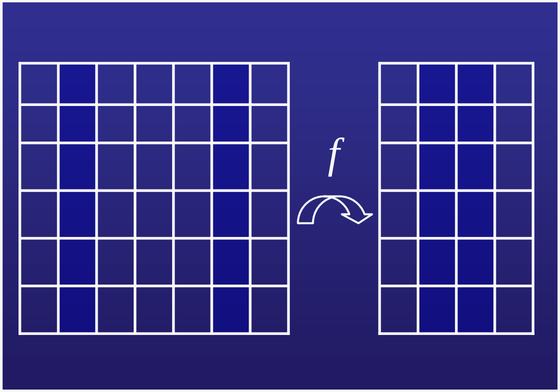
Regression



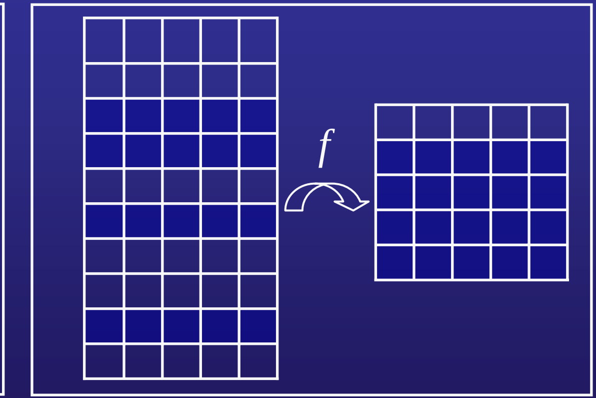
Clustering



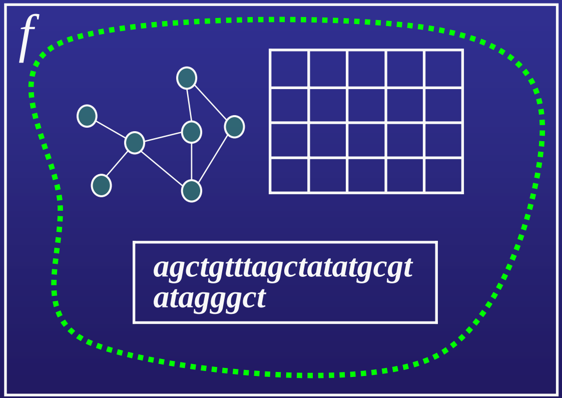
Dimensionality Reduction



Data Reduction



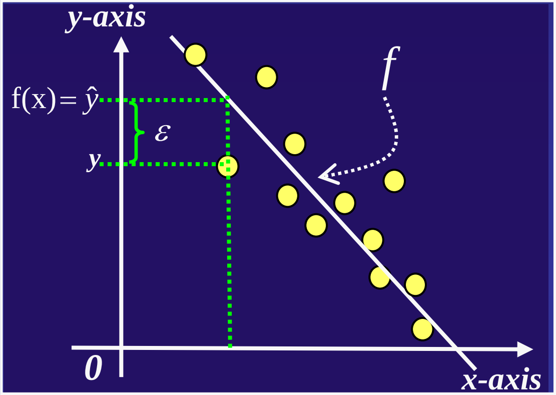
Data Integration



Learning

Building a model *f* given dataset {X,Y} is called "Learning" or "Training"

Regression



Ex) Regression Model

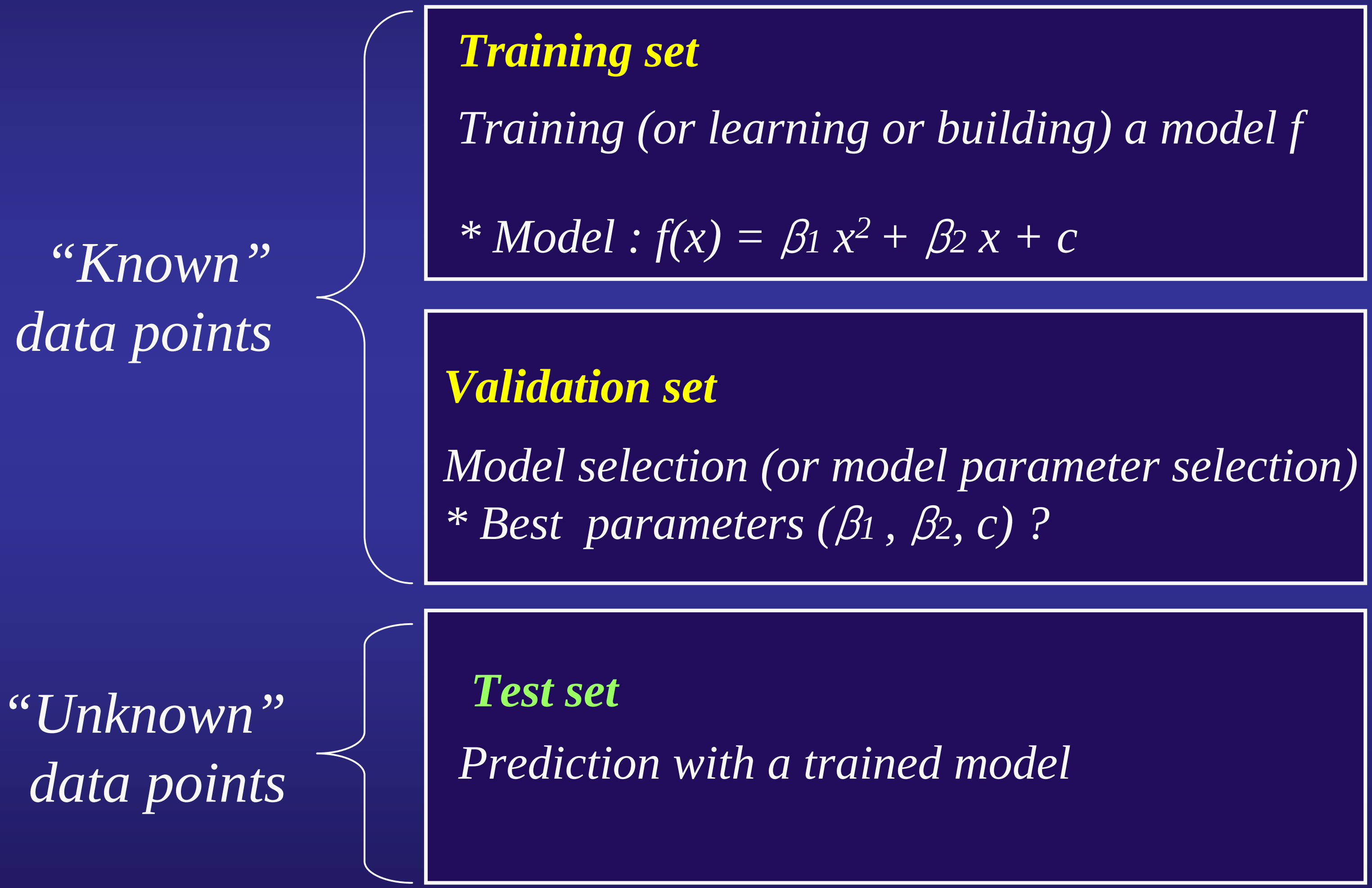
$$f(x) = \beta_1 x^2 + \beta_2 x + c$$

In other words, given data {X, Y}, finding the values of parameters, β_1, β_2, and *c* is "Learning"

Data Set Split

		A_1	A_2	A_3	…	A_{10}	y
“Known” data points	x_1	10	5	red	…	1000	1
	x_2	6	6	blue	…	3500	20
	x_3	7	7	yellow	…	400	45
	…	…	…	…	…	…	…
	x_{18}	3	56	red	…	0	30
	x_{19}	15	62	red	…	500	100
	x_{20}	3	88	blue	…	700	3
“Unknown” data points	x_{21}	5	42	red	…	560	?
	…	…	…	…	…	…	?
	x_{50}	25	56	blue	…	600	?

Data Set Split



“Known” data points

Training set

Training (or learning or building) a model *f*

* Model : $f(x) = \beta_1 x^2 + \beta_2 x + c$

Validation set

Model selection (or model parameter selection)

* Best parameters (β_1 , β_2, c) ?

“Unknown” data points

Test set

Prediction with a trained model

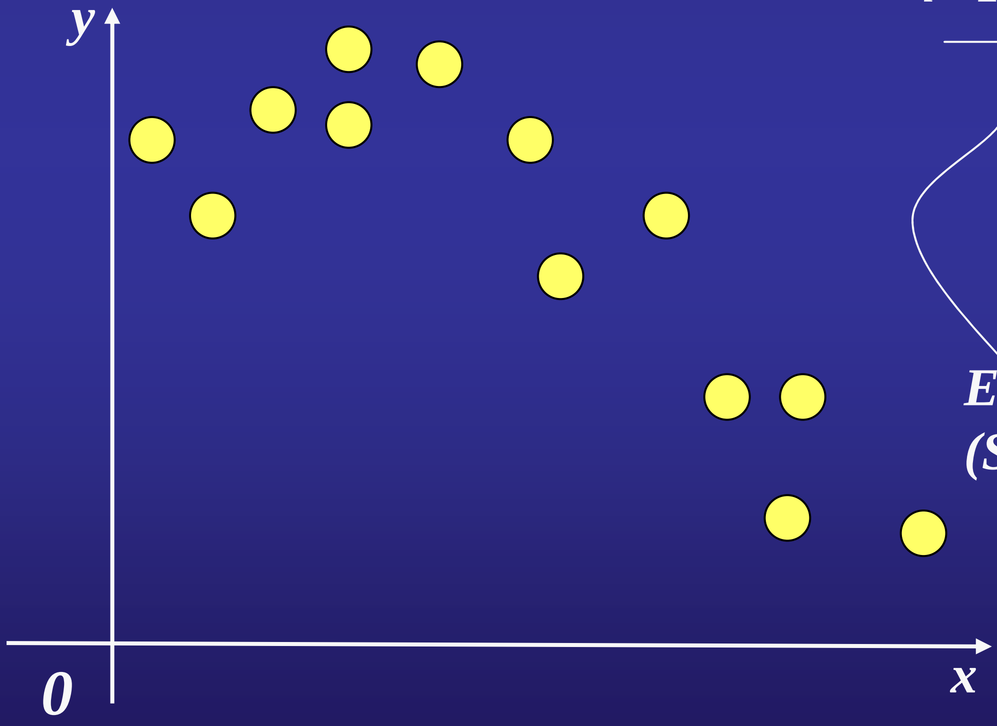
Data Set Split & Learning

Training set

Build a model "*f*" minimizing the errors

$$\underbrace{\sum_{i=1}^{n} \varepsilon^{2}}_{\textit{Error}} = \sum_{i=1}^{n} (\underbrace{f(x_i)}_{\textit{Prediction}} - \underbrace{y_i}_{\textit{Target}})^{2}$$

(SSE: sum of squared error)

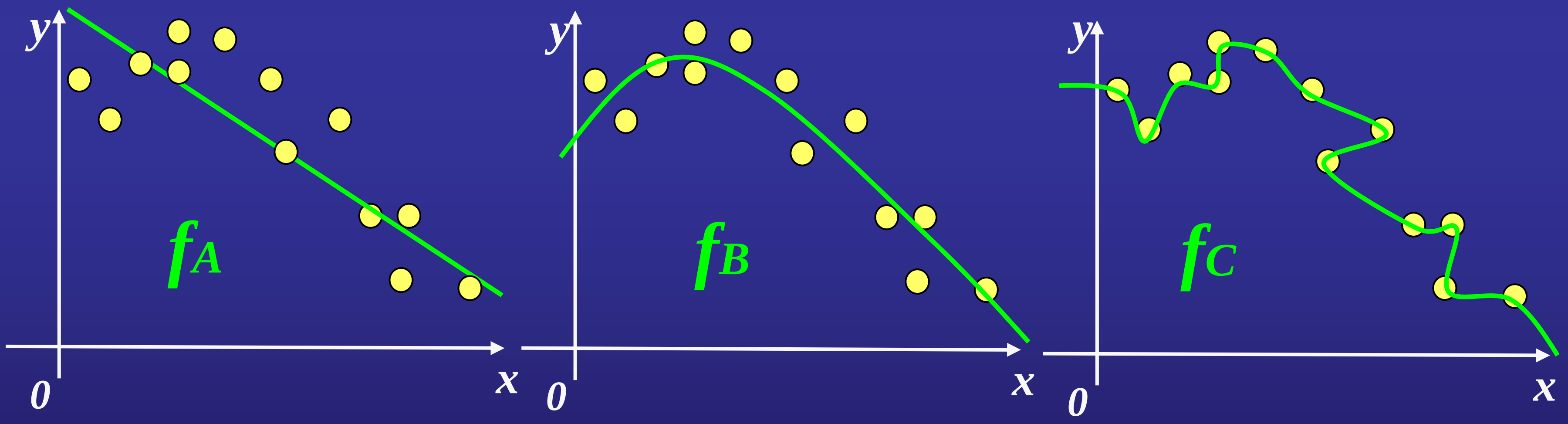


Data Set Split & Learning

Training set

Build a model "*f*" minimizing the errors

$$\text{min.} \quad \sum_{i=1}^{n} \varepsilon^2 = \sum_{i=1}^{n} (f(x_i) - y_i)^2$$

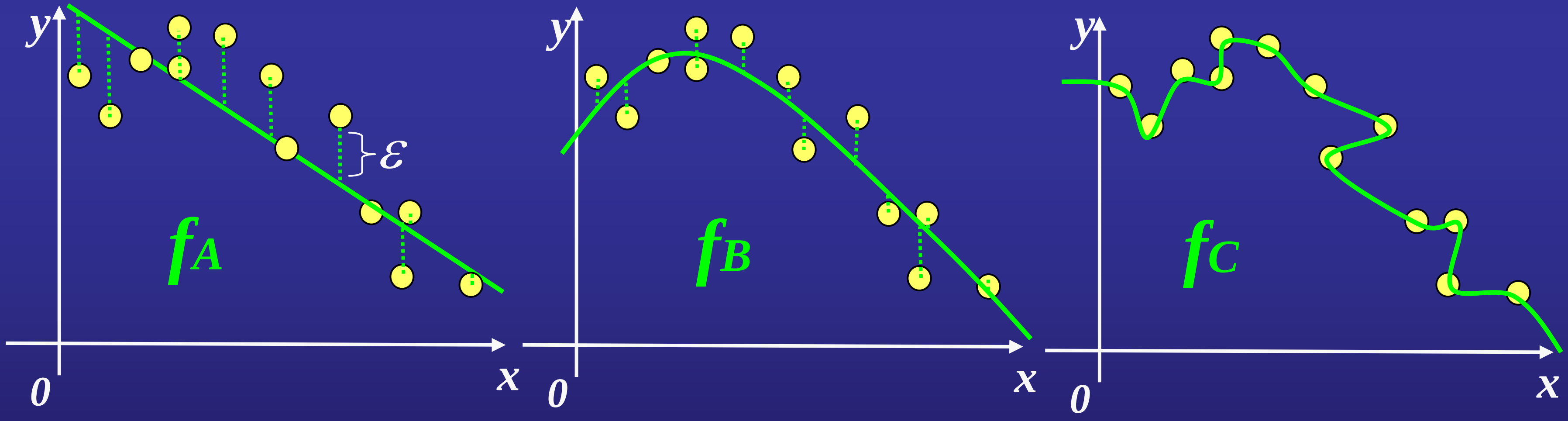


Data Set Split & Learning

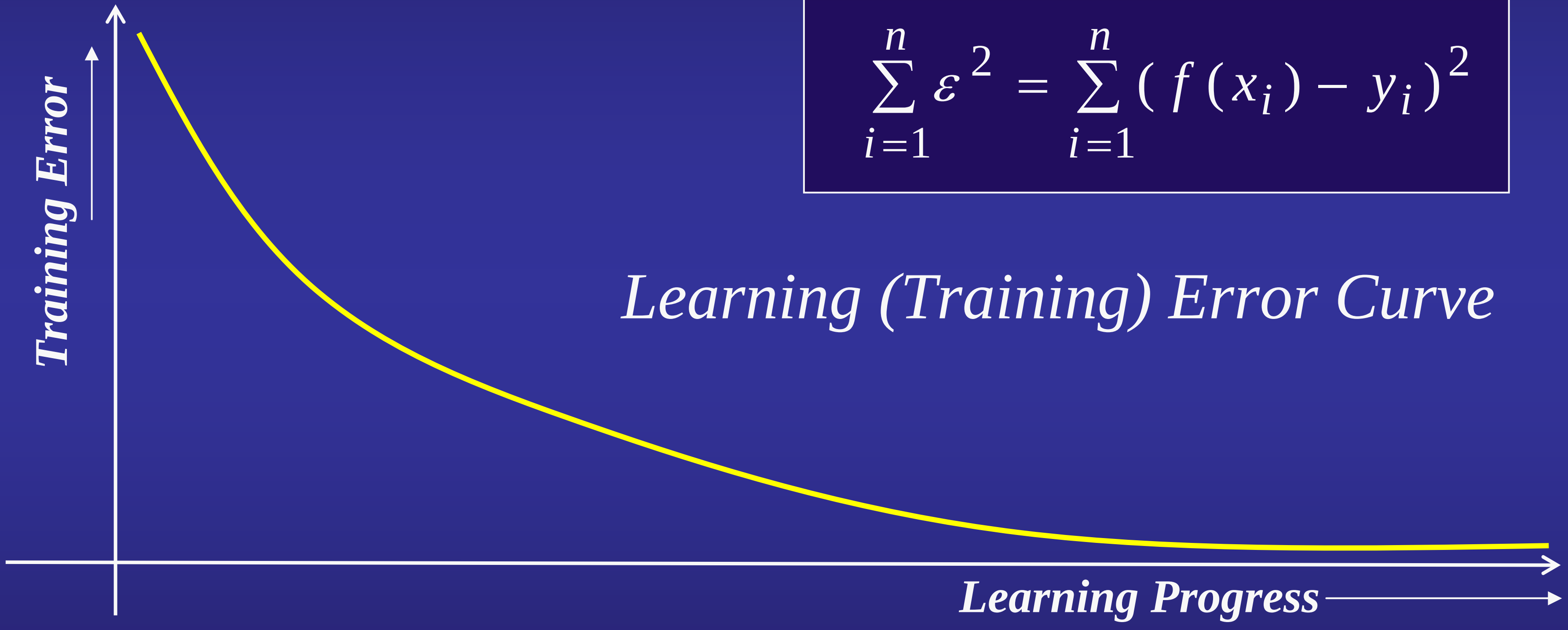
Training set

Build a model "*f*" minimizing the errors

$$\text{min.}\quad \sum_{i=1}^{n} \varepsilon^2 = \sum_{i=1}^{n} (f(x_i) - y_i)^2$$



Data Set Split & Learning

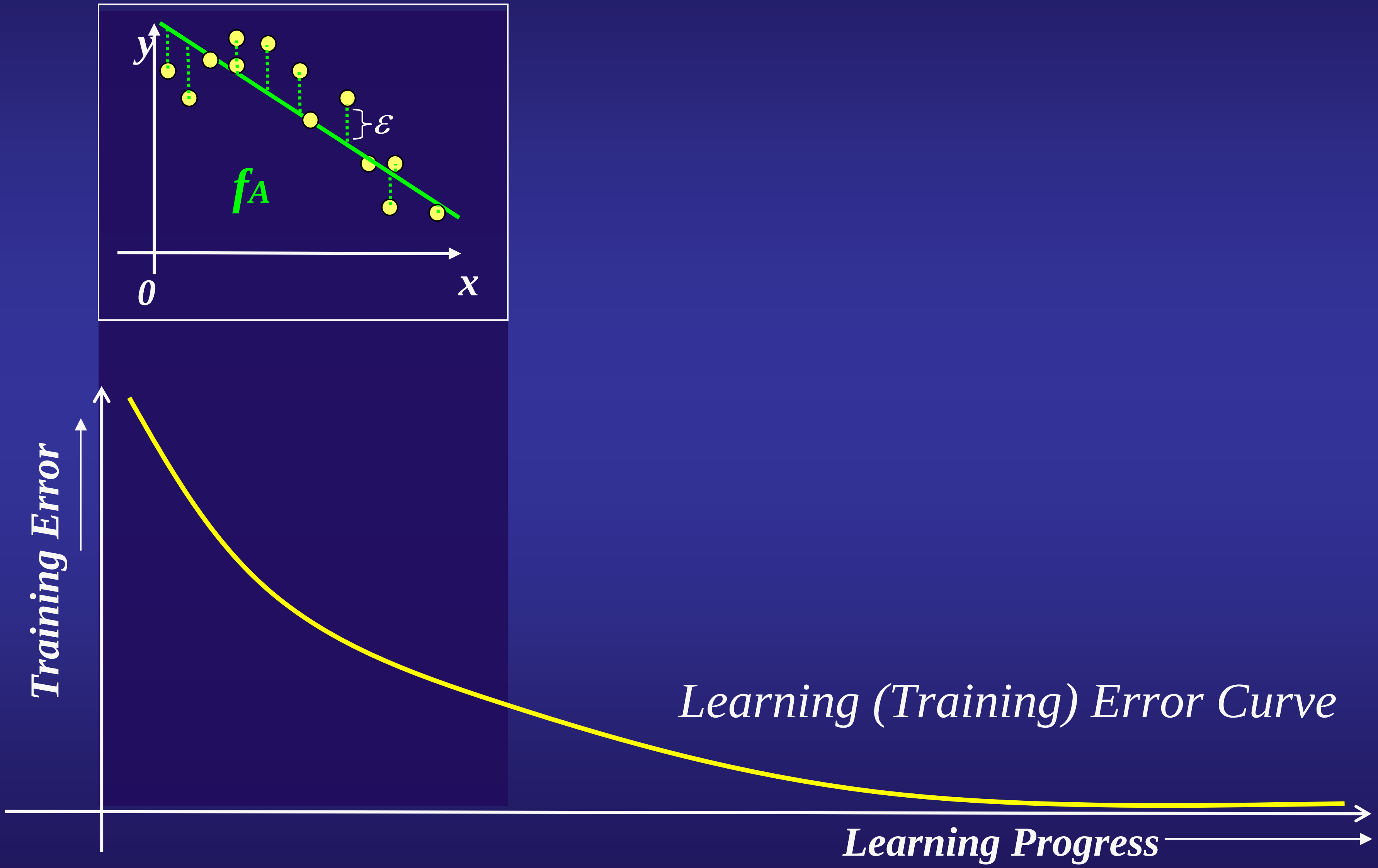


Training Error

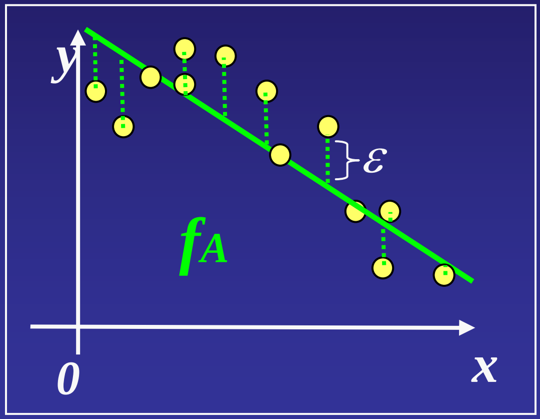
$$\sum_{i=1}^{n} \varepsilon^{2} = \sum_{i=1}^{n} \left(f(x_i) - y_i \right)^{2}$$

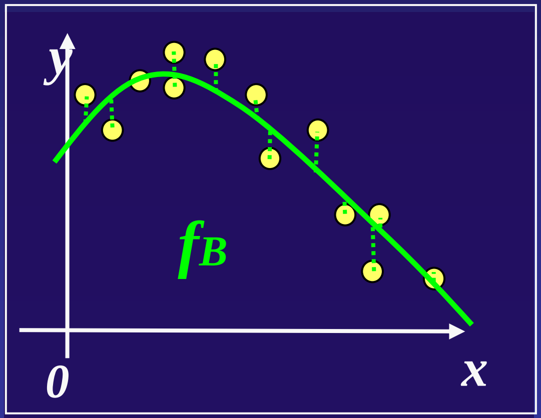
Learning (Training) Error Curve

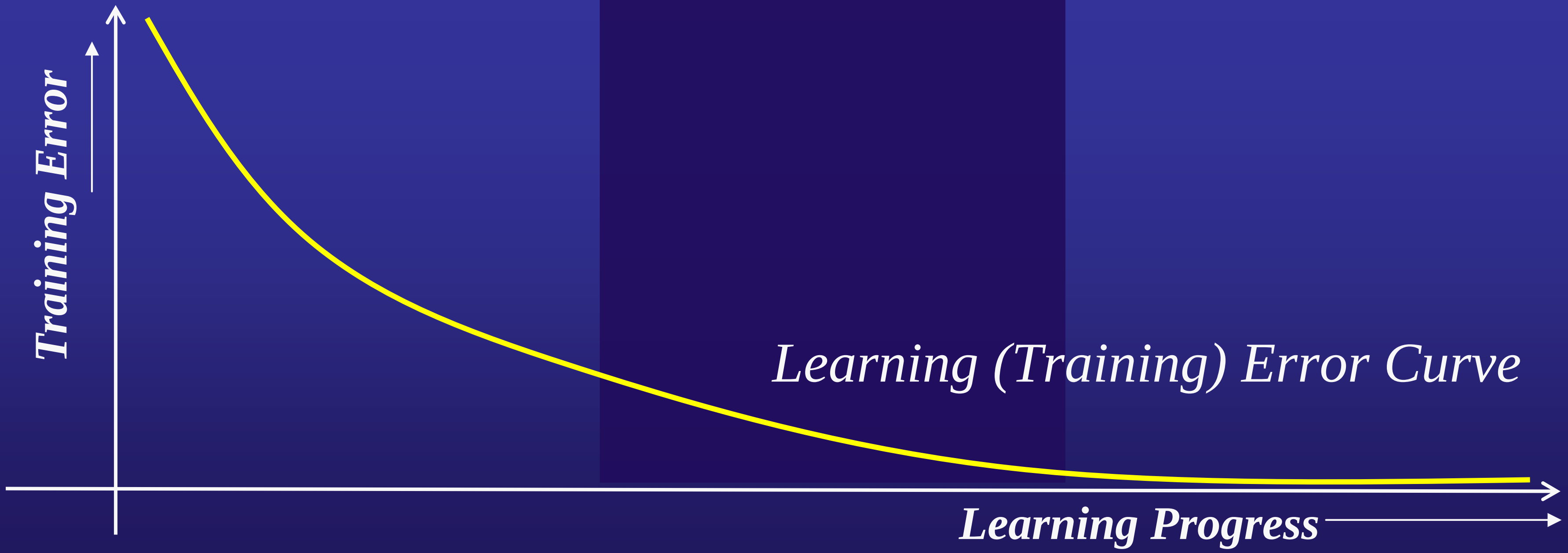
Data Set Split & Learning



Data Set Split & Learning

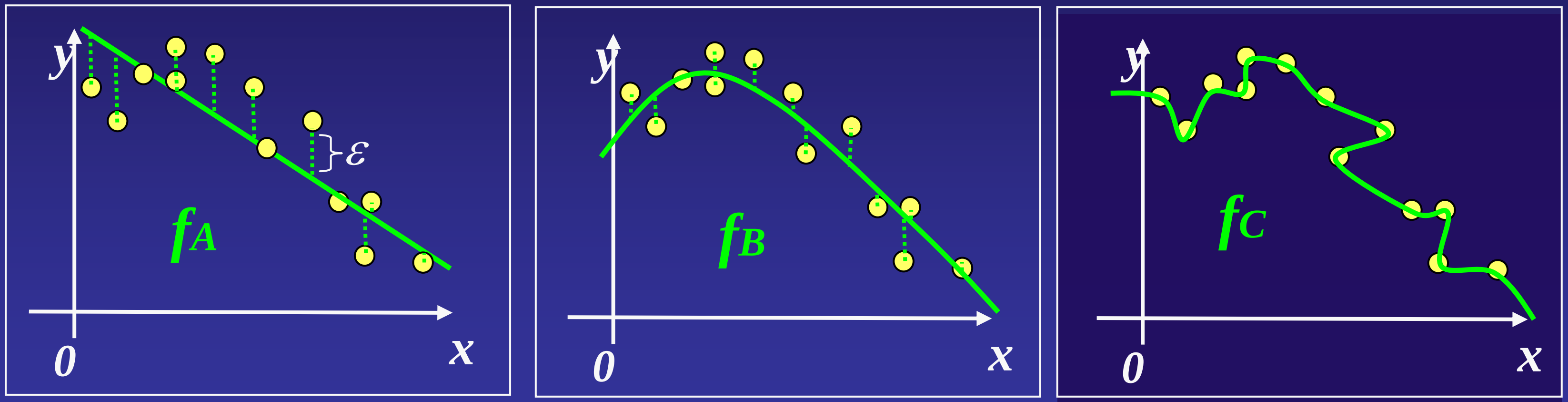


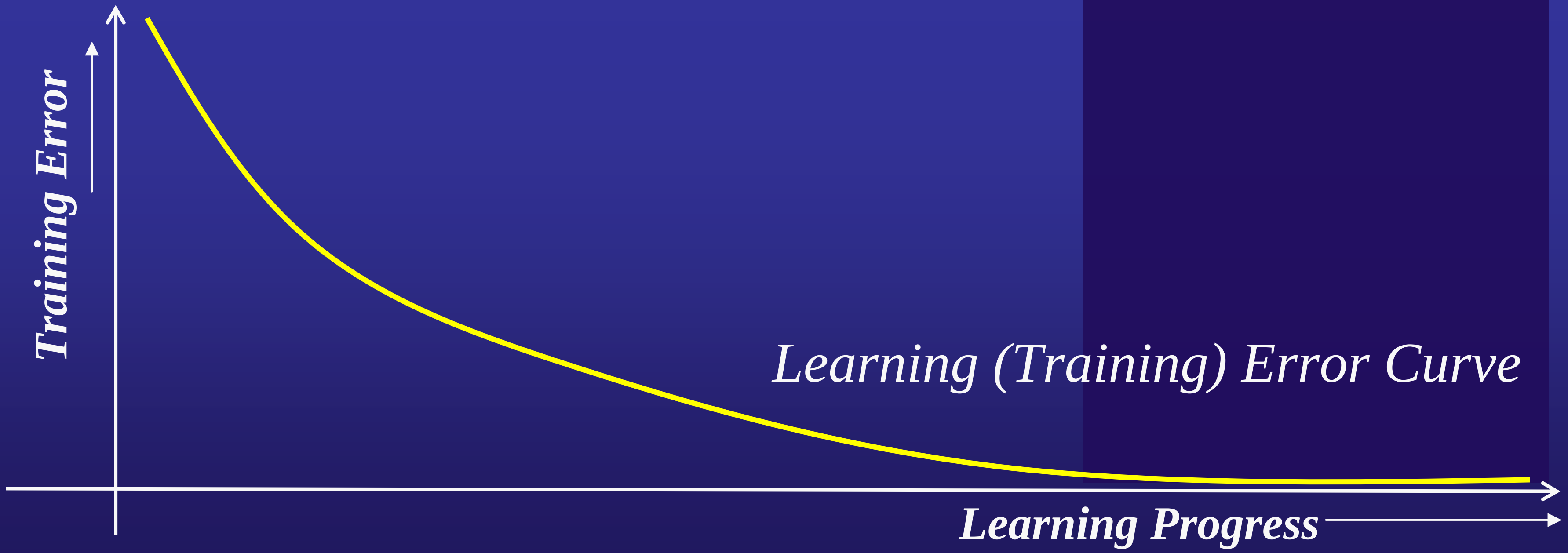




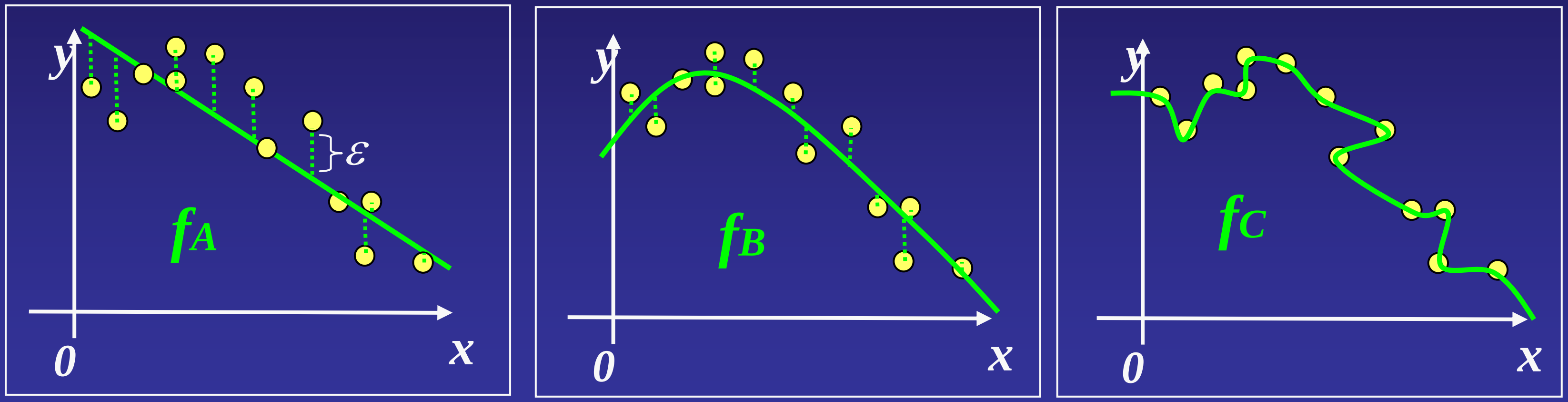
Learning (Training) Error Curve

Data Set Split & Learning

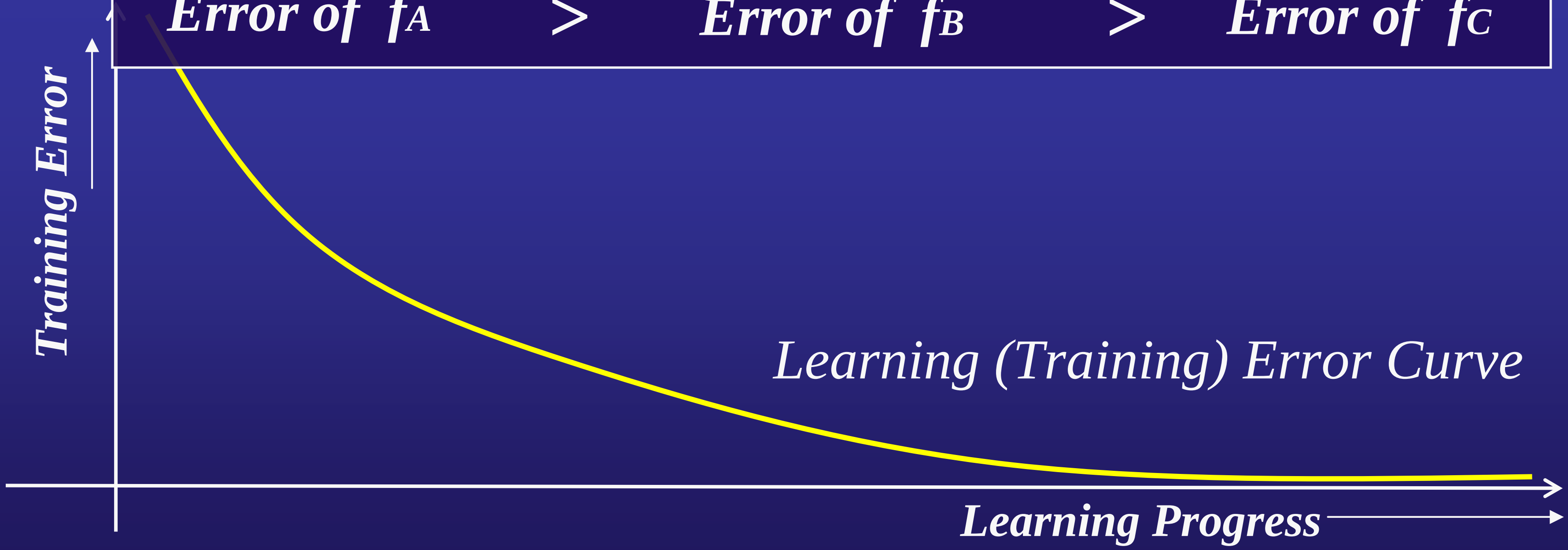




Data Set Split & Learning



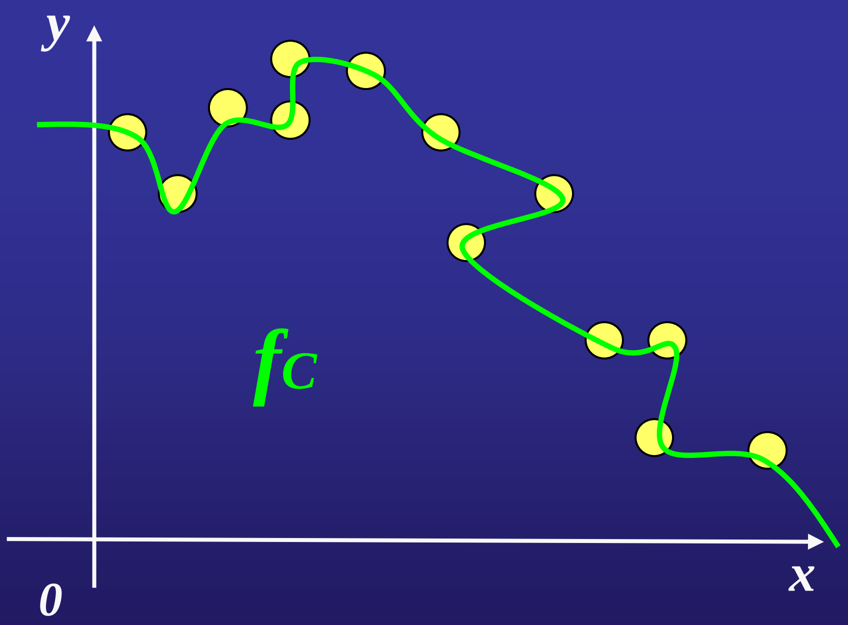
Error of f_A > Error of f_B > Error of f_C



Data Set Split & Learning

Training set

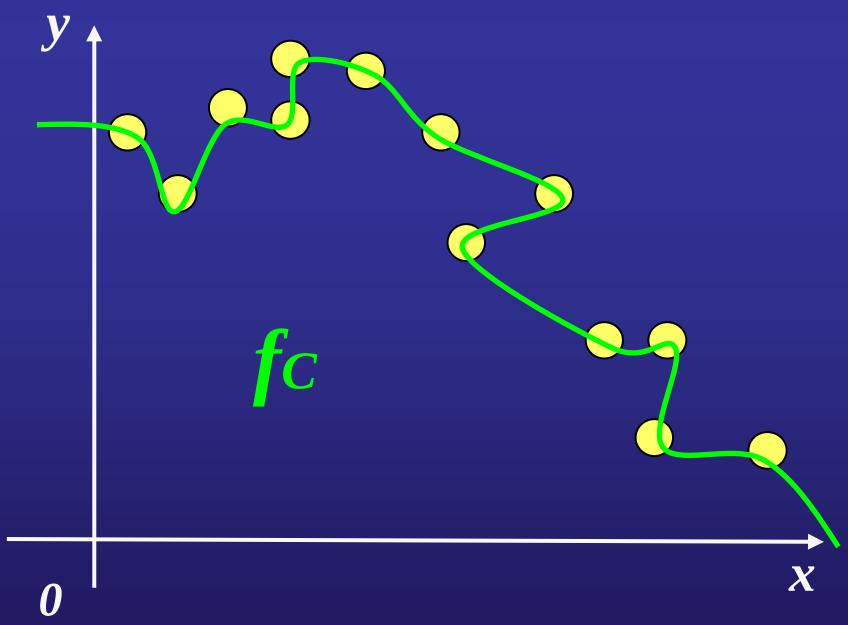
fc is the best model ?



Data Set Split & Learning

Training set

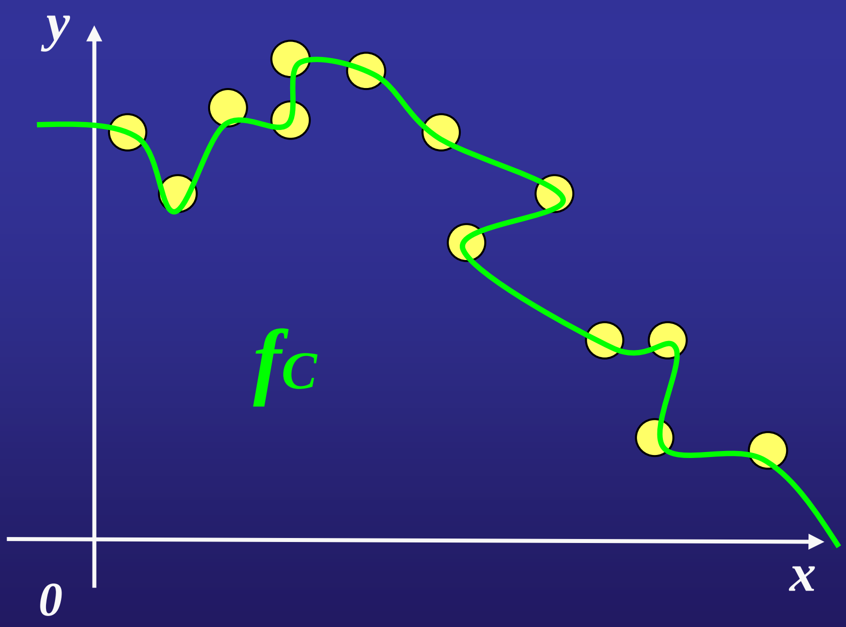
fc is the best model ? ***No***



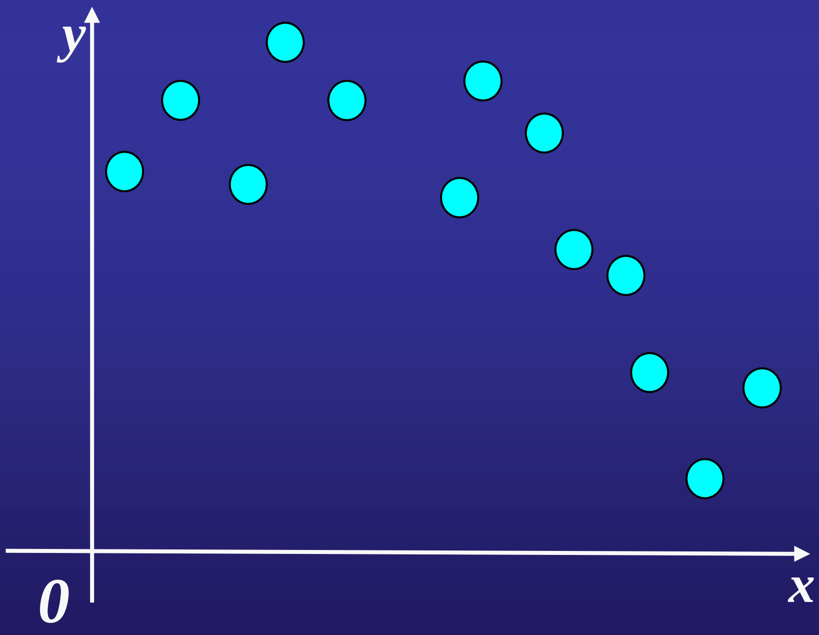
Why not?

Data Set Split & Learning

Training set



Test set

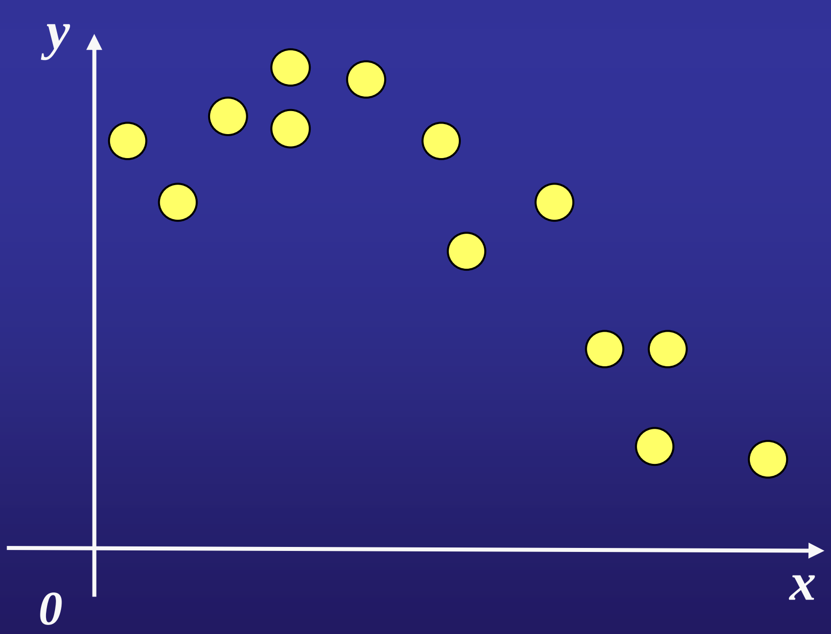


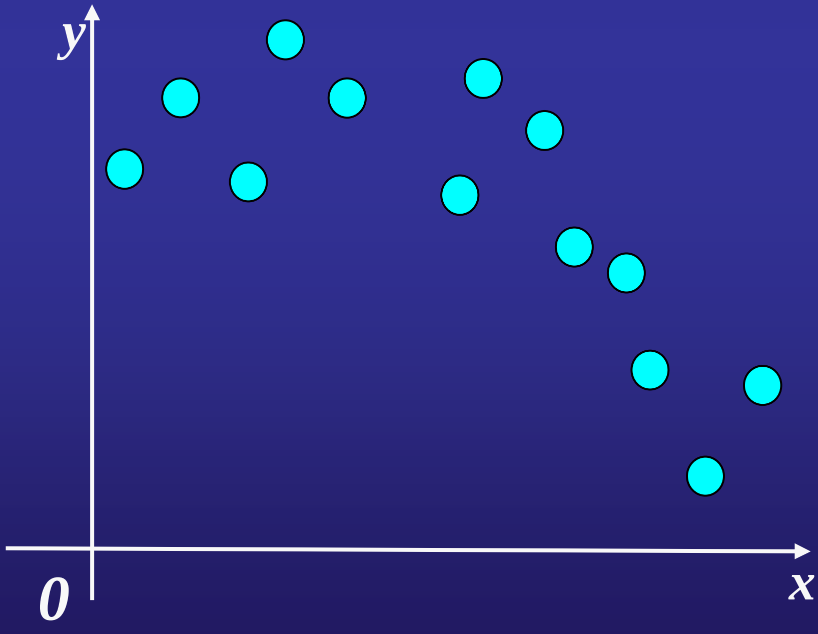
Data Set Split & Learning

Training set

Test set

The data points in Test set are assumed to be drawn the same distribution as those in Training set



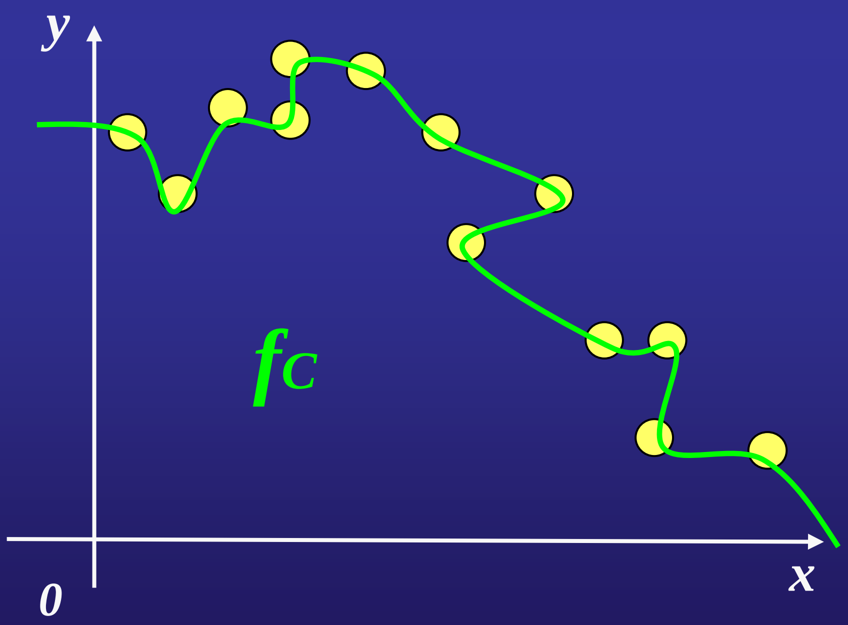


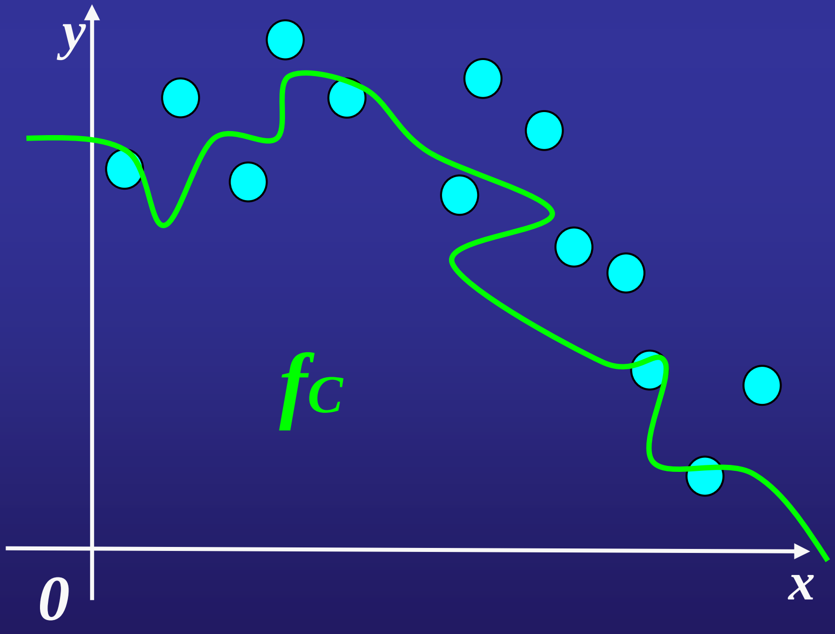
Data Set Split & Learning

Training set

Test set

However,
when the fully trained model *fc* is applied to the
test data points, it does <u>not</u> fit them well any more



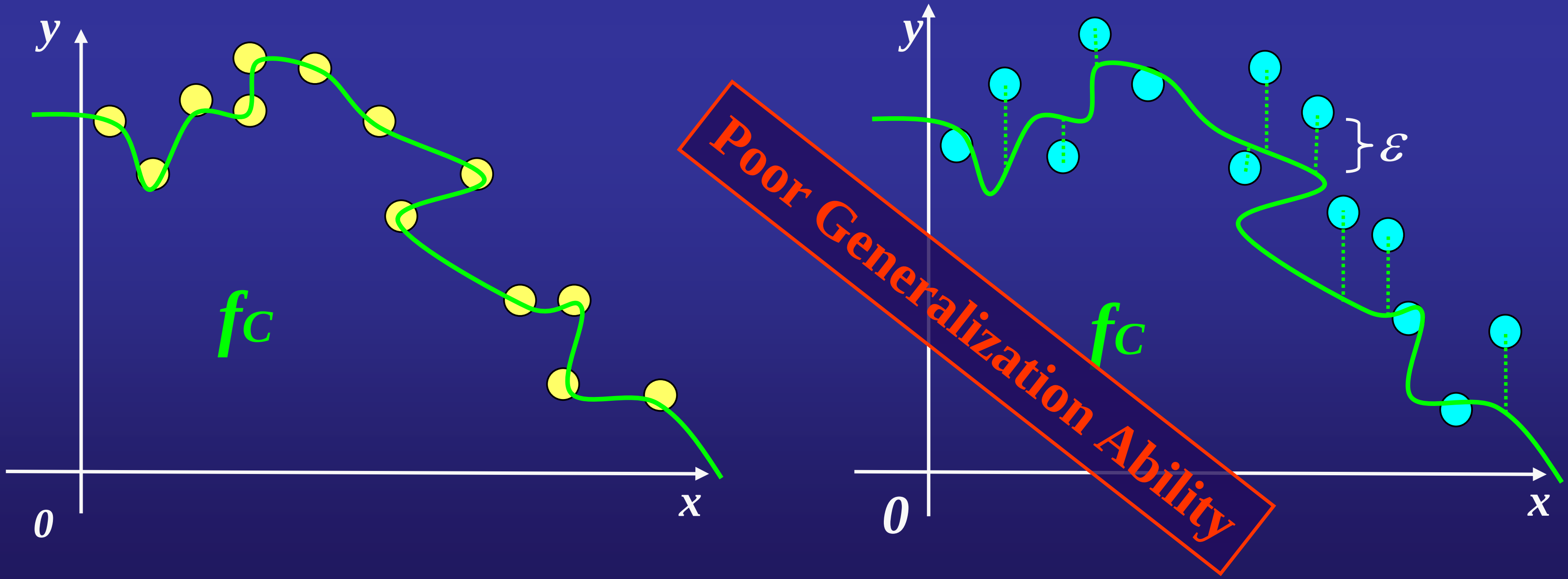


Data Set Split & Learning

Training set

Test set

However,
when the fully trained model f_C is applied to the
test data points, it does not fit them well any more



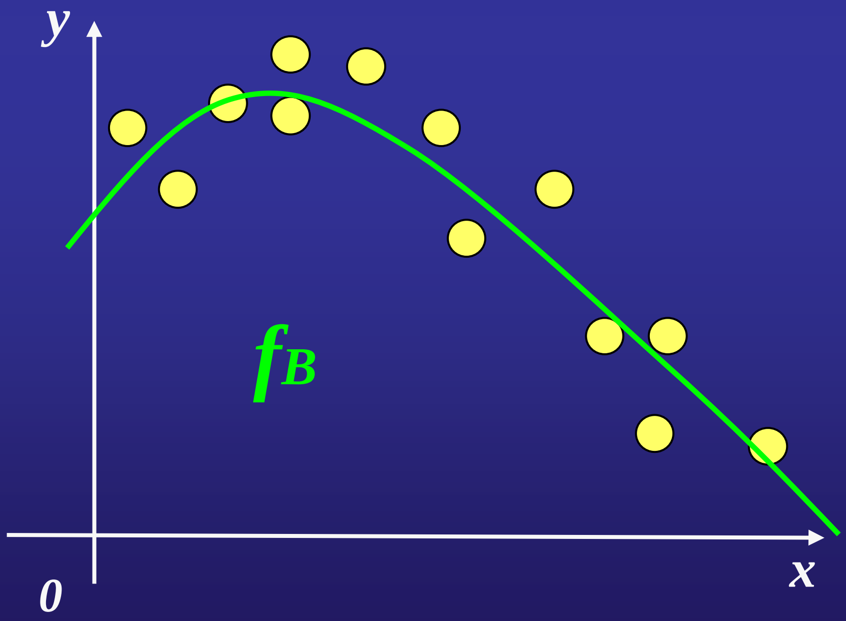
Poor Generalization Ability

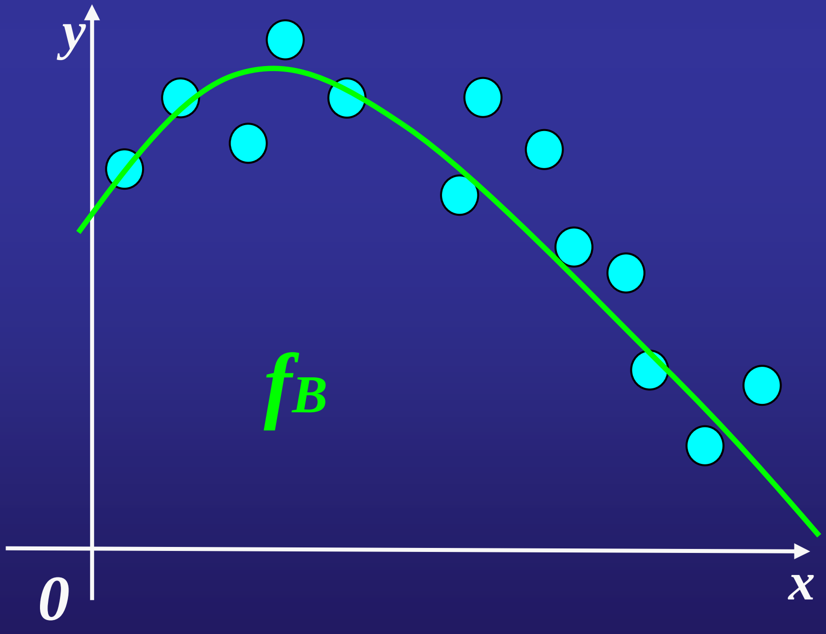
Data Set Split & Learning

Training set

Test set

On the contrary,
a “properly” trained model f_B has <u>more generalization ability</u>





Data Set Split & Learning

Training set

Test set

(note that Test set is unknown during Training)

Then,
how can we find a “proper” model
with absence of Test set ?

Data Set Split & Learning

Training set ***Validation set***

Then,
how can we find a "proper" model
with absence of Test set ?

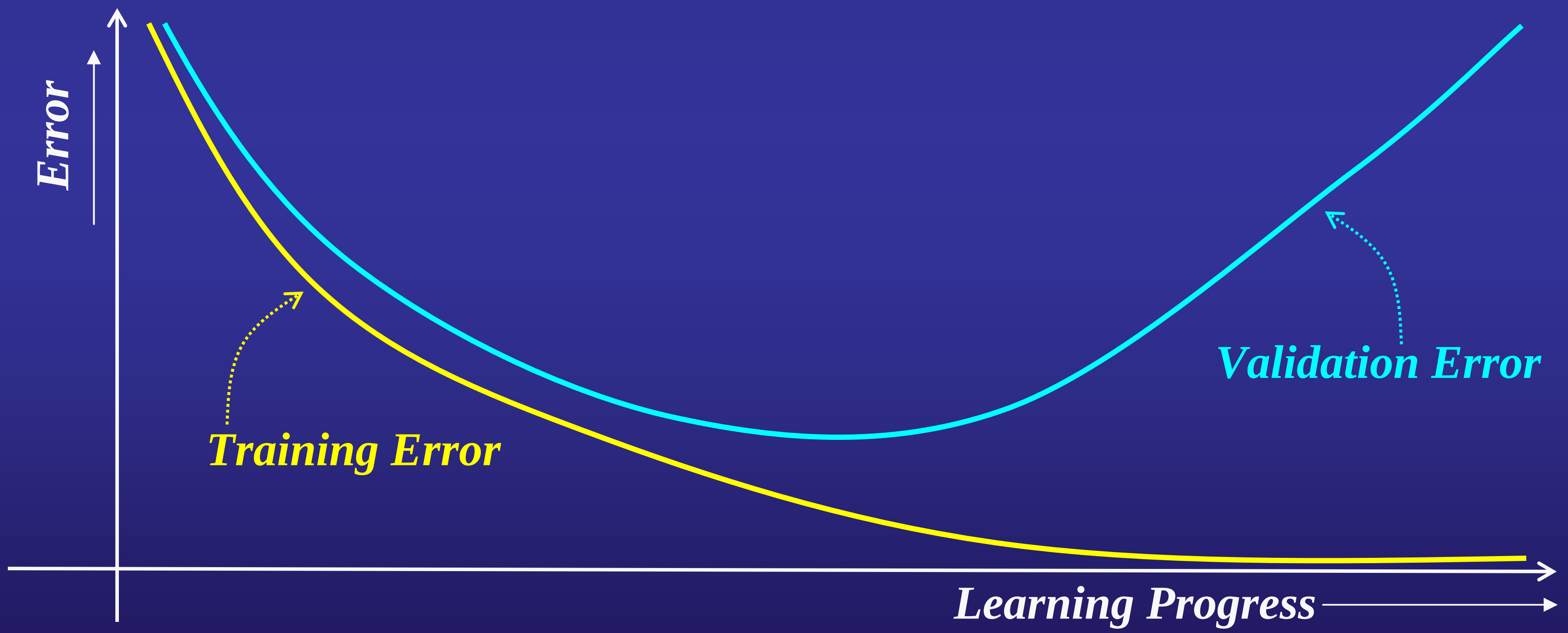
Use Validation set (say, a Pseudo Test Set) !

: Temporarily assume that the data set is "Unknown"

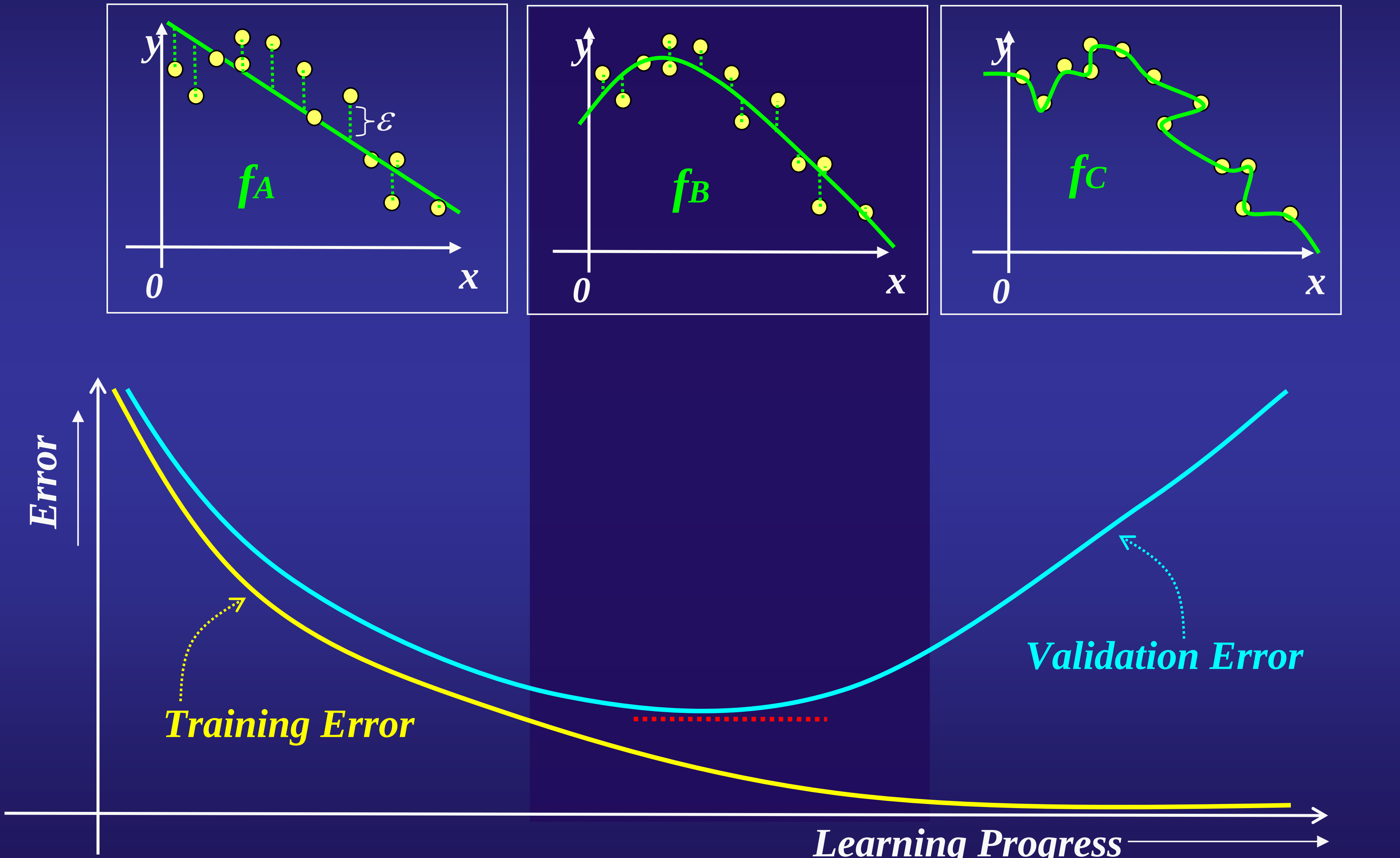
Data Set Split & Learning

Training set

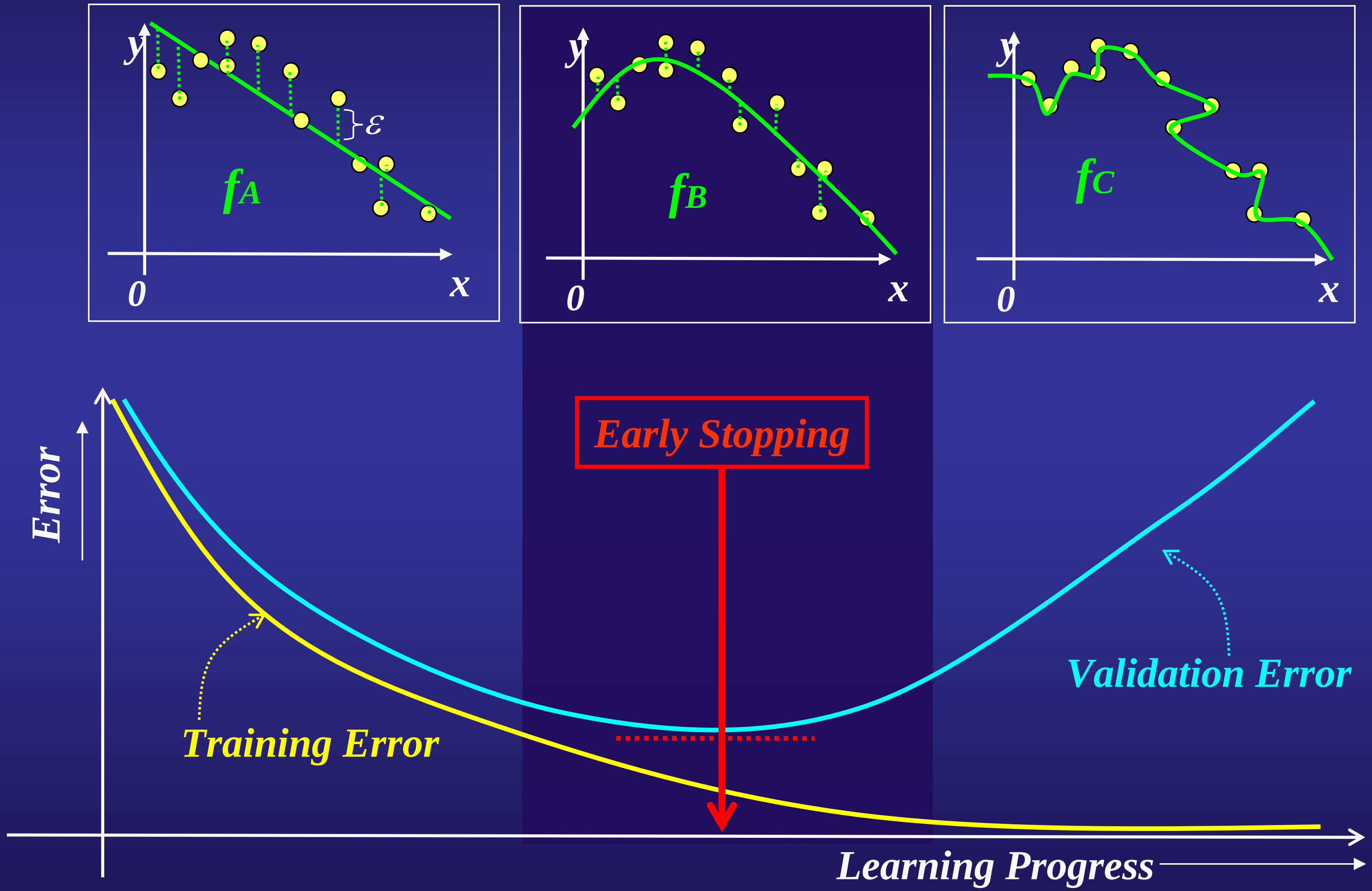
Validation set



Data Set Split & Learning

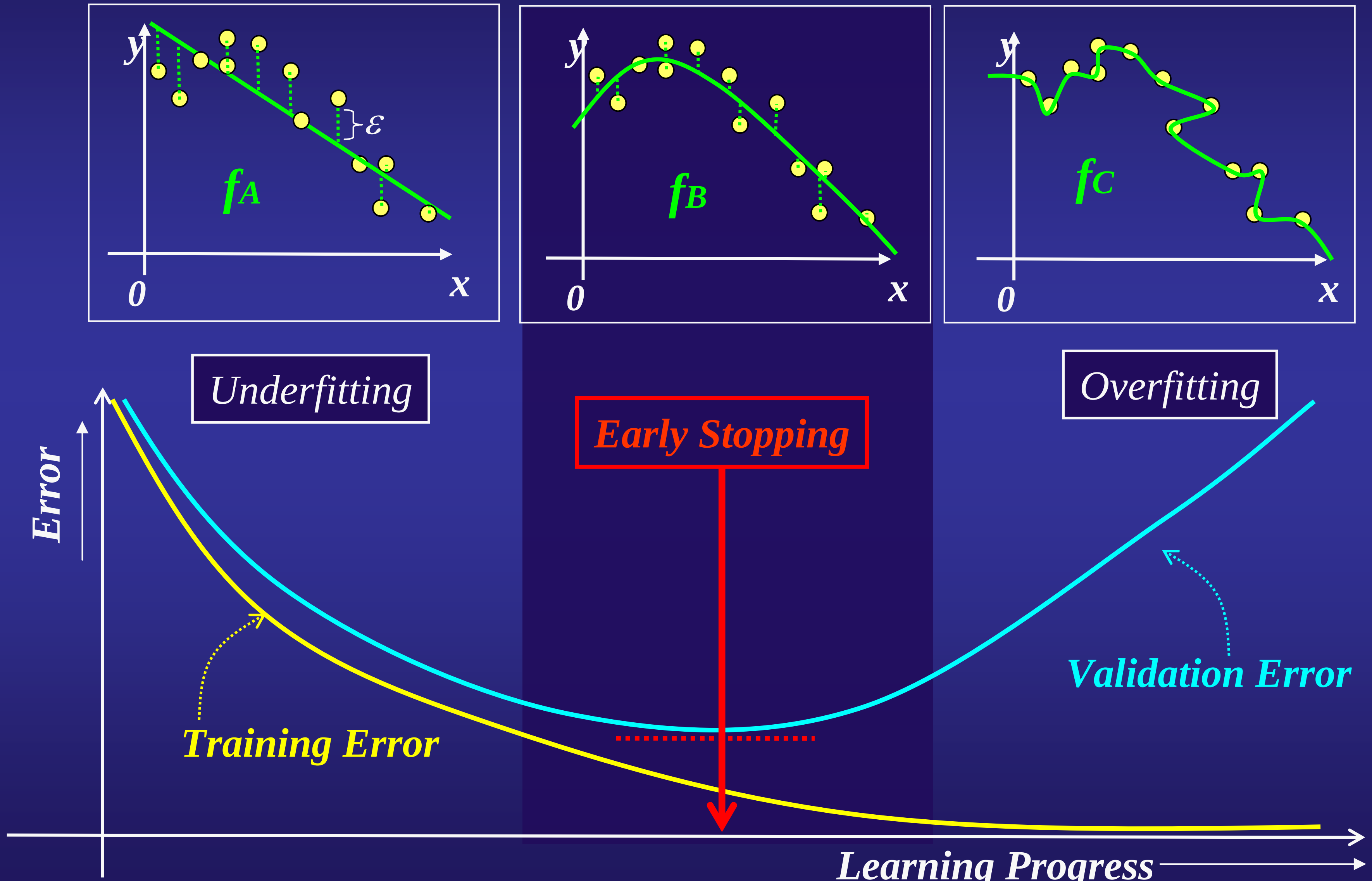


Data Set Split & Learning



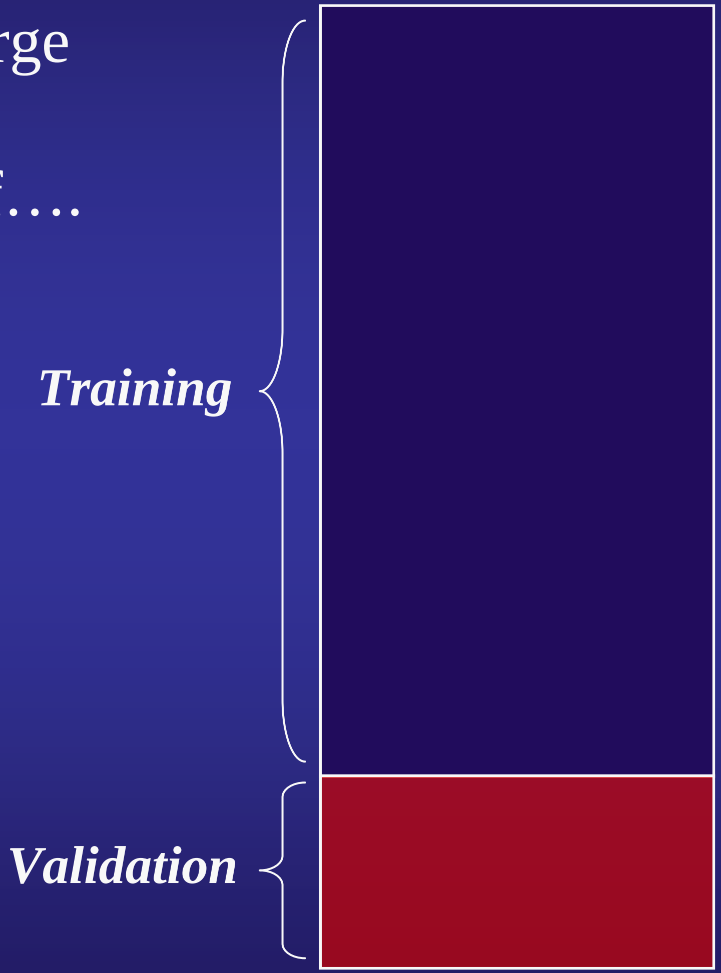
HyunJung (Helen) Shin, Max Planck Society, European School of Genetic Medicine, 03. 2006

Data Set Split & Learning

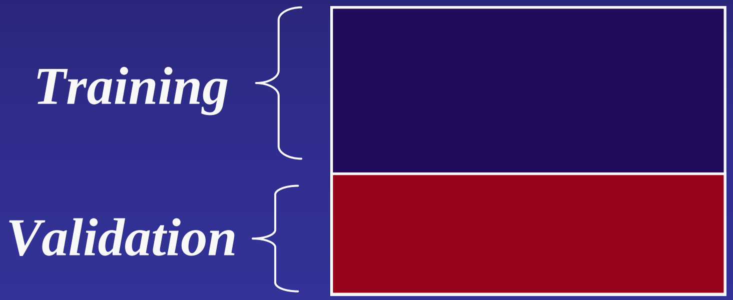


Data Set Split & Learning

If the known data points are large enough for training after separating the validation set off….



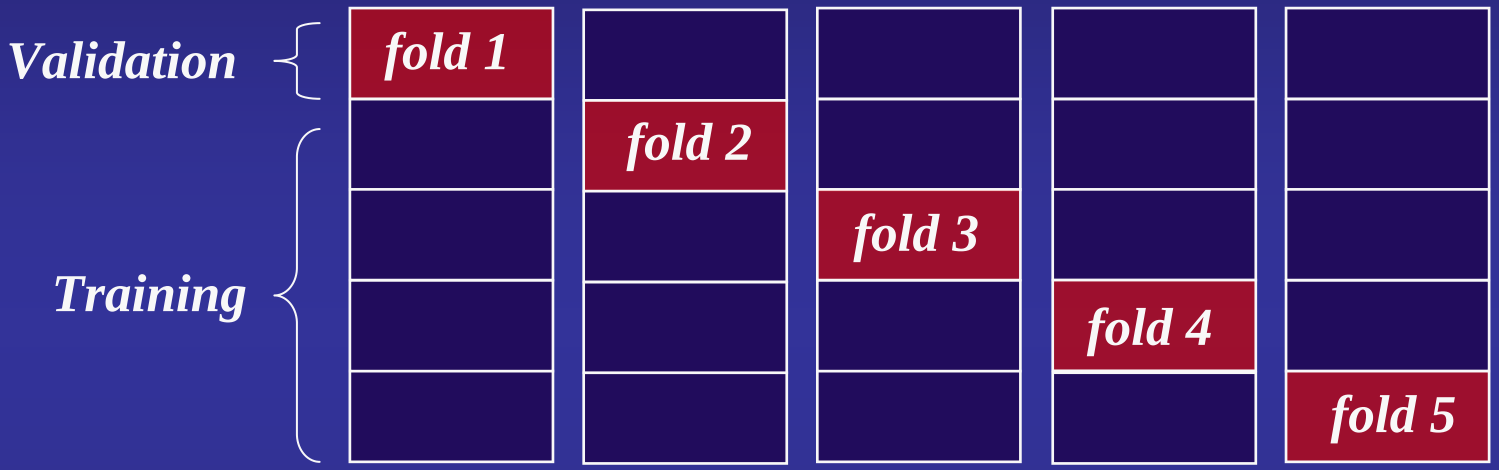
Data Set Split & Learning



If the known data points are insufficient for training after taking the validation set out ?

Data Set Split & Learning

(Ex) 5 Cross-Validation (5CV)



$$\text{Cross-Validation Error} = \frac{\text{Validation Error (fold 1)} + \text{Validation Error (fold 2)} + \cdots + \text{Validation Error (fold 5)}}{5}$$

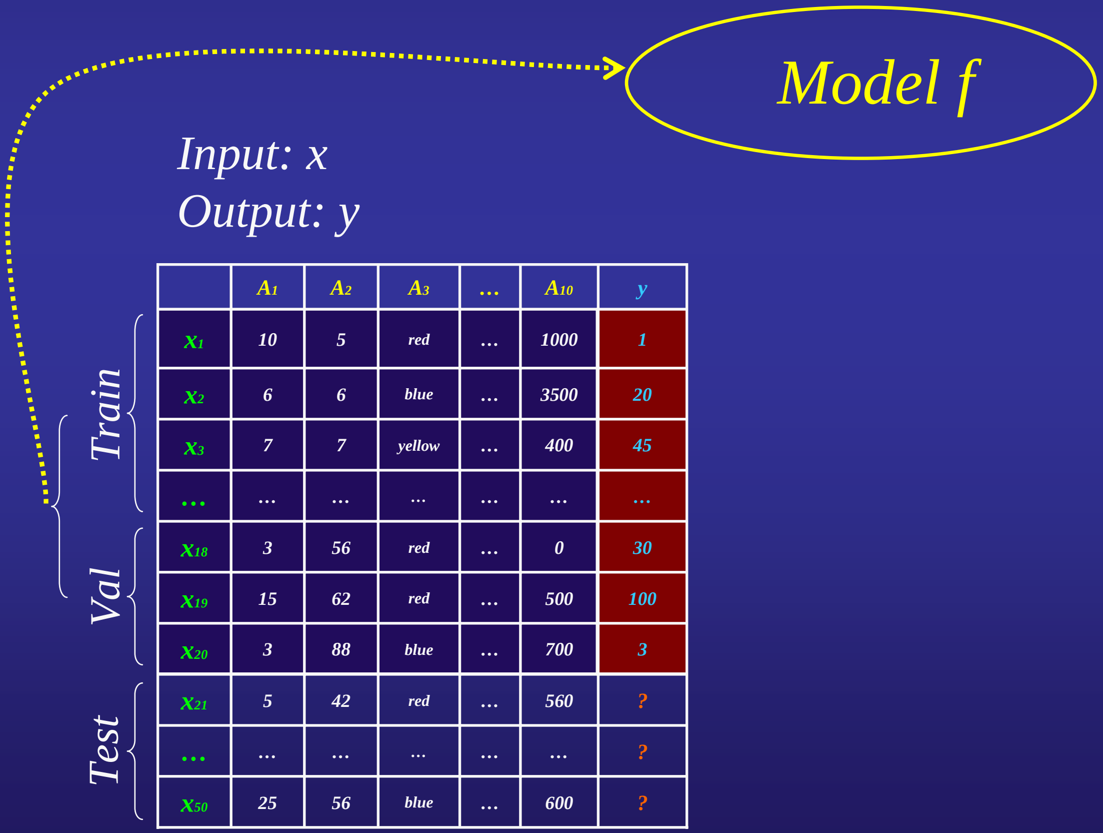
Learning Schemes

- *Supervised*
- *Unsupervised*
- *Semi-Supervised*

Supervised Learning

Learning or Training

$f(x) \approx y$: *Supervised*



Model f

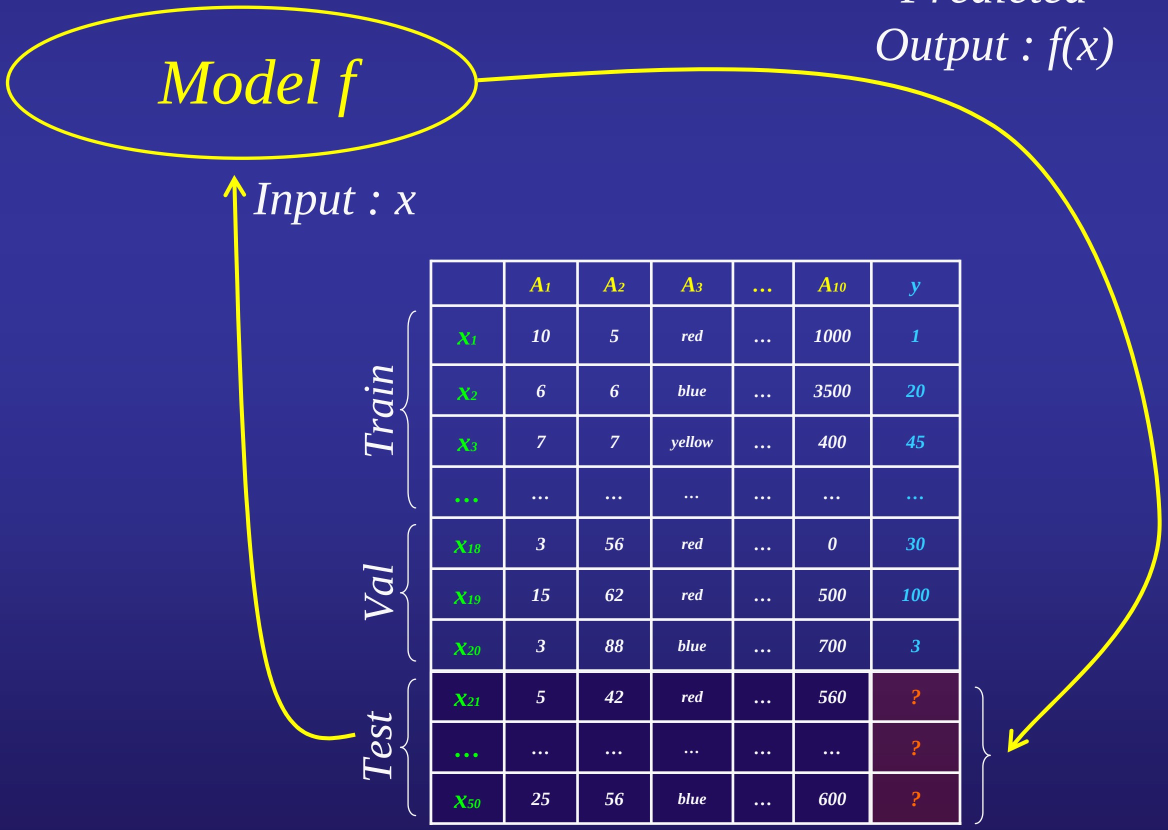
Input: x

Output: y

		A_1	A_2	A_3	…	A_{10}	y
Train	x_1	10	5	red	…	1000	1
	x_2	6	6	blue	…	3500	20
	x_3	7	7	yellow	…	400	45
	…	…	…	…	…	…	…
Val	x_{18}	3	56	red	…	0	30
	x_{19}	15	62	red	…	500	100
	x_{20}	3	88	blue	…	700	3
Test	x_{21}	5	42	red	…	560	?
	…	…	…	…	…	…	?
	x_{50}	25	56	blue	…	600	?

Supervised Learning

Prediction or Test

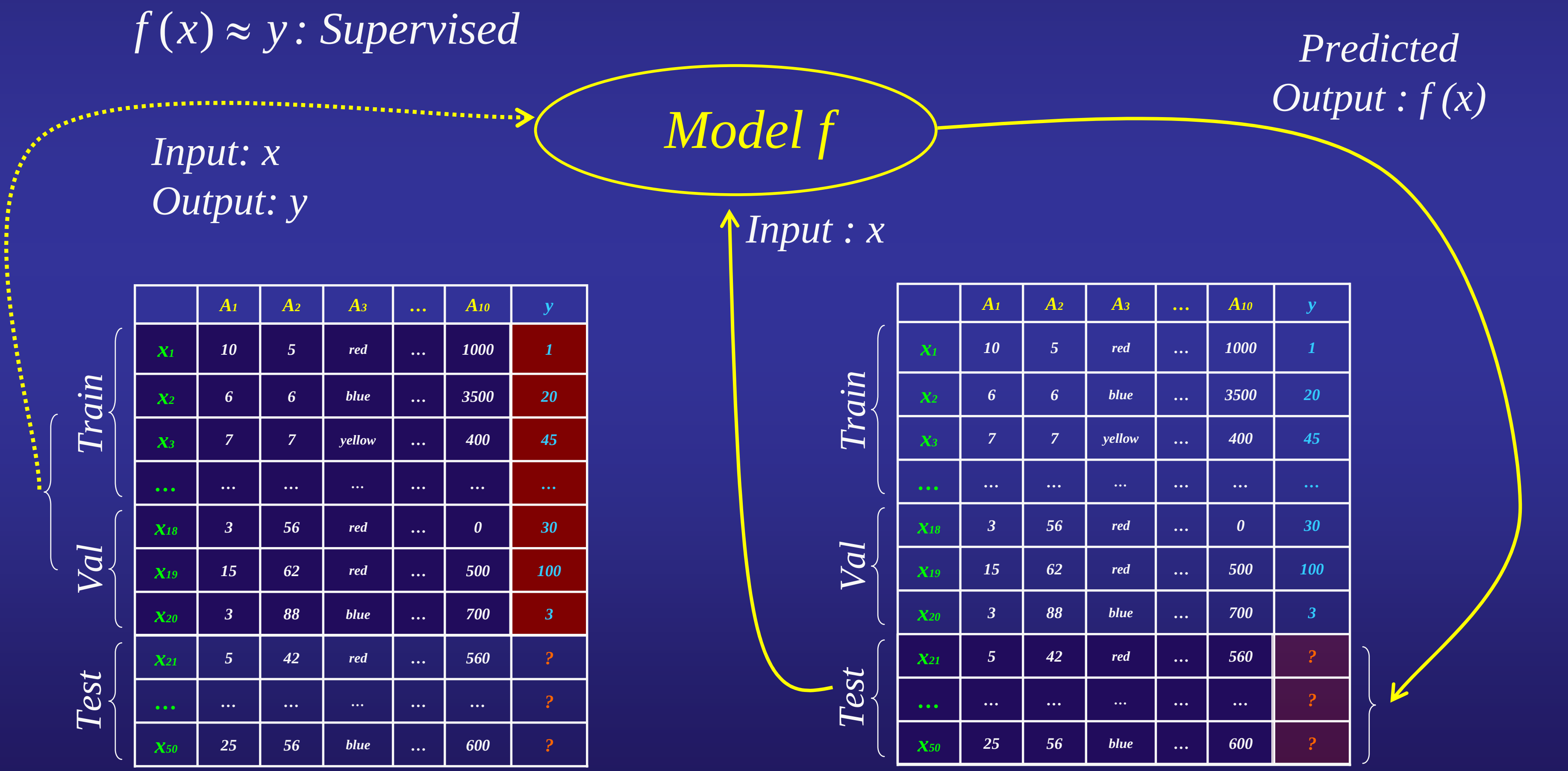


		A_1	A_2	A_3	…	A_{10}	y
Train	x_1	10	5	red	…	1000	1
	x_2	6	6	blue	…	3500	20
	x_3	7	7	yellow	…	400	45
	…	…	…	…	…	…	…
Val	x_{18}	3	56	red	…	0	30
	x_{19}	15	62	red	…	500	100
	x_{20}	3	88	blue	…	700	3
Test	x_{21}	5	42	red	…	560	?
	…	…	…	…	…	…	?
	x_{50}	25	56	blue	…	600	?

Supervised Learning

Learning or Training

Prediction or Test



$f(x) \approx y$: Supervised

Model f

Input: x

Output: y

Input : x

Predicted Output : $f(x)$

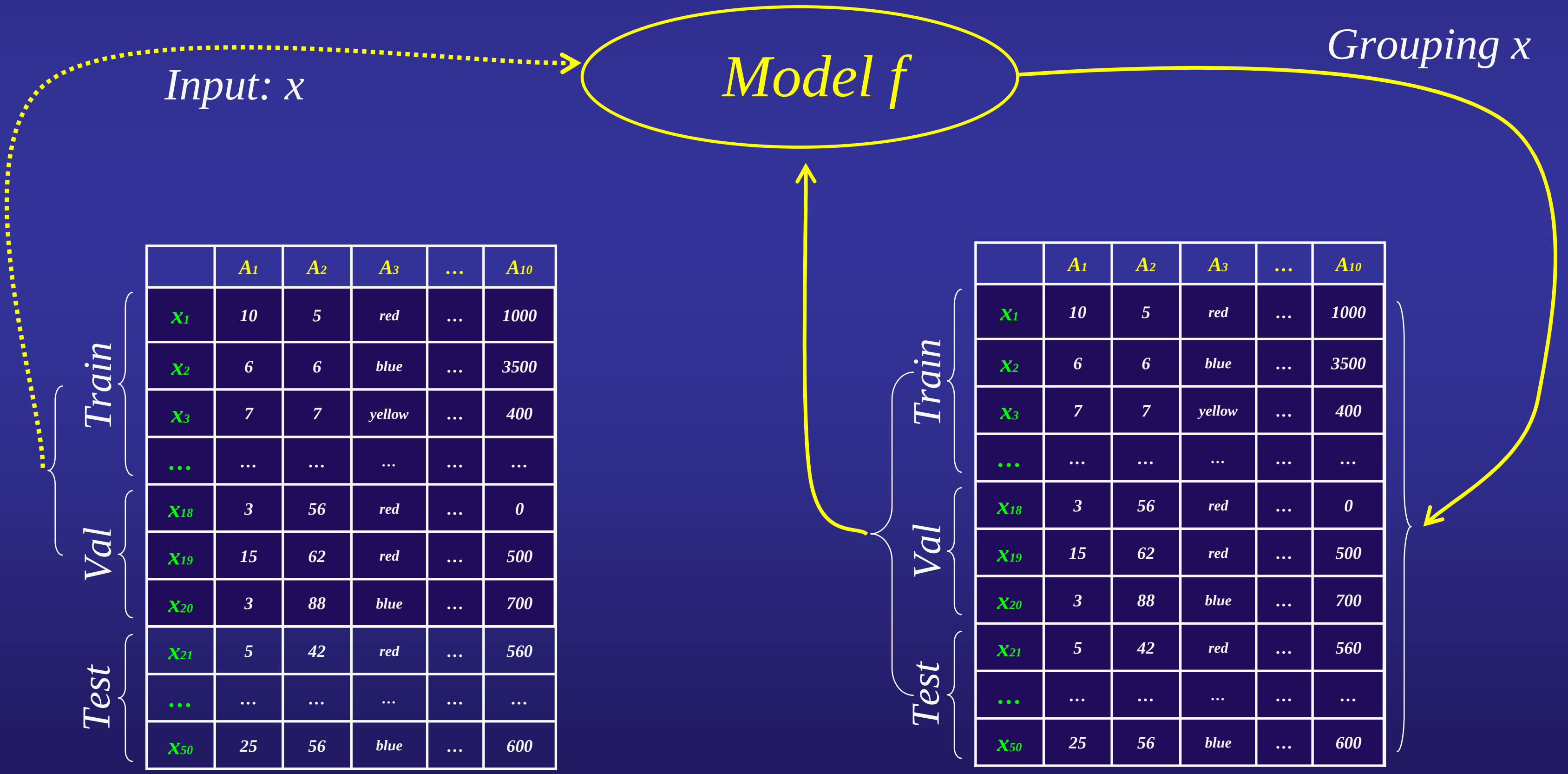
		A_1	A_2	A_3	…	A_{10}	y
Train	x_1	10	5	red	…	1000	1
	x_2	6	6	blue	…	3500	20
	x_3	7	7	yellow	…	400	45
	…	…	…	…	…	…	…
Val	x_{18}	3	56	red	…	0	30
	x_{19}	15	62	red	…	500	100
	x_{20}	3	88	blue	…	700	3
Test	x_{21}	5	42	red	…	560	?
	…	…	…	…	…	…	?
	x_{50}	25	56	blue	…	600	?

		A_1	A_2	A_3	…	A_{10}	y
Train	x_1	10	5	red	…	1000	1
	x_2	6	6	blue	…	3500	20
	x_3	7	7	yellow	…	400	45
	…	…	…	…	…	…	…
Val	x_{18}	3	56	red	…	0	30
	x_{19}	15	62	red	…	500	100
	x_{20}	3	88	blue	…	700	3
Test	x_{21}	5	42	red	…	560	?
	…	…	…	…	…	…	?
	x_{50}	25	56	blue	…	600	?

Unsupervised Learning

Learning or Training

Prediction or Test



		A_1	A_2	A_3	…	A_{10}
Train	x_1	10	5	red	…	1000
	x_2	6	6	blue	…	3500
	x_3	7	7	yellow	…	400
	…	…	…	…	…	…
Val	x_{18}	3	56	red	…	0
	x_{19}	15	62	red	…	500
	x_{20}	3	88	blue	…	700
Test	x_{21}	5	42	red	…	560
	…	…	…	…	…	…
	x_{50}	25	56	blue	…	600

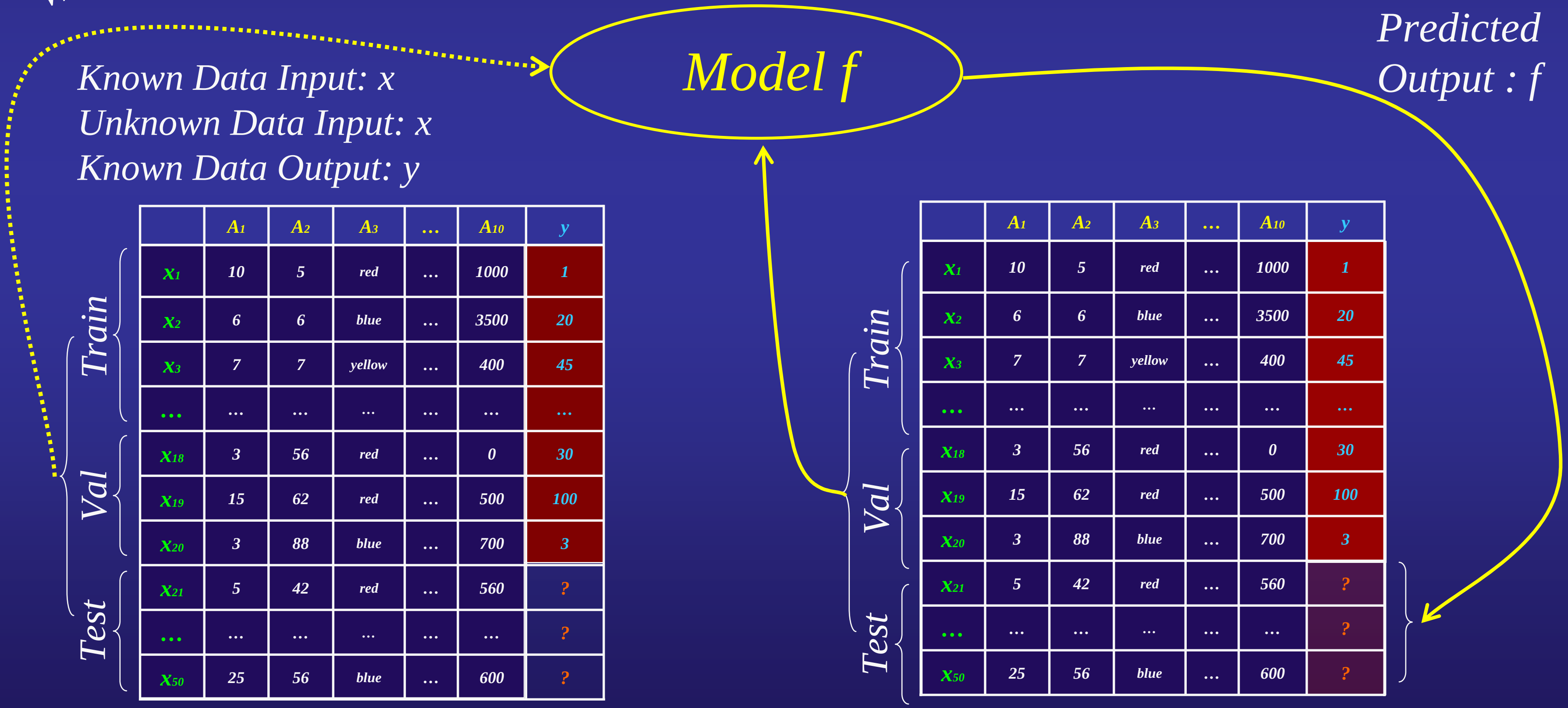
		A_1	A_2	A_3	…	A_{10}
Train	x_1	10	5	red	…	1000
	x_2	6	6	blue	…	3500
	x_3	7	7	yellow	…	400
	…	…	…	…	…	…
Val	x_{18}	3	56	red	…	0
	x_{19}	15	62	red	…	500
	x_{20}	3	88	blue	…	700
Test	x_{21}	5	42	red	…	560
	…	…	…	…	…	…
	x_{50}	25	56	blue	…	600

Semi-Supervised Learning

Learning or Training

Prediction or Test

Why "Semi-" ? :In learning, supervised for known data, $f(x) \approx y$,
but unsupervised for known data, $f(x) \approx ??$



Known Data Input: x
Unknown Data Input: x
Known Data Output: y

		A_1	A_2	A_3	…	A_{10}	y
Train	x_1	10	5	red	…	1000	1
	x_2	6	6	blue	…	3500	20
	x_3	7	7	yellow	…	400	45
	…	…	…	…	…	…	…
Val	x_{18}	3	56	red	…	0	30
	x_{19}	15	62	red	…	500	100
	x_{20}	3	88	blue	…	700	3
Test	x_{21}	5	42	red	…	560	?
	…	…	…	…	…	…	?
	x_{50}	25	56	blue	…	600	?

		A_1	A_2	A_3	…	A_{10}	y
Train	x_1	10	5	red	…	1000	1
	x_2	6	6	blue	…	3500	20
	x_3	7	7	yellow	…	400	45
	…	…	…	…	…	…	…
Val	x_{18}	3	56	red	…	0	30
	x_{19}	15	62	red	…	500	100
	x_{20}	3	88	blue	…	700	3
Test	x_{21}	5	42	red	…	560	?
	…	…	…	…	…	…	?
	x_{50}	25	56	blue	…	600	?

Models

Content

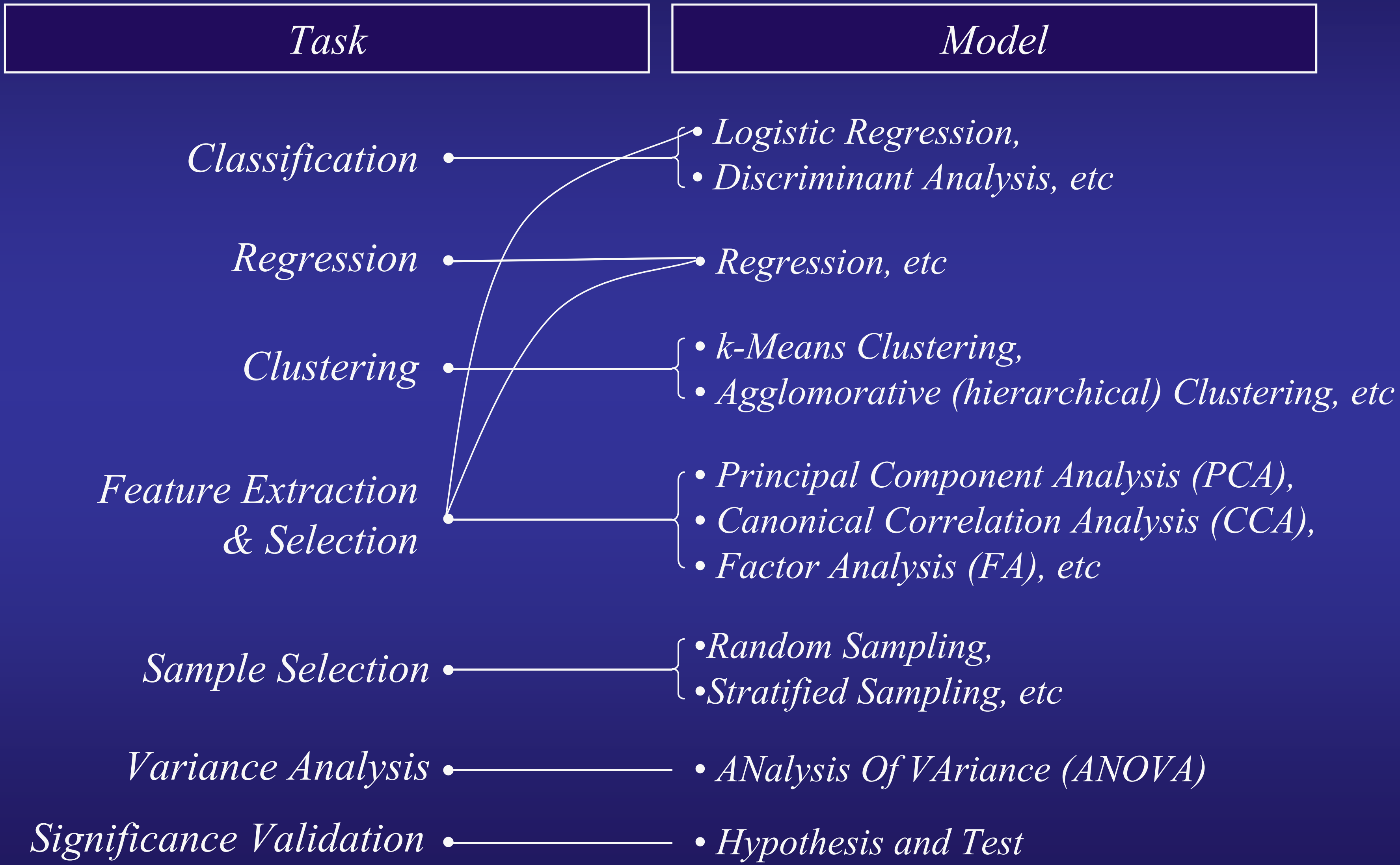
1. Basics
2. Tasks
3. Learning
4. Models with Examples

Models

Models

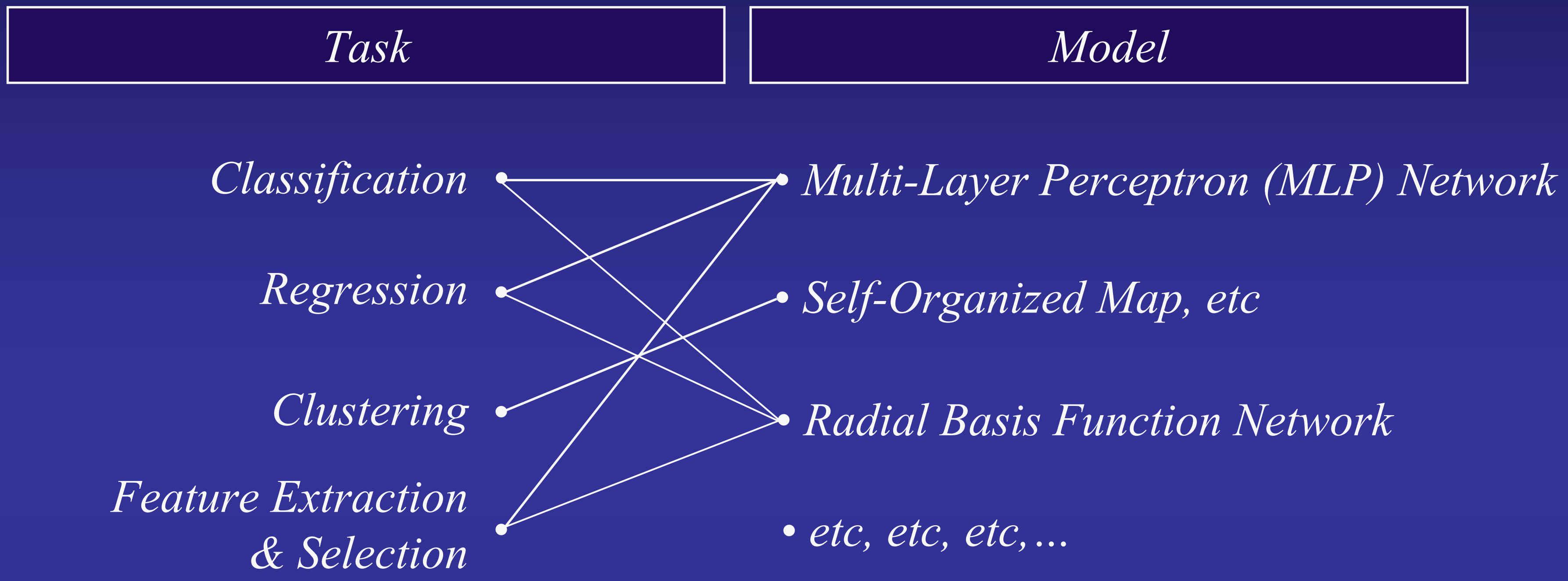
- Traditional Statistical Methods
- Neural Networks
- Decision Trees
- Kernel Methods
- Semi-Supervised Learning (SSL)
- Ensemble Methods
- Generative (Probabilistic) Methods

Models: Traditional Statistical Models



Task	Model
Classification	• Logistic Regression, • Discriminant Analysis, etc
Regression	Regression, etc
Clustering	• k-Means Clustering, • Agglomorative (hierarchical) Clustering, etc
Feature Extraction & Selection	• Principal Component Analysis (PCA), • Canonical Correlation Analysis (CCA), • Factor Analysis (FA), etc
Sample Selection	•Random Sampling, •Stratified Sampling, etc
Variance Analysis	• ANalysis Of VAriance (ANOVA)
Significance Validation	• Hypothesis and Test

Models: Neural Networks



Task

Model

Classification

Regression

Clustering

Feature Extraction
& Selection

Multi-Layer Perceptron (MLP) Network

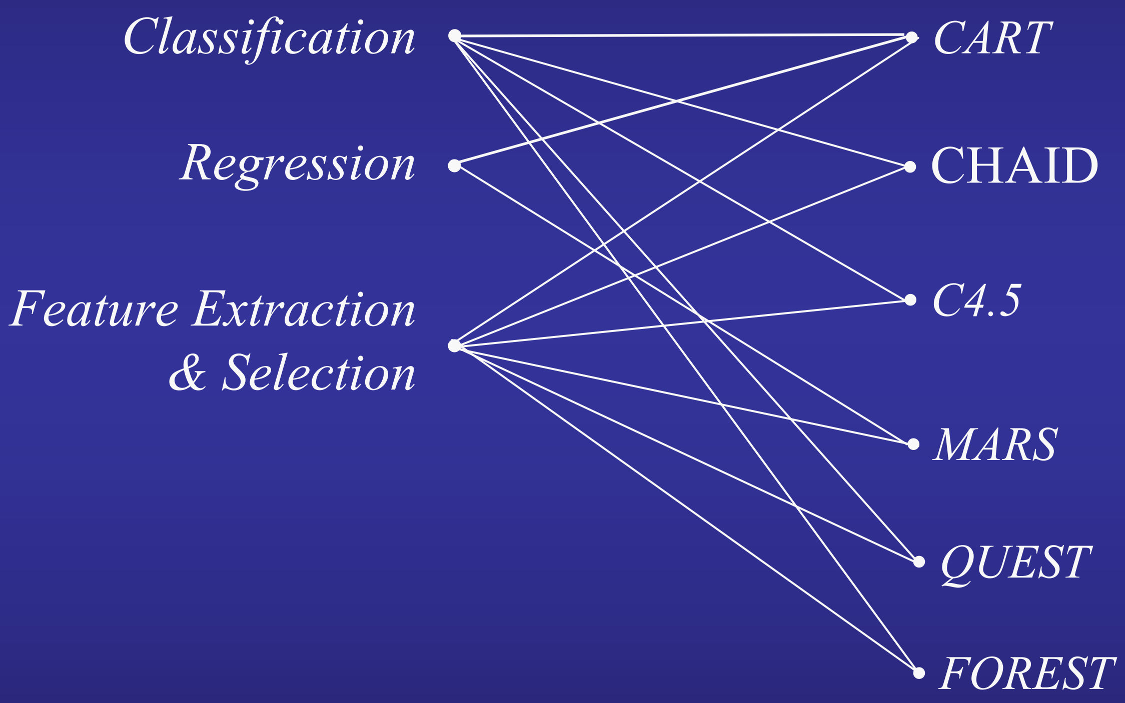
Self-Organized Map, etc

Radial Basis Function Network

• etc, etc, etc,…

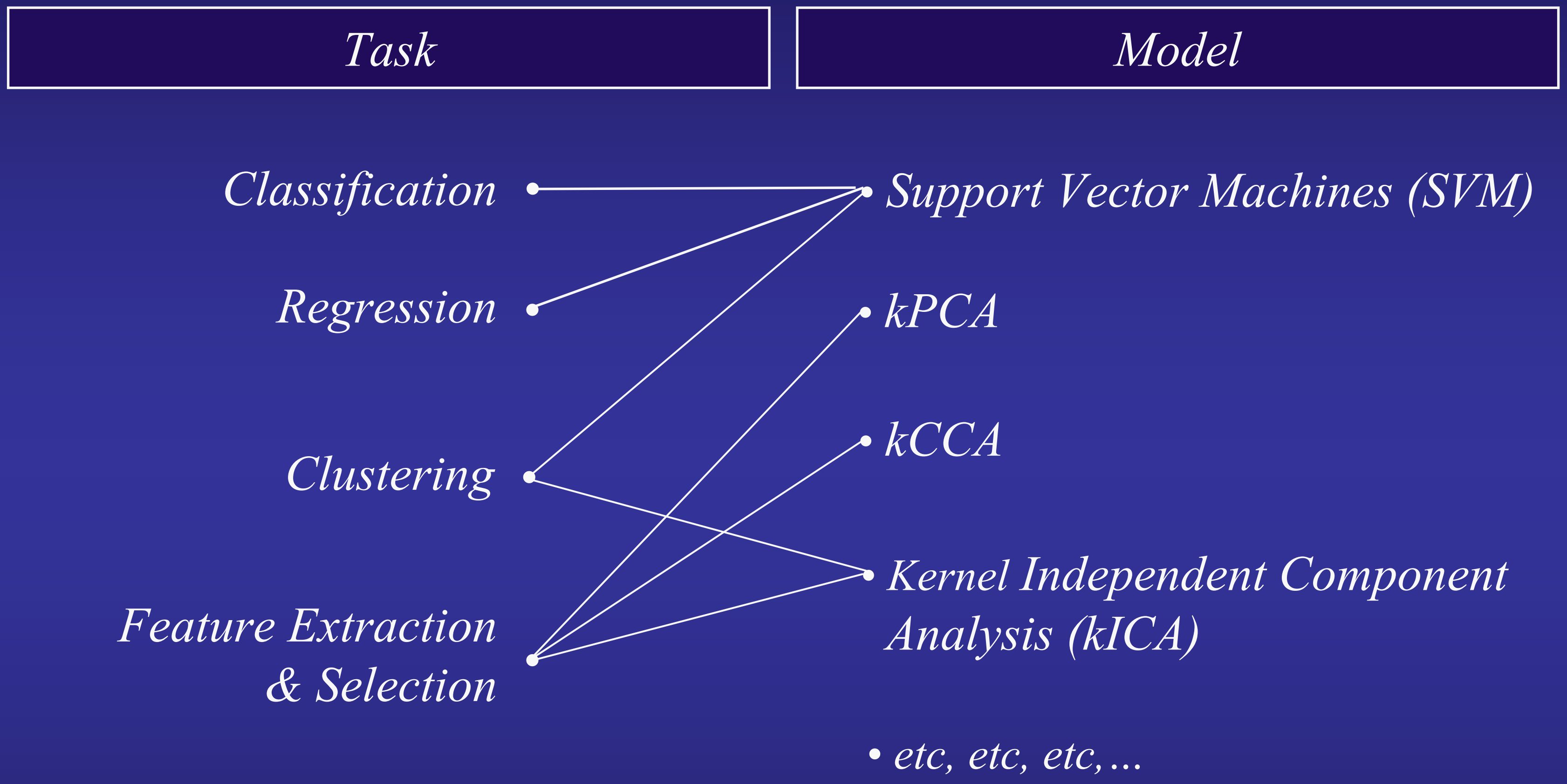
Models: Decision Trees (or Rule-base)

Task	*Model*



- *etc, etc, etc,…*

Models: Kernel Methods



Task

Model

Classification

Regression

Clustering

Feature Extraction & Selection

Support Vector Machines (SVM)

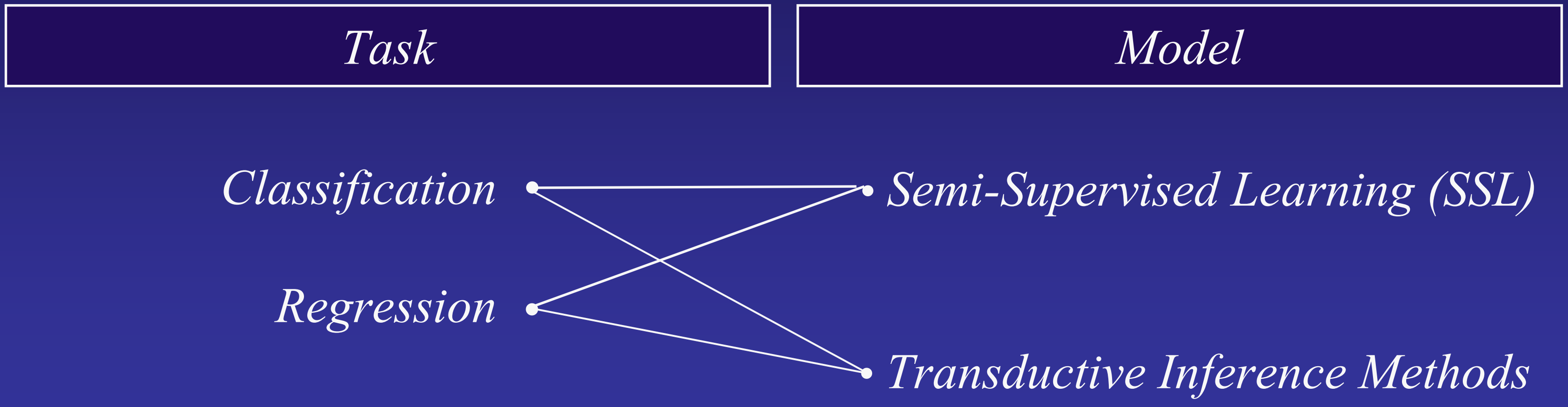
kPCA

kCCA

Kernel Independent Component Analysis (kICA)

- etc, etc, etc,…

Models: Semi-Supervised Learning



Task

Model

Classification

Regression

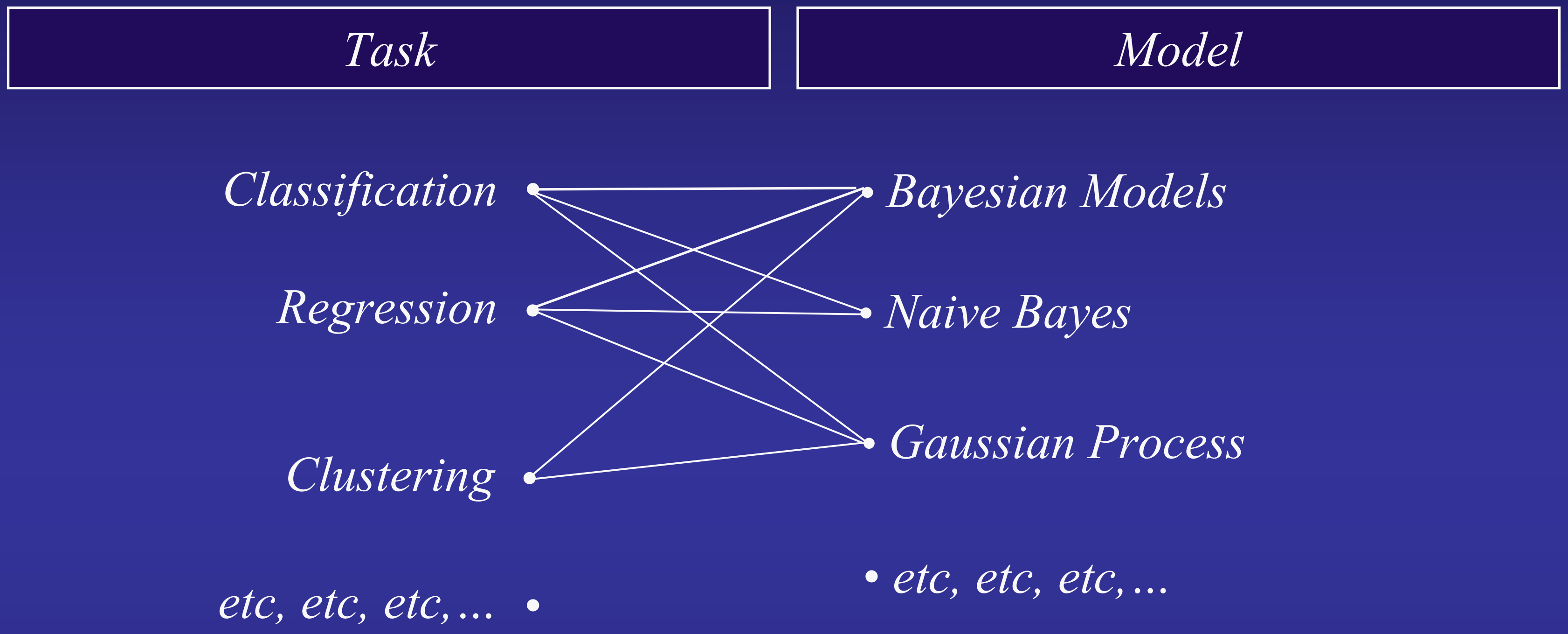
Semi-Supervised Learning (SSL)

Transductive Inference Methods

- etc, etc, etc,…

* Note that, currently, the term of "Semi-Supervised Learning" has been used as a name of "model" as well as the concept of "learning scheme"

Models: Generative (Probabilistic) Methods



Task

Model

Classification

Regression

Clustering

etc, etc, etc,…

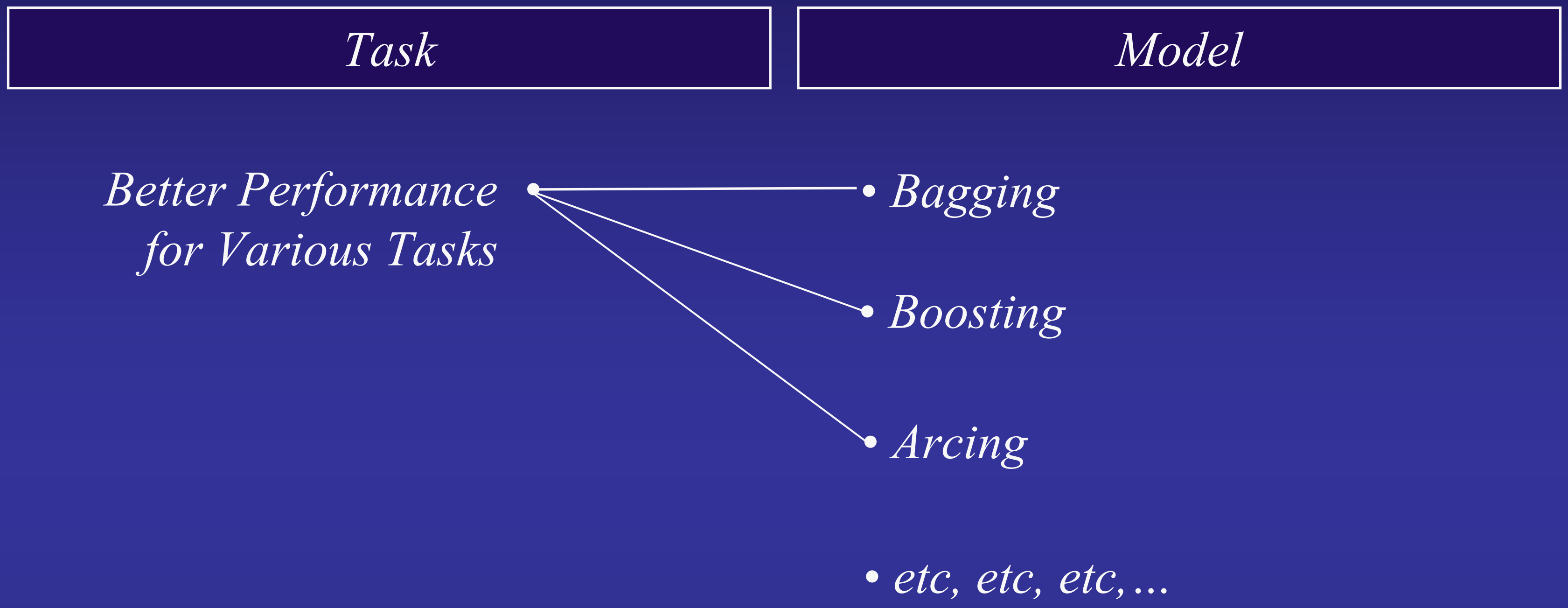
Bayesian Models

Naive Bayes

Gaussian Process

etc, etc, etc,…

Models: Ensemble Methods



Task

Model

Better Performance for Various Tasks

- Bagging
- Boosting
- Arcing
- etc, etc, etc,…

The Most Up-To-Date Models

Kernel Methods

Support Vector Machines (SVM), kPCA, kCCA, kICA, etc

Semi-Supervised Learning Methods

Graph-based SSL,Transductive Inference Methods, etc

* *Note that, currently, the term of "Semi-Supervised Learning" has been used as a name of "model" as well as the concept of "learning scheme"*

The Most Up-To-Date Models

Kernel Methods

Support Vector Machines (SVM), kPCA, kCCA, kICA, etc

Semi-Supervised Learning Methods

Graph-based SSL, Transductive Inference Methods

Kernel Methods:
Support Vector Machines (SVM)

Kernel Methods

Why KM?

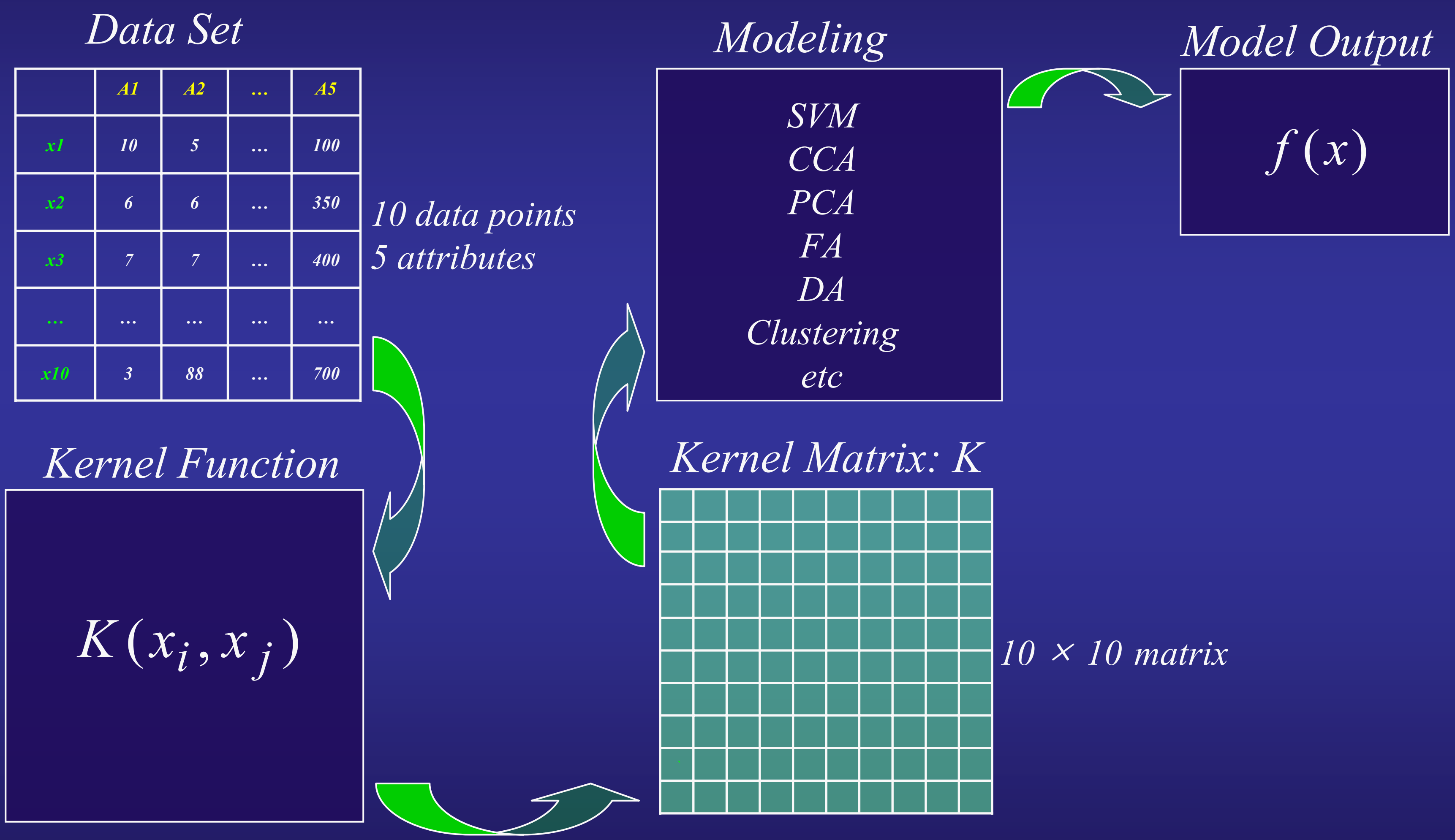
- *Kernel methods can operate on very general types of data and can detect very general types of relations*

- *Various tasks--* $\left\{\begin{array}{l} \textit{PCA,} \\ \textit{CCA,} \\ \textit{FA,} \\ \textit{DA,} \\ \textit{Clustering} \end{array}\right\}$ *can be performed on diverse data* $\left\{\begin{array}{c} \textit{vectors,} \\ \textit{sequences,} \\ \textit{text,} \\ \textit{images,} \\ \textit{graphs} \end{array}\right\}$

- *Integration of different types of data is easy and natural*

Kernel Methods

Procedure



Data Set

	A1	A2	…	A5
x1	10	5	…	100
x2	6	6	…	350
x3	7	7	…	400
…	…	…	…	…
x10	3	88	…	700

10 data points
5 attributes

Kernel Function

$K(x_i, x_j)$

Kernel Matrix: K

10 × 10 matrix

Modeling

SVM
CCA
PCA
FA
DA
Clustering
etc

Model Output

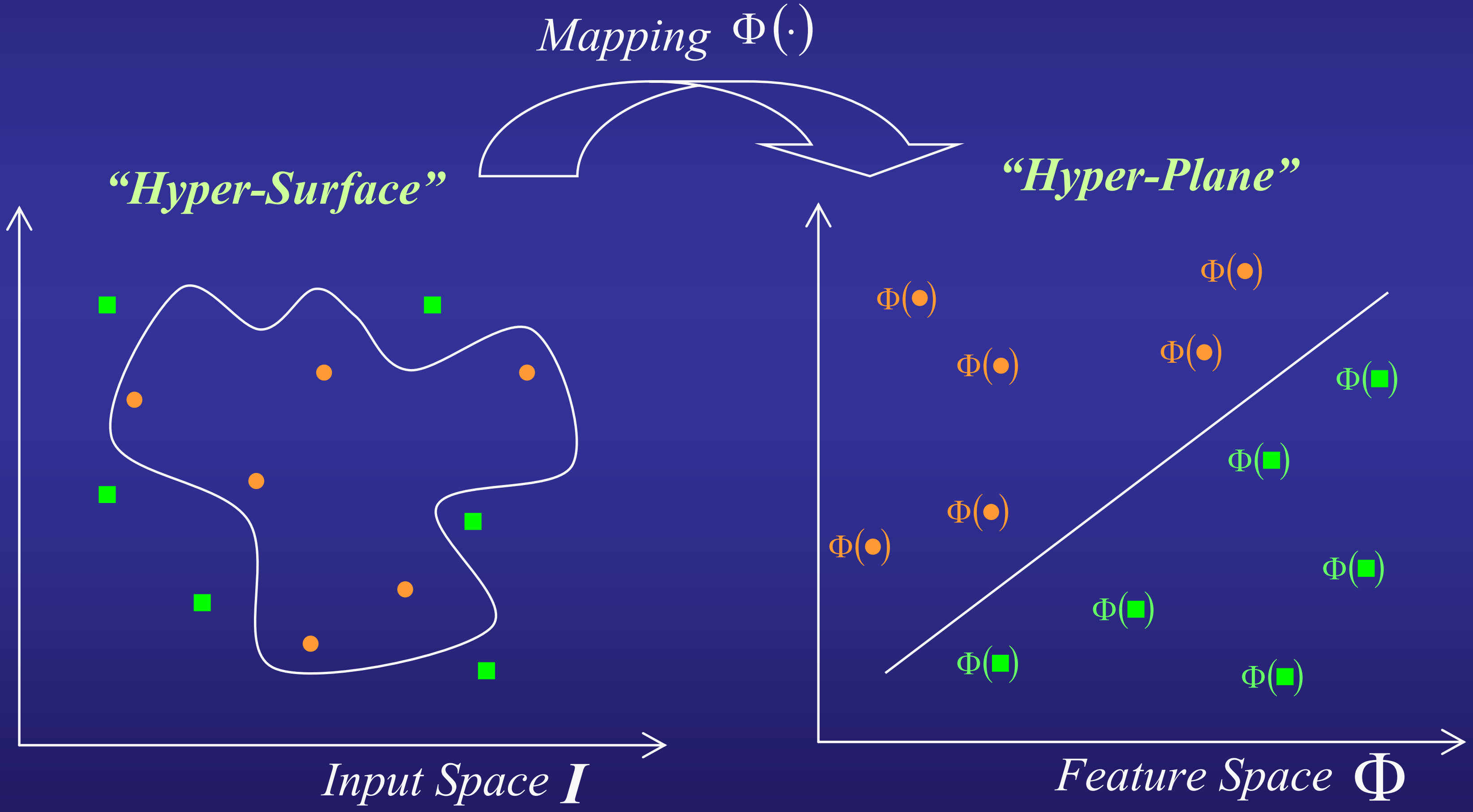
$f(x)$

$$K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$$ *where Φ(.) is a mapping function*

Kernel Methods

Feature Space

KM operates in Feature Space!



Mapping $\Phi(\cdot)$

"Hyper-Surface"

"Hyper-Plane"

Input Space $\boldsymbol{I}$

Feature Space $\boldsymbol{\Phi}$

Kernel Methods

Feature Space

The Mapping from Input to Feature space is…

- *Highly Nonlinear*
- *Dimension Expanding (up to infinite dim.)*
- *Not unique to a Feature Space, Probably Unknown*

Finding the mapping function has been the most difficult barrier in the traditional statistics and early machine learning algorithms

Kernel Methods

Kernel Function

Kernel Function: $K(\boldsymbol{x}, \boldsymbol{y}) = \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{y})$

In KM, those difficulties could be circumvented
by means of "Kernel Trick"
which replaces the dot product between mapping functions

Kernel Methods

Kernel Function

Kernel Function: $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y})$

Functions Satisfying Mercers's Theorem

Polynomial kernels $K(x,y) = (x \cdot y)^P$

Radial Basis (Gaussian) kernels $K(x,y) = \exp\left(\frac{-\|x - y\|^2}{2\sigma^2}\right)$

Sigmoid Kernels (3-MLP NN) $K(x,y) = \tanh\{\kappa(x \cdot y) + \Theta)\}$

A Single Kernel Produces Multiple Mappings

Ex) Input Space : R^{2,} Polynomial Kernel

$$K(\vec{x},\vec{y}) = (\vec{x}\cdot\vec{y})^2 = \left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\cdot\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}\right)^2 = \Phi(\vec{x})\cdot\Phi(\vec{y})$$

(1) Feature Space : R^3

$$\Phi(\vec{\mathrm{x}}) = \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix}$$

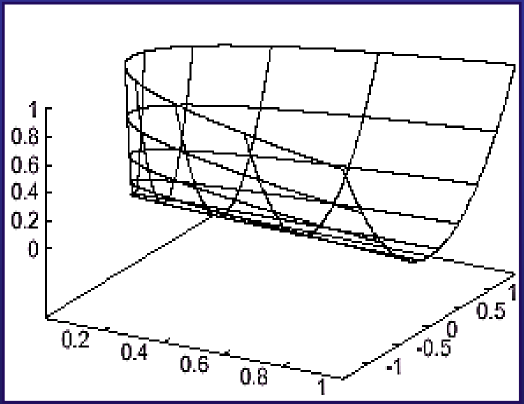
(2) Feature Space : R^4

$$\Phi(\vec{\mathrm{x}}) = \frac{1}{\sqrt{2}}\begin{pmatrix} (x_1^2 - x_1^2) \\ 2x_1x_2 \\ (x_1^2 + x_2^2) \end{pmatrix}$$

(3) Feature Space : R^4

$$\Phi(\vec{\mathrm{x}}) = \begin{pmatrix} x_1^2 \\ x_1x_2 \\ x_1x_2 \\ x_2^2 \end{pmatrix}$$

$$(\vec{\mathrm{x}}\cdot\vec{\mathrm{y}})^2 = \left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\cdot\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}\right)^2 = \left(\begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix}\cdot\begin{pmatrix} y_1^2 \\ \sqrt{2}y_1y_2 \\ y_2^2 \end{pmatrix}\right) = \Phi(\vec{\mathrm{x}})\cdot\Phi(\vec{\mathrm{y}})$$



Kernel Methods

The flexible combination of appropriate kernel design and relevant kernel algorithms has given rise to a powerful class of methods, whose computational and statistical properties are well understood

Particularly, KM has increasingly been used in in Bioinformatics as diverse as biosequences and microarray data analysis, etc.

SVM Classification

Basic Idea of SVM

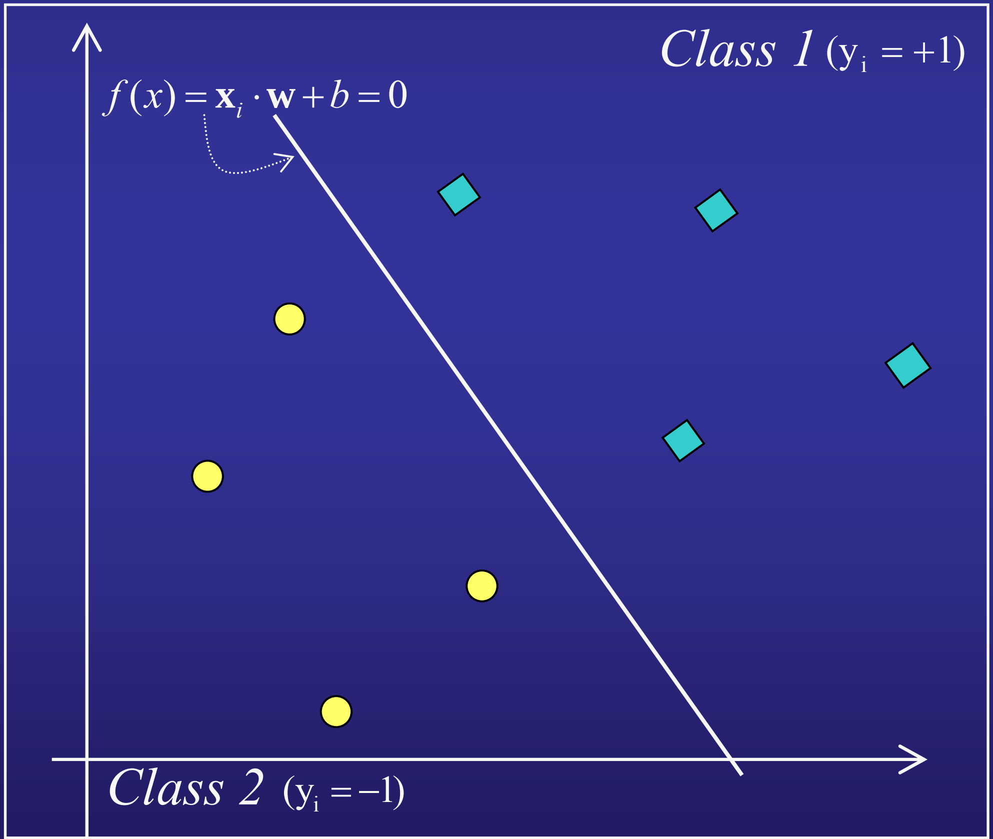
Properties of SVM *…Optional*

- *Margin*
- *Convexity*
- *Duality*
- *Kernels*
- *Sparseness*

Basic Idea of SVM

Basic Idea of SVM

SVM looks for the Separating Hyperplane with the Largest Margin.



Training data

$$\{\mathbf{x}_i, y_i\},\ i = 1,\dots,l, \quad y_i \in \{-1,1\}$$

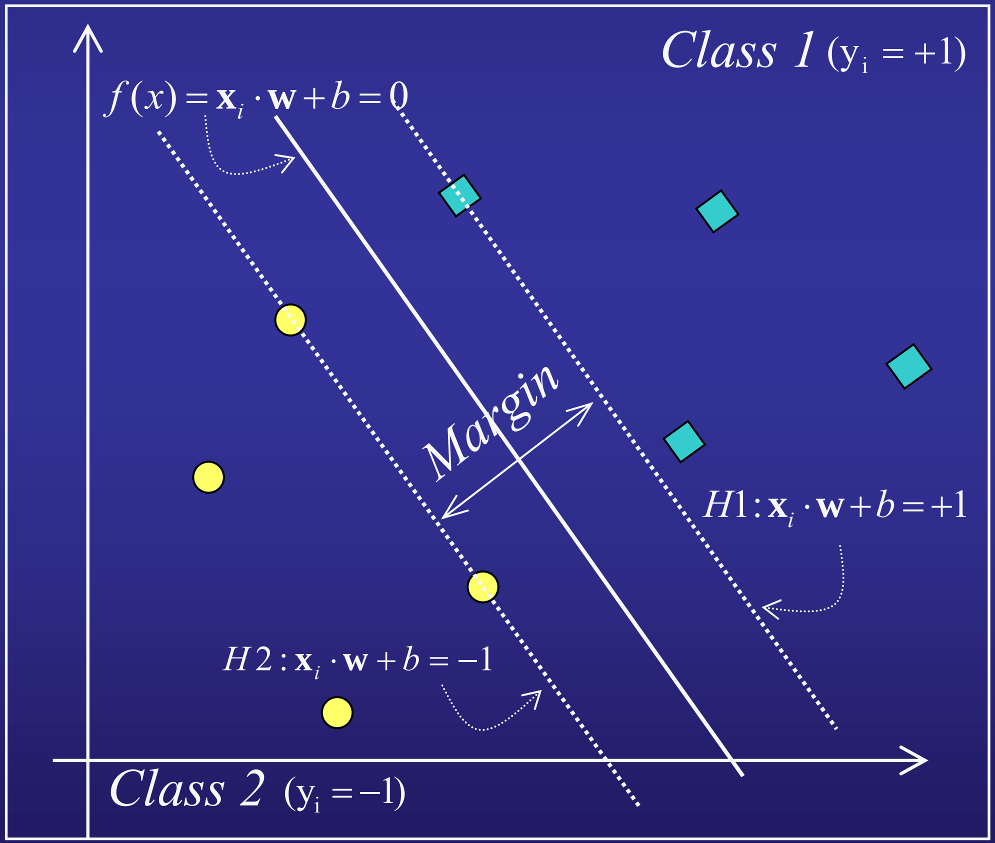
Separating Hyperplane

$$f(x) = \mathbf{w} \cdot \mathbf{x} + b = 0$$

$$\text{sign}(f(x)) = \begin{cases} +1 & \textit{if}\ \boldsymbol{x}\cdot\boldsymbol{w}+b \geq 0 \\ -1 & \textit{if}\ \boldsymbol{x}\cdot\boldsymbol{w}+b < 0 \end{cases}$$

Basic Idea of SVM

SVM looks for the Separating Hyperplane with the Largest Margin.


Class 1 $(y_i = +1)$
$f(x) = \mathbf{x}_i \cdot \mathbf{w} + b = 0$
Margin
$H1: \mathbf{x}_i \cdot \mathbf{w} + b = +1$
$H2: \mathbf{x}_i \cdot \mathbf{w} + b = -1$
Class 2 $(y_i = -1)$

Supporting Hyperplanes

$$H1: \mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \quad \text{for } y_i = +1$$

$$H2: \mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \quad \text{for } y_i = -1$$

Margin

Distance between H1 and H2

$$\frac{|1-b|}{\|\mathbf{w}\|} - \frac{|-1-b|}{\|\mathbf{w}\|} = \frac{\mathbf{2}}{\|\mathbf{w}\|}$$

Basic Idea of SVM

Find the Pair of Hyperplanes (Support Vectors)

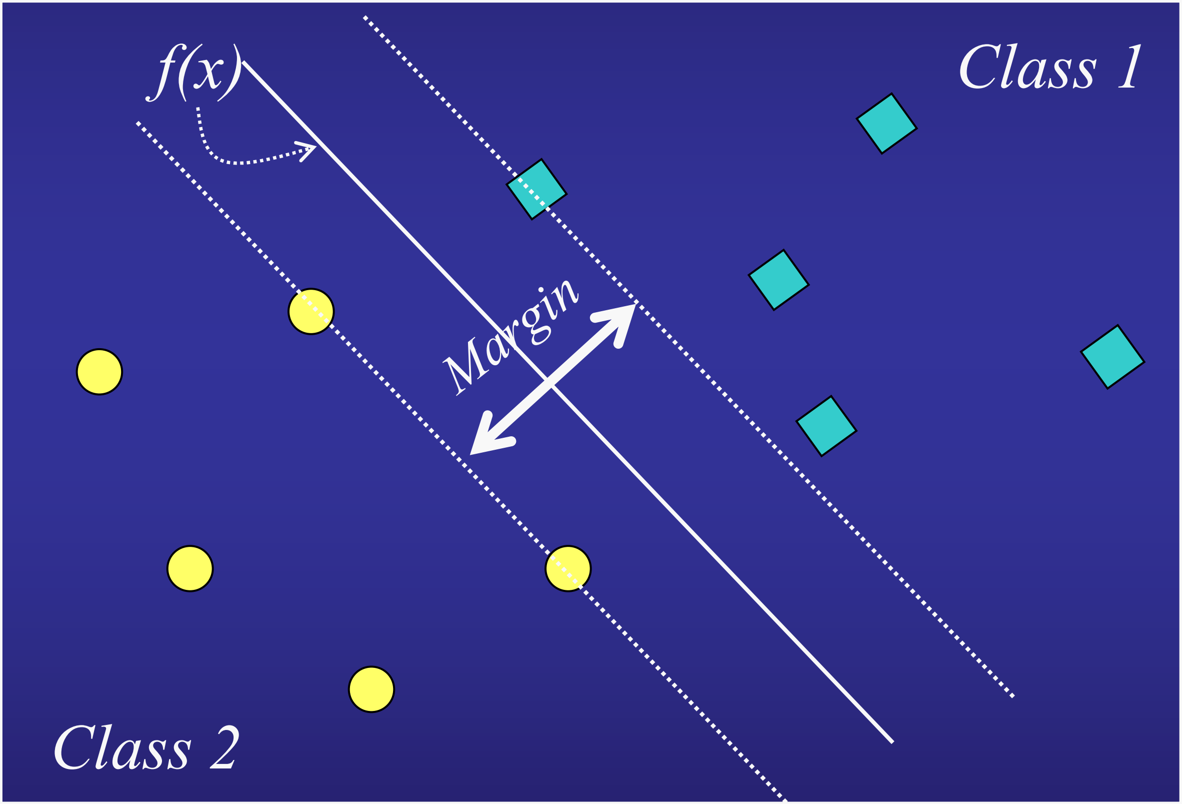
$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \quad \text{for } \mathrm{y_i} = +1$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \quad \text{for } \mathrm{y_i} = -1$$

under the constraints which gives Maximum Margin $\frac{2}{\|\mathbf{w}\|}$ *!*

Basic Idea of SVM

Separable Case



Basic Idea of SVM

Separable Case

Minimize $||w||^2$ *under the constraints !!*

$$\min \frac{1}{2} \| \mathbf{w} \|^2$$

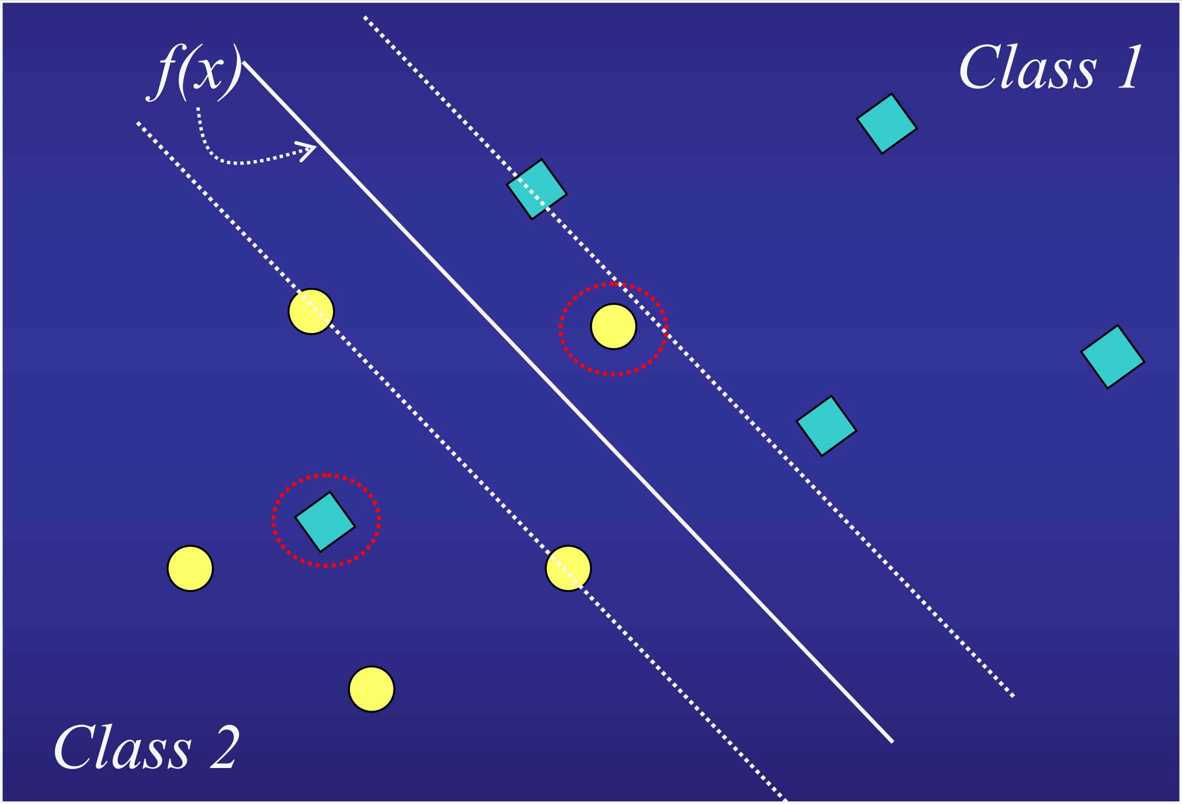
$$s.t. \quad y_i(\mathbf{x_i} \cdot \mathbf{w} + b) - 1 \geqslant 0 \quad \forall_{\mathrm{i}}$$

Quadratic Programming

(convex QP : obj ftn is convex, constraints form a convex set)

Basic Idea of SVM

Non-Separable Case ?



Basic Idea of SVM

Use Slack Variables !

NonSeparable Case

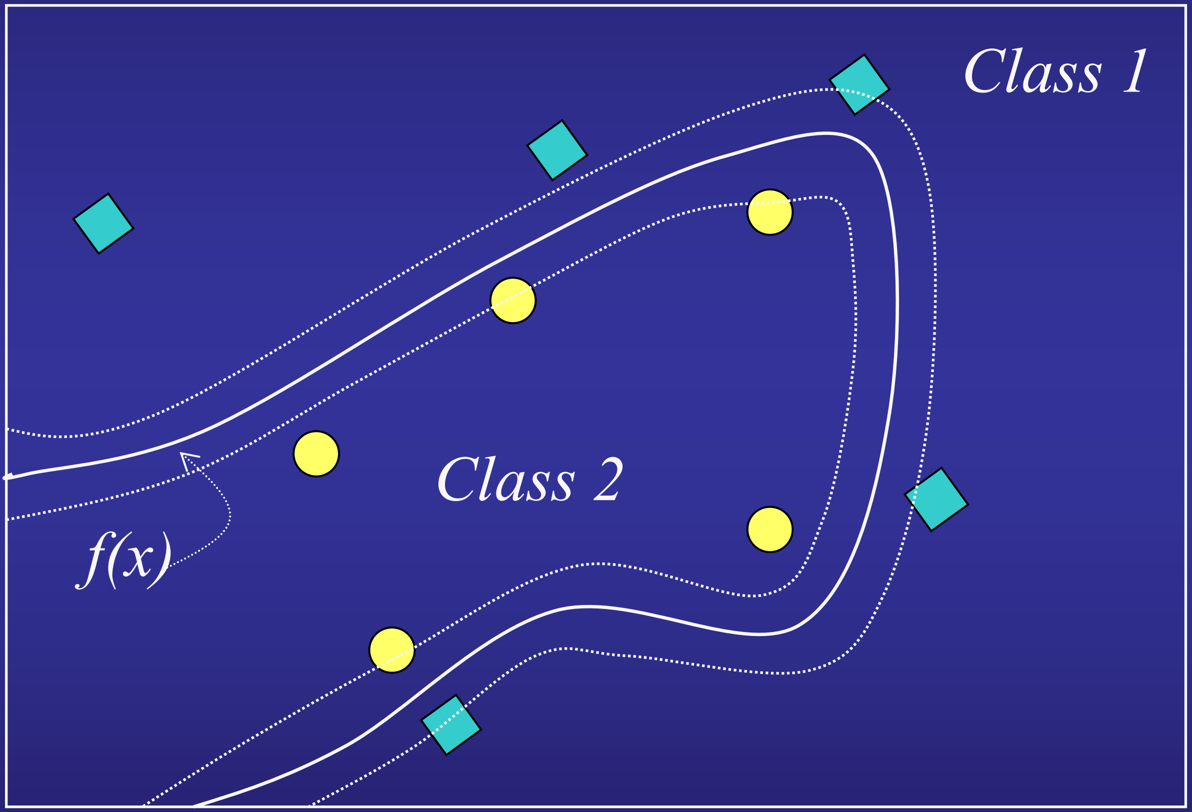
$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i$$

$$s.t.\quad y_i(\mathbf{x_i}\cdot\mathbf{w}+b) \ \geq 1-\xi_i, \quad \forall_{\mathrm{i}}$$

C: Error Tolerance Parameter

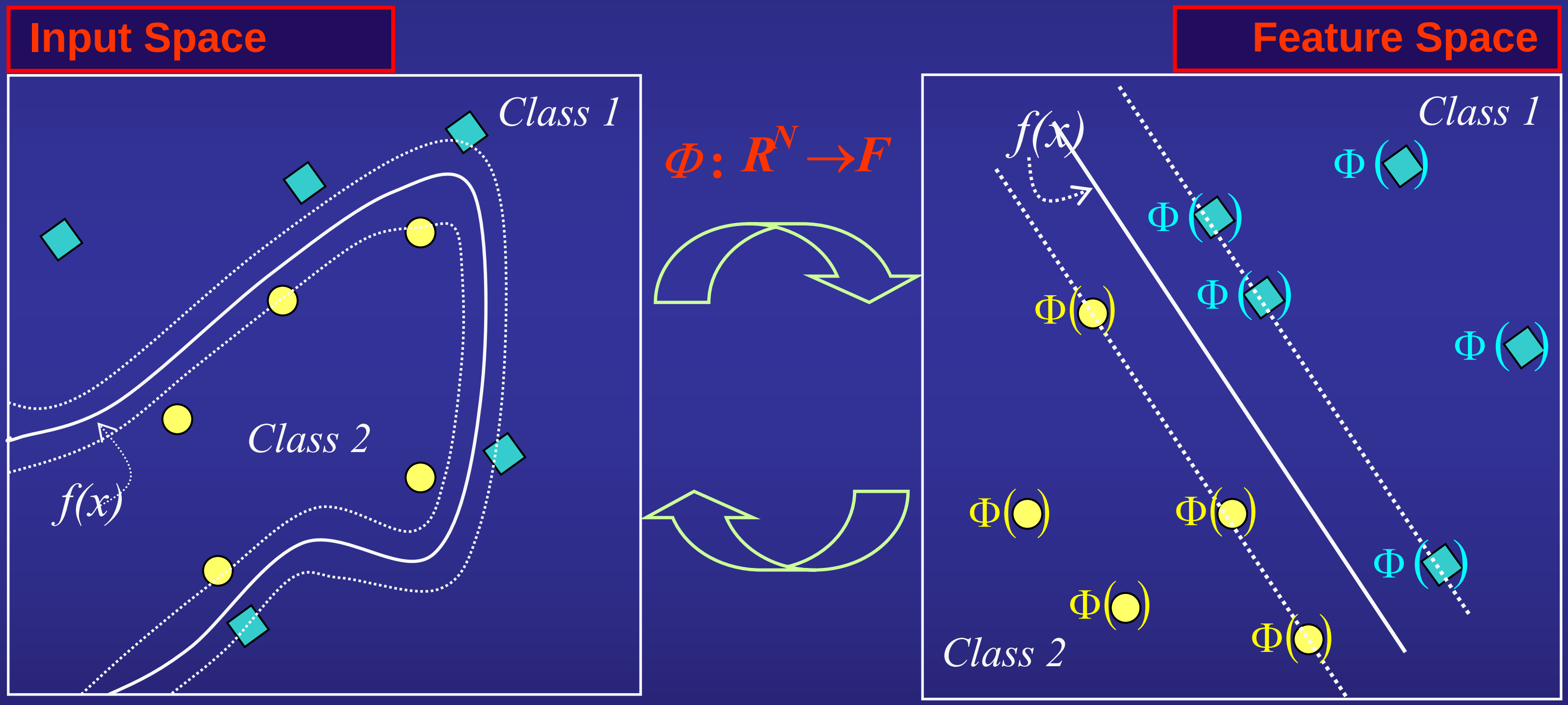
Basic Idea of SVM

Nonlinear Case ?



Basic Idea of SVM

Solve (linear) problem in the Feature Space !



NonLinear Algorithm in Input Space

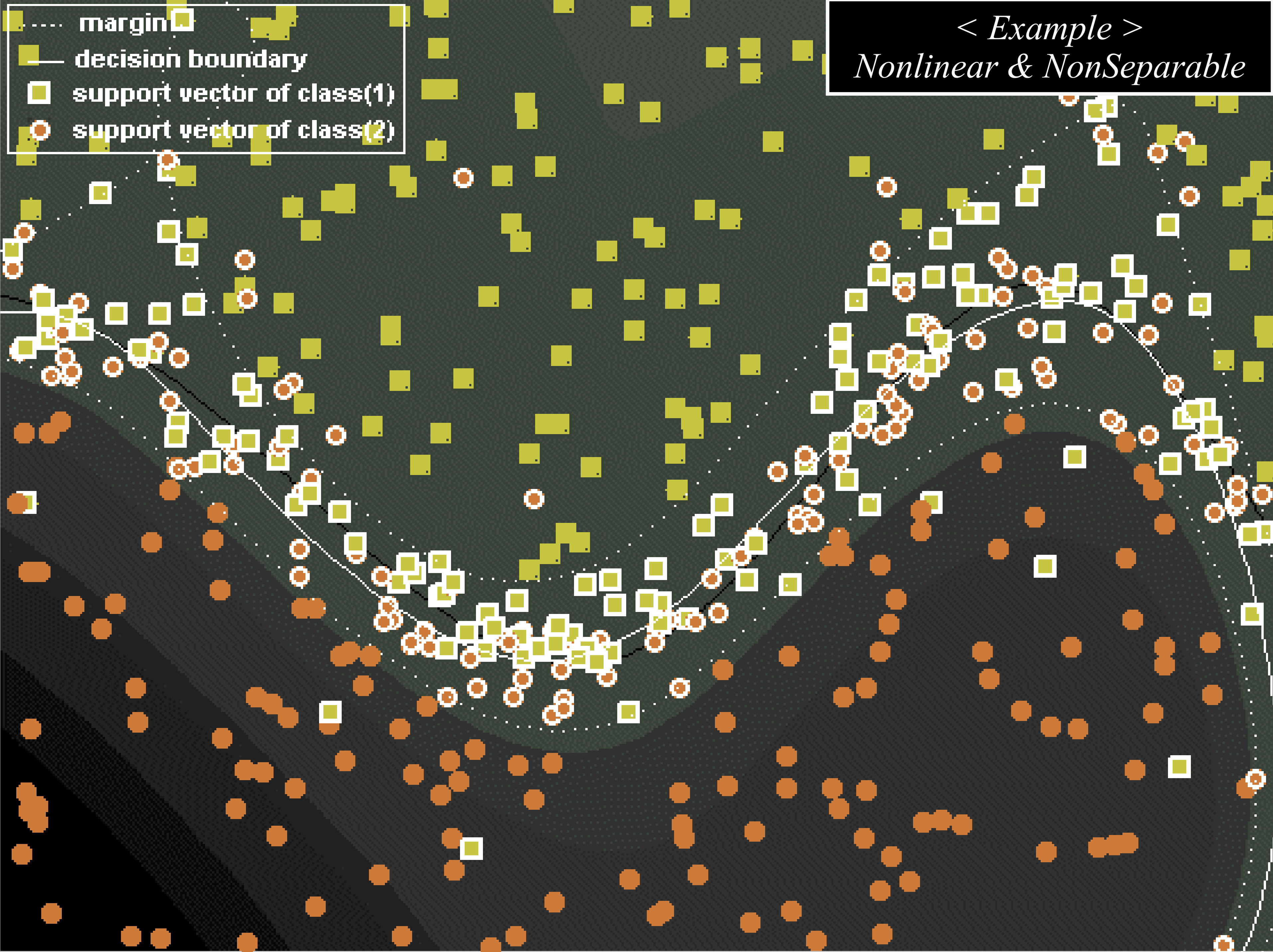
Linear Algorithm in Feature Space

Basic Idea of SVM

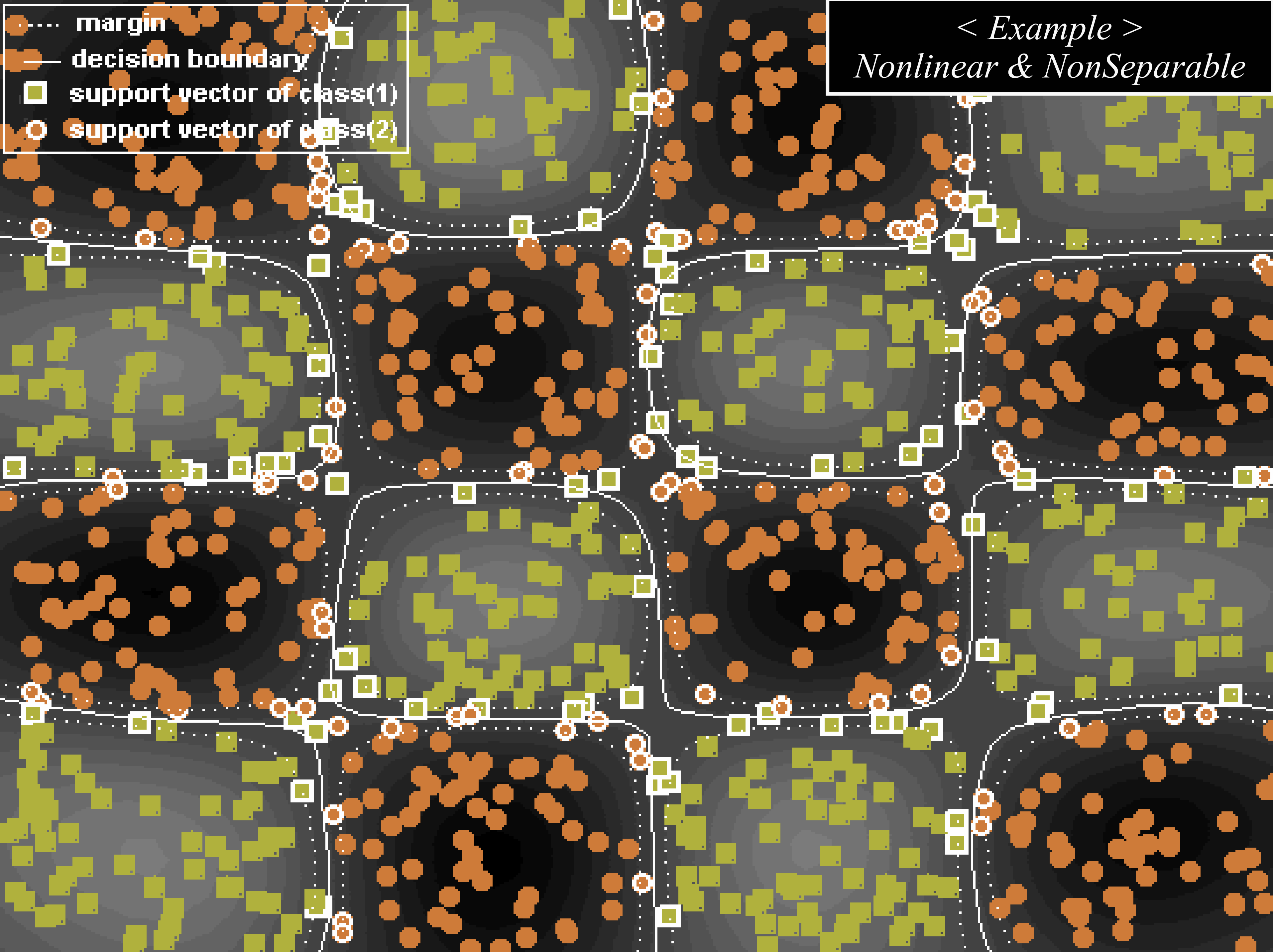
Feature Space

SVMs map the training data nonlinearly into a higher-dimensional feature space via ϕ and construct a separating hyperplane with maximum margin there.

This yields a nonlinear decision boundary in input space.



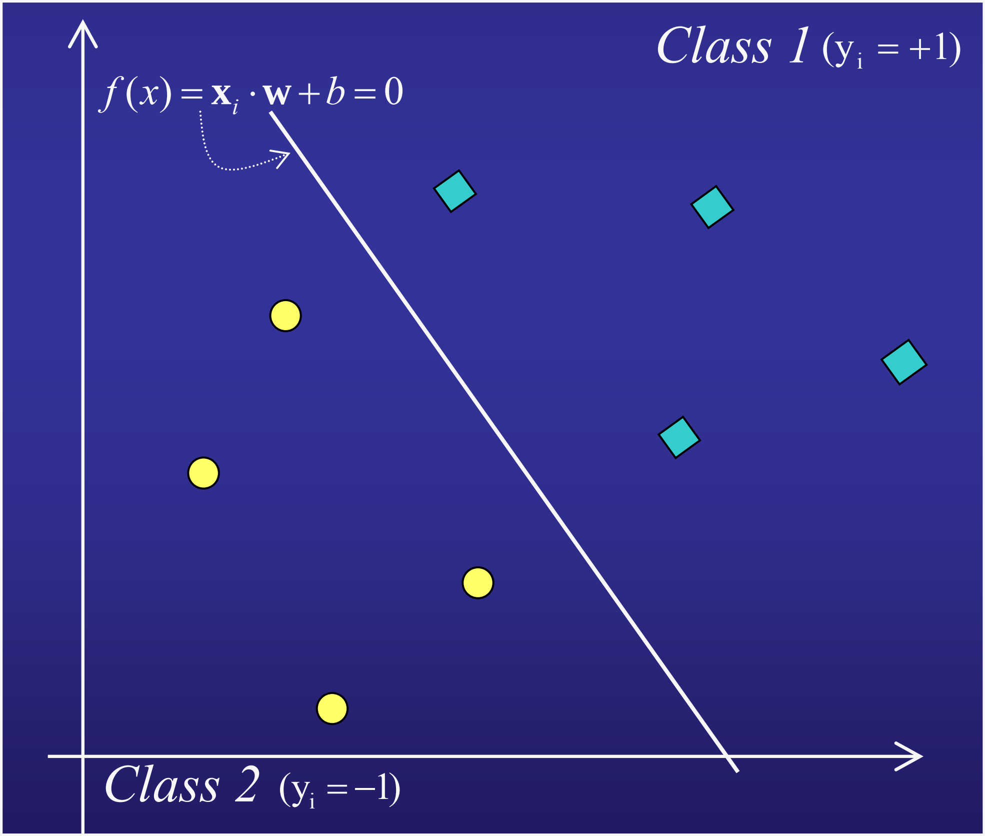
< Example >
Nonlinear & NonSeparable


< Example >
Nonlinear & NonSeparable
margin
decision boundary
support vector of class(1)
support vector of class(2)

Properties of SVM *…optional*

[Margin] *Convexity Duality Kernel Sparseness*

SVM looks for the Separating Hyperplane with the Largest Margin.



Training data

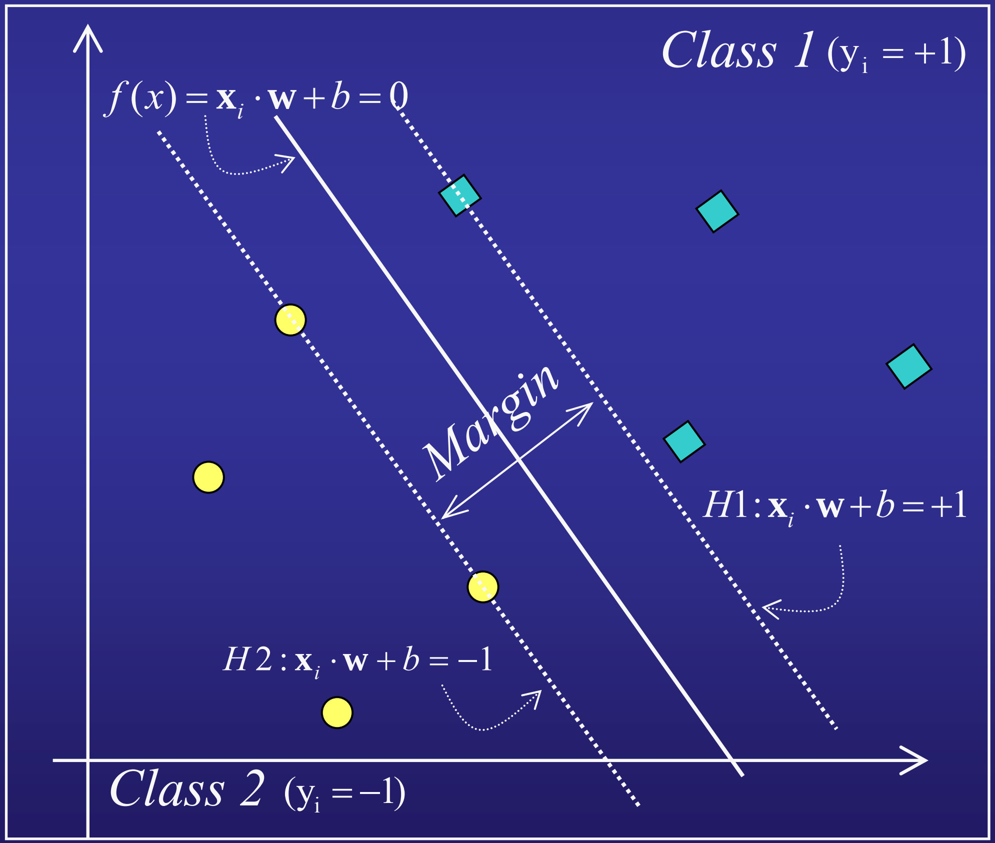
$$\{\mathbf{x}_i, y_i\},\ i = 1, \dots, l, \quad y_i \in \{-1, 1\}$$

Separating Hyperplane

$$f(x) = \mathbf{w} \cdot \mathbf{x} + b = 0$$

$$\mathrm{sign}(f(x)) = \begin{cases} +1 & \text{if } \boldsymbol{x} \cdot \boldsymbol{w} + b \geq 0 \\ -1 & \text{if } \boldsymbol{x} \cdot \boldsymbol{w} + b < 0 \end{cases}$$

SVM looks for the Separating Hyperplane with the Largest Margin.



Class 1 $(y_i = +1)$
$f(x) = \mathbf{x}_i \cdot \mathbf{w} + b = 0$
Margin
$H1: \mathbf{x}_i \cdot \mathbf{w} + b = +1$
$H2: \mathbf{x}_i \cdot \mathbf{w} + b = -1$
Class 2 $(y_i = -1)$

Supporting Hyperplanes

$$H1: \mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \quad \text{for } y_i = +1$$

$$H2: \mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \quad \text{for } y_i = -1$$

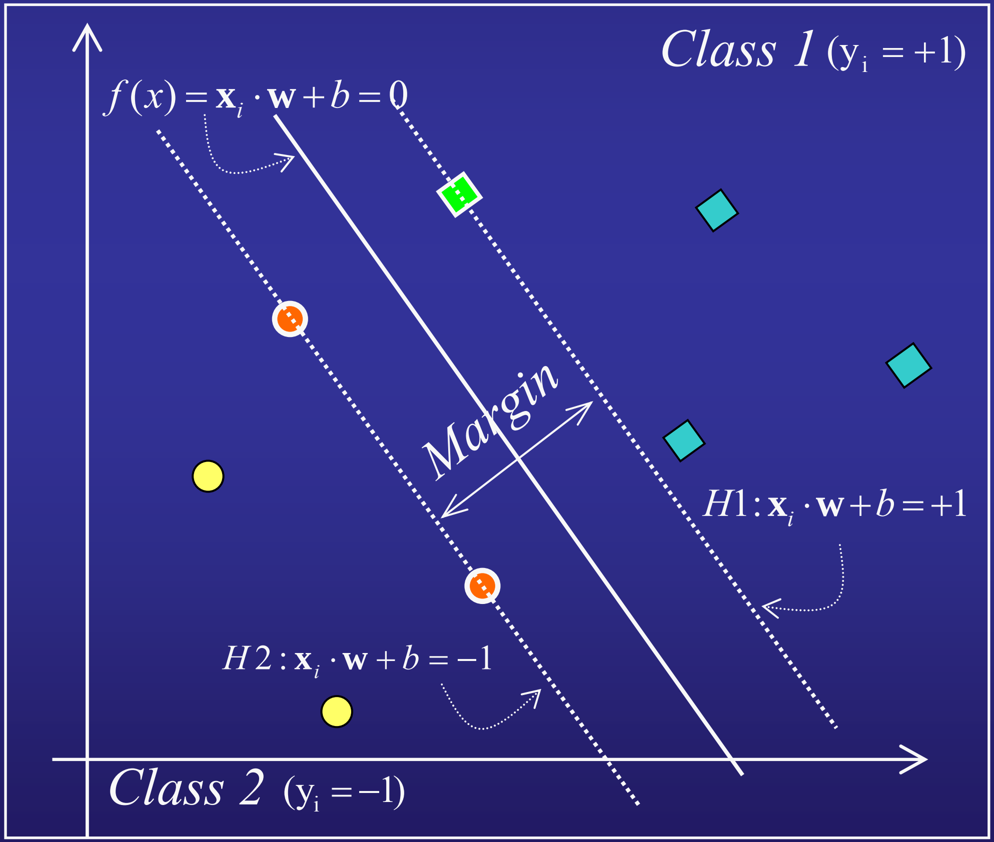
Margin

Distance between H1 and H2

$$\frac{|1 - b|}{\|\mathbf{w}\|} - \frac{|-1 - b|}{\|\mathbf{w}\|} = \frac{\mathbf{2}}{\|\mathbf{w}\|}$$

[Margin] *Convexity* *Duality* *Kernel* *Sparseness*

SVM looks for the Separating Hyperplane with the Largest Margin.


Class 1 $(y_i = +1)$
$f(x) = \mathbf{x}_i \cdot \mathbf{w} + b = 0$
Margin
$H1: \mathbf{x}_i \cdot \mathbf{w} + b = +1$
$H2: \mathbf{x}_i \cdot \mathbf{w} + b = -1$
Class 2 $(y_i = -1)$

Support Vectors

$$H1:\ \mathbf{x}_i \cdot \mathbf{w} + b - 1 = 0 \quad \text{for } y_i = +1$$

$$H2:\ \mathbf{x}_i \cdot \mathbf{w} + b + 1 = 0 \quad \text{for } y_i = -1$$

x_i's are the Closest Data from Separating Hyperplane,

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

[Margin] *Convexity* *Duality* *Kernel* *Sparseness*

Find the Pair of Hyperplanes (Support Vectors)

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \quad \text{for } \mathrm{y}_\mathrm{i} = +1$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \quad \text{for } \mathrm{y}_\mathrm{i} = -1$$

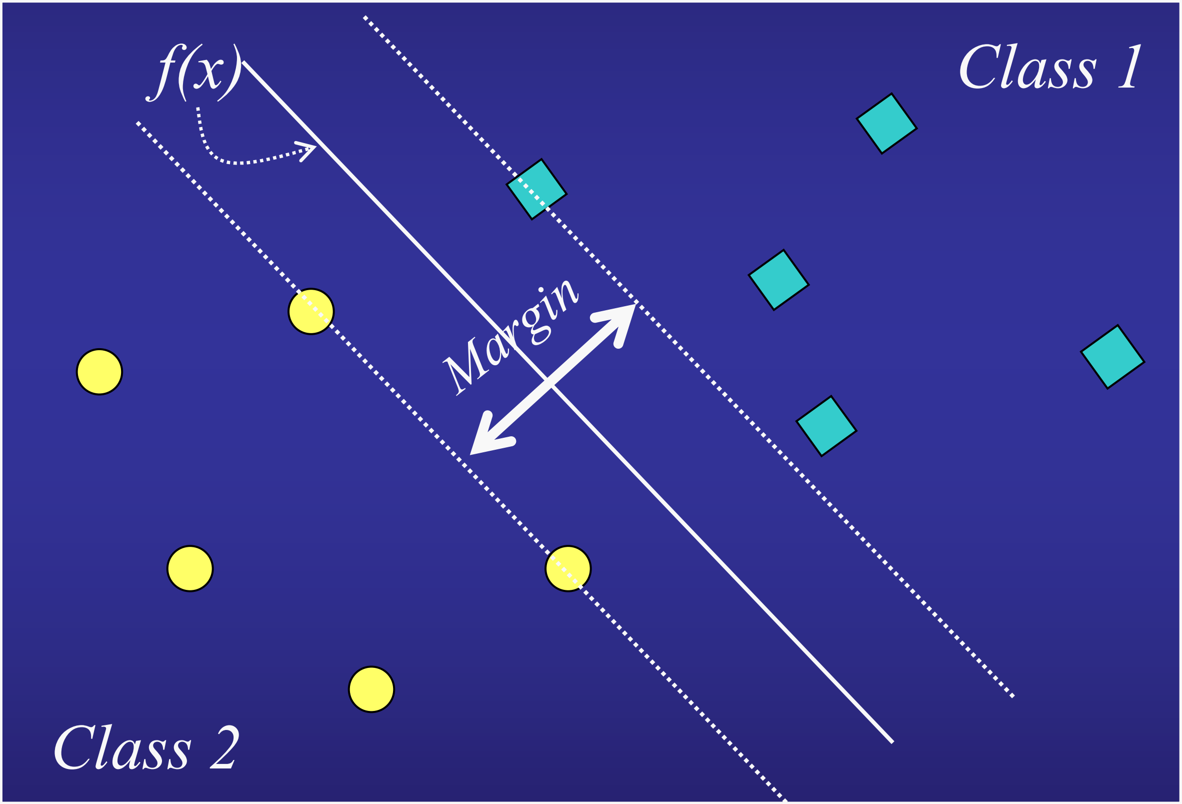
under the constraints which gives Maximum Margin $\frac{2}{\|\mathbf{w}\|}$ *!*

Minimize $||w||^2$ *under the constraints !!*

$$\min \frac{1}{2} \|\mathbf{w}\|^2$$

$$s.t. \quad y_i(\mathbf{x_i} \cdot \mathbf{w} + b) - 1 \geqslant 0 \quad \forall_\mathrm{i}$$

Separable Case



[Margin] [Convexity] *Duality Kernel Sparseness*

Minimize $||w||^2$ under the constraints !!

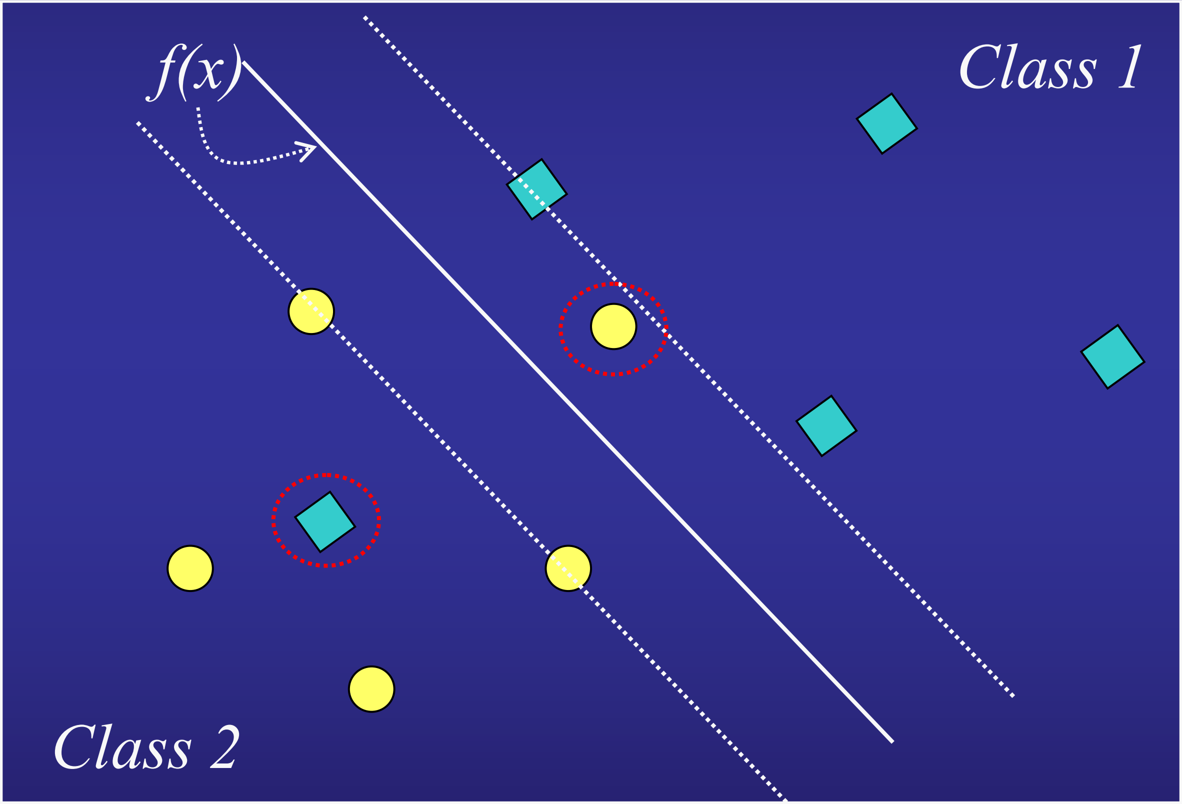
$$\min \frac{1}{2} \| \mathbf{w} \|^2$$

$$s.t. \quad y_i(\mathbf{x_i} \cdot \mathbf{w} + b) - 1 \geqslant 0 \quad \forall_{\mathrm{i}}$$

Quadratic Programming

(convex QP : obj ftn is convex, constraints form a convex set)

Non-Separable Case ?



[Margin] [Convexity] *Duality Kernel Sparseness*

Use Slack Variables !

Separable Case

Problem

$$\min \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$s.t.\quad y_i(\boldsymbol{x_i}\cdot\boldsymbol{w}+b)-1 \geqslant 0 \quad \forall_{\mathrm{i}}$$

NonSeparable Case

Problem

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i$$

$$s.t.\quad y_i(\mathbf{x_i}\cdot\mathbf{w}+b) \quad \geq 1-\xi_i, \quad \forall_{\mathrm{i}}$$

C: Error Tolerance Parameter

Primal Problem

Minimize ||w||^2 under the constraints !!

$$\min \frac{1}{2} \| \mathbf{w} \|^2$$

$$s.t. \quad y_i(\mathbf{x_i} \cdot \mathbf{w} + b) - 1 \geqslant 0 \quad \forall_{\mathrm{i}}$$

How to Solve?

Use Lagrange theory !

(Karush-Kuhn-Tucker Condition)

Karush-Kuhn-Tucker Condition

Min: f(x)

s.t. h(x) = 0 (m equality constraints)

g(x) $\leq$ 0 (k inequality constraints)

Lagrangian: L(x,a,m) = f(x) + a h(x) + ∑u_i (g_i (x) + s_i)

1) Gradient of the Lagrangian = 0

2) Constraints: h(x) = 0 & g(x) $\leq$ 0

3) Complementary Slackness: u.s = 0

4) Feasibility for the inequality constraints: s $\geq$ 0

5) Sign condition on the inequality multipliers: u >= 0

KKT conditions are satisfied
at the solution of any constrained optimization problem

For convex problem,
KKT conditions are necessary and sufficient condition
for primal, dual solution.

Primal Problem

$$\min \frac{1}{2} \| \mathbf{w} \|^2$$

$$s.t.\ \ y_i(\mathbf{x_i} \cdot \mathbf{w} + b) \text{-} 1 \geqslant 0 \quad \forall_{\mathrm{i}}$$

Lagrangian

$$L(w,b) \equiv \frac{1}{2} \| \boldsymbol{w} \|^2 - \sum_i^l \alpha_i y_i (\boldsymbol{x_i} \cdot \boldsymbol{w} + b) + \sum_i^l \alpha_i$$

Margin **[Convexity]** *Duality Kernel Sparseness*

$$L(w,b) \equiv \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_i^l \alpha_i y_i (\boldsymbol{x_i} \cdot \boldsymbol{w} + b) + \sum_i^l \alpha_i$$

... Lagrangian

$$\left.\begin{aligned} \frac{\partial}{\partial w_v} L_P = w_v - \sum_i^l \alpha_i y_i \boldsymbol{x_i} = 0 \\ \frac{\partial}{\partial \boldsymbol{b}} \boldsymbol{L_P} = -\sum_i^l \alpha_i y_i = \mathbf{0} \end{aligned}\right\}$$

...Gradient of the Lagrangian = 0

$$y_i(\mathbf{x_i} \cdot \mathbf{w} + b) - 1 \geq 0 \quad \forall \mathrm{i}$$

... Primal Feasibility

$$\alpha_i \geq 0 \quad \forall \mathrm{i}$$

... Dual Feasibility

$$\alpha_i(y_i(\mathbf{x_i} \cdot \mathbf{w} + b) - 1) = 0 \quad \forall \mathrm{i}$$

... Complementarity Conditions

Solving the SVM problem is equivalent to finding a solution KKT conditions.

Lagrangian L has to be minimized w.r.t. the primal variables w and b and maximized w.r.t. the dual variables α_i

- *Minimize Lp with respect to w, b :*

$$\min \quad L_P \equiv \frac{1}{2}\| w \|^2 - \sum_{i}^{l} \alpha_i y_i (x_i \bullet w + b) + \sum_{i}^{l} \alpha_i$$

$$\mapsto \quad w = \sum_{i}^{l} \alpha_i y_i x_i \quad , \quad \sum_{i}^{l} \alpha_i y_i = 0$$

- *Maximize L_D with respect to α_i :*

$$\max \quad L_D \equiv \sum_{i}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j x_i \bullet x_j$$

$$s.t. \quad \alpha_i \geqslant 0, \quad \sum_{i}^{l} \alpha_i \; y_i = 0, \quad \forall i$$

Why Dual ?

$$\max \quad L_D \equiv \sum_{i}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j \boldsymbol{x_i} \bullet \boldsymbol{x}_j$$

$$\boldsymbol{s.t.} \quad \alpha_i \geqslant \mathbf{0}, \quad \sum_{i}^{l} \alpha_i \, \boldsymbol{y}_i = \boldsymbol{0}, \quad \forall \boldsymbol{i}$$

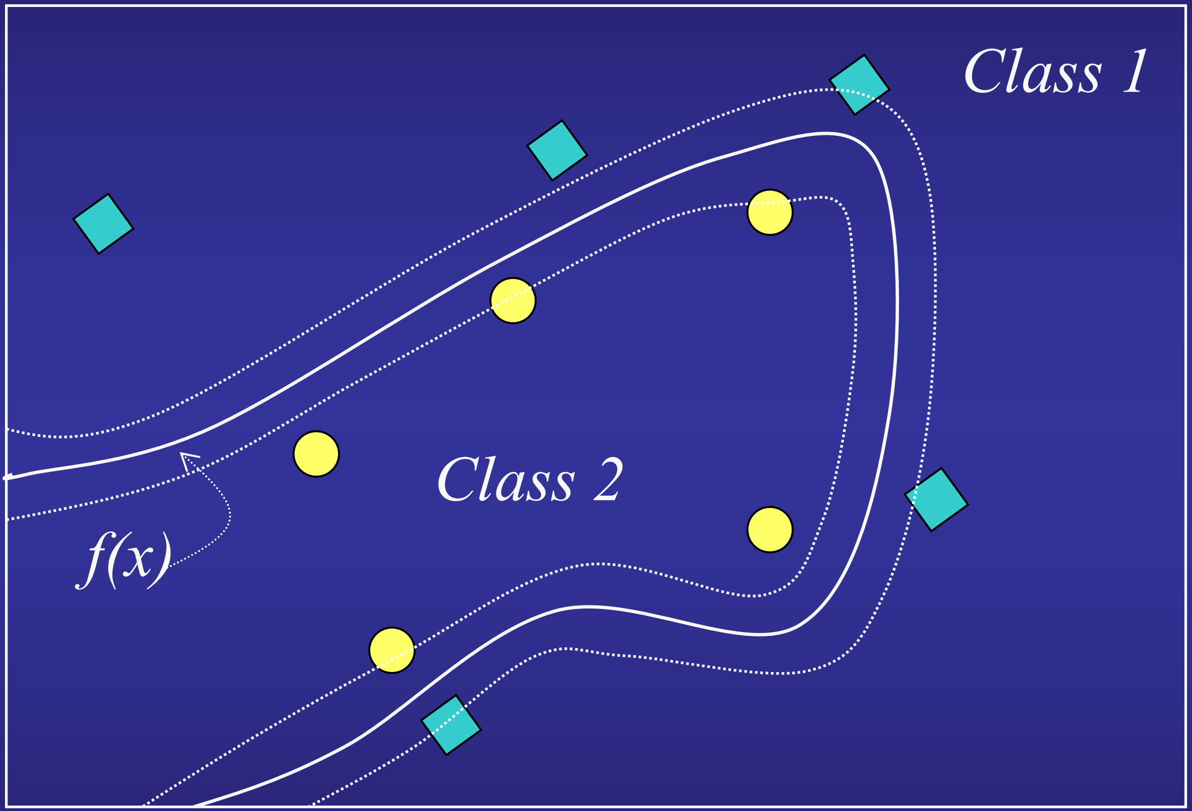
Why Dual ?

$$\max \quad L_D \equiv \sum_{i}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x_i} \bullet \mathbf{x}_j$$

$$s.t. \quad \alpha_i \geqslant 0, \quad \sum_{i}^{l} \alpha_i \, y_i = 0, \quad \forall i$$

Dot Product between Training Vectors:
We can use Kernel functions !

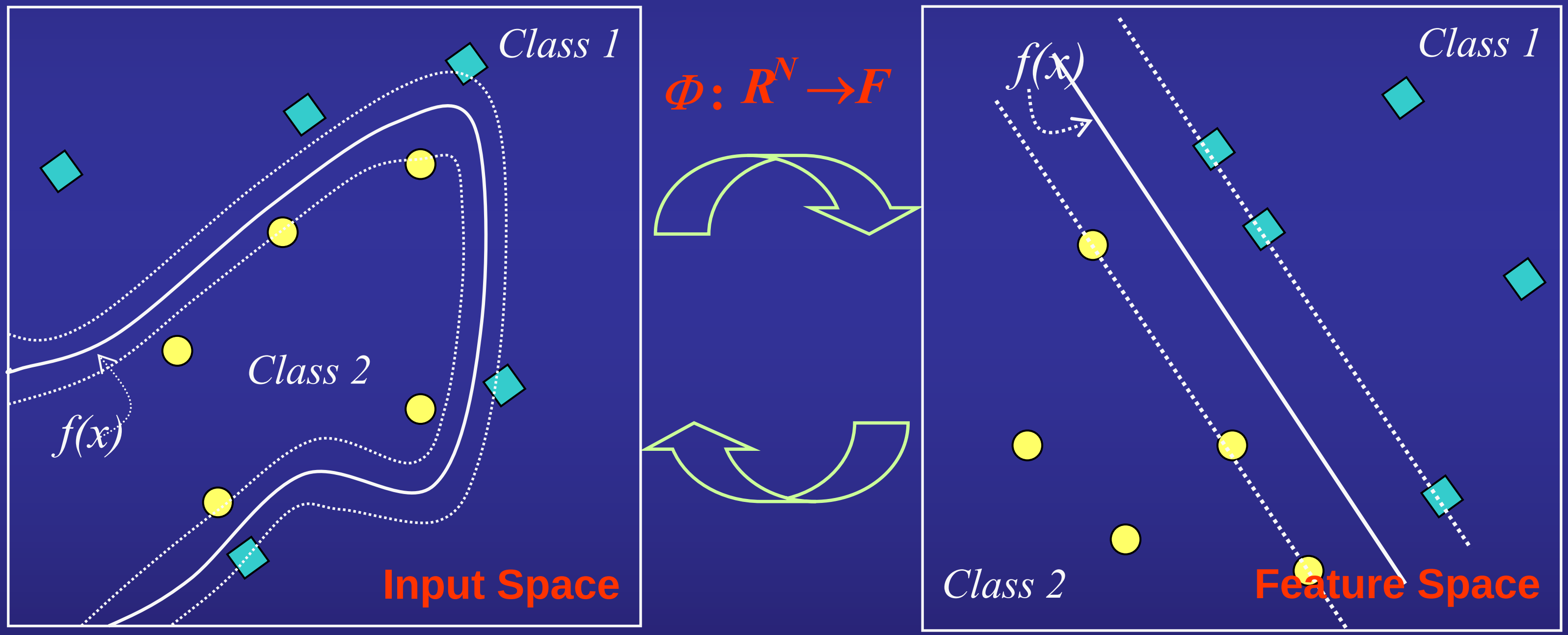
Nonlinear Case ?



Margin *Convexity* *Duality* **[Kernel]** *Sparseness*

Nonlinear Mapping from Input space(R^N) to Feature Space(F)

(EX) $\Phi : R^2 \rightarrow R^3$, $(x_1, x_2) \mapsto (z_1, z_2, z_3)$



NonLinear Algorithm in Input Space

Linear Algorithm in Feature Space

Feature Space

SVMs map the training data nonlinearly into a higher-dimensional feature space via ϕ and construct a separating hyperplane with maximum margin there.

This yields a nonlinear decision boundary in input space.

Linear Case

Problem

$$\max \quad L_D \equiv \sum_{i}^{l} \alpha_i - \frac{1}{2}\sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x_i} \cdot \mathbf{x}_j$$

$$s.t. \quad \alpha_i \geqslant 0, \quad \sum_{i}^{l} \alpha_i \, y_i = 0, \quad \forall i$$

Decision Function

$$f(x) = sign(w \cdot x + b) = sign\left(\sum_{i=1}^{l} \alpha_i y_i x_i \cdot x + b\right)$$

From Input Space to Feature Space

Nonlinear Case

Problem

$$\max \quad L_D \equiv \sum_{i}^{l} \alpha_i - \frac{1}{2}\sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j \phi(\mathbf{x_i}) \cdot \phi(\mathbf{x}_j)$$

$$s.t. \quad \alpha_i \geqslant 0, \quad \sum_{i}^{l} \alpha_i \, y_i = 0, \quad \forall i$$

Decision Function

$$f(x) = sign(w \cdot \phi(x) + b) = sign\left(\sum_{i=1}^{l} \alpha_i y_i \phi(x_i) \cdot \phi(x) + b\right)$$

Mapping Function (Φ) ?

However,

Mapping function is not unique.

difficult to find !

Feature Space could be (possibly) infinite dimensional.

Computation Demanding

Mapping Function (Φ) ?

How can we know the mapping function ?

How can we to handle the infinite dimensionality?

Use Kernel Functions !

SVM depends only on Dot Products between patterns.

Problem

$$\max \quad L_D \equiv \sum_{i}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j \phi(\mathbf{x_i}) \cdot \phi(\mathbf{x}_j)$$

$$s.t. \quad \alpha_i \geqslant 0, \quad \sum_{i}^{l} \alpha_i \, y_i = 0, \quad \forall i$$

Decision Function

$$f(x) = sign(w \cdot \phi(x) + b) = sign(\sum_{i=1}^{l} \alpha_i y_i \phi(x_i) \cdot \phi(x) + b)$$

By the use of a kernel function, it is possible to compute the dot product in input space without explicitly carrying out the map into the feature space

Margin *Convexity* *Duality* **[Kernel]** *Sparseness*

Kernel Function: $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y})$

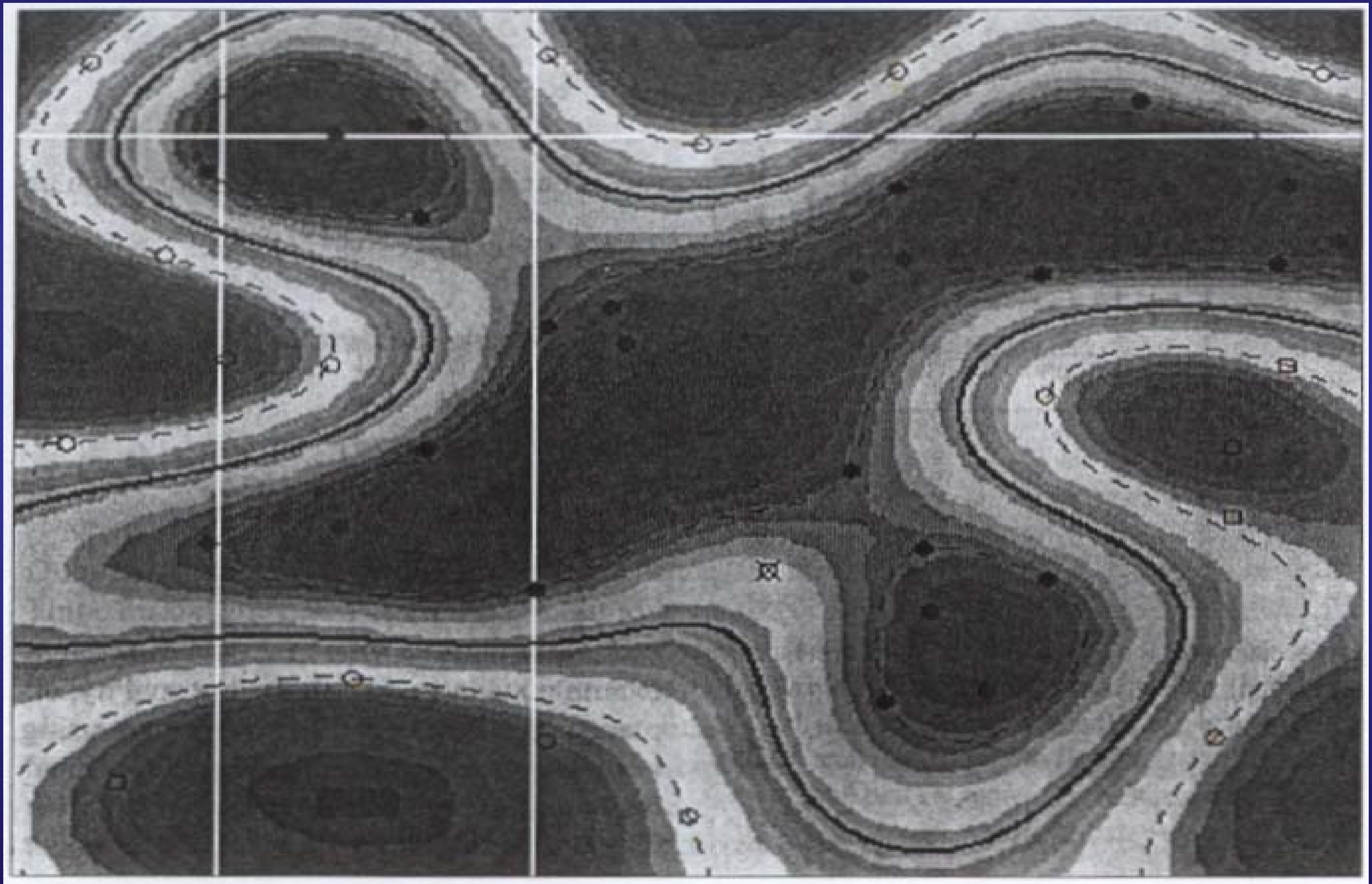
Functions Satisfying Mercers's Theorem

Polynomial kernels $k(\mathbf{x},\mathbf{y}) = (\mathbf{x} \cdot \mathbf{y})^P$

Radial Basis kernels $k(\mathbf{x}, \mathbf{y}) = \exp\left(\frac{-\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}\right)$

Sigmoid Kernels (3-MLP NN) $k(\mathbf{x}, \mathbf{y}) = \tanh(\kappa(\mathbf{x} \bullet \mathbf{y}) + \Theta)$

Nonlinear & Nonseparable Case

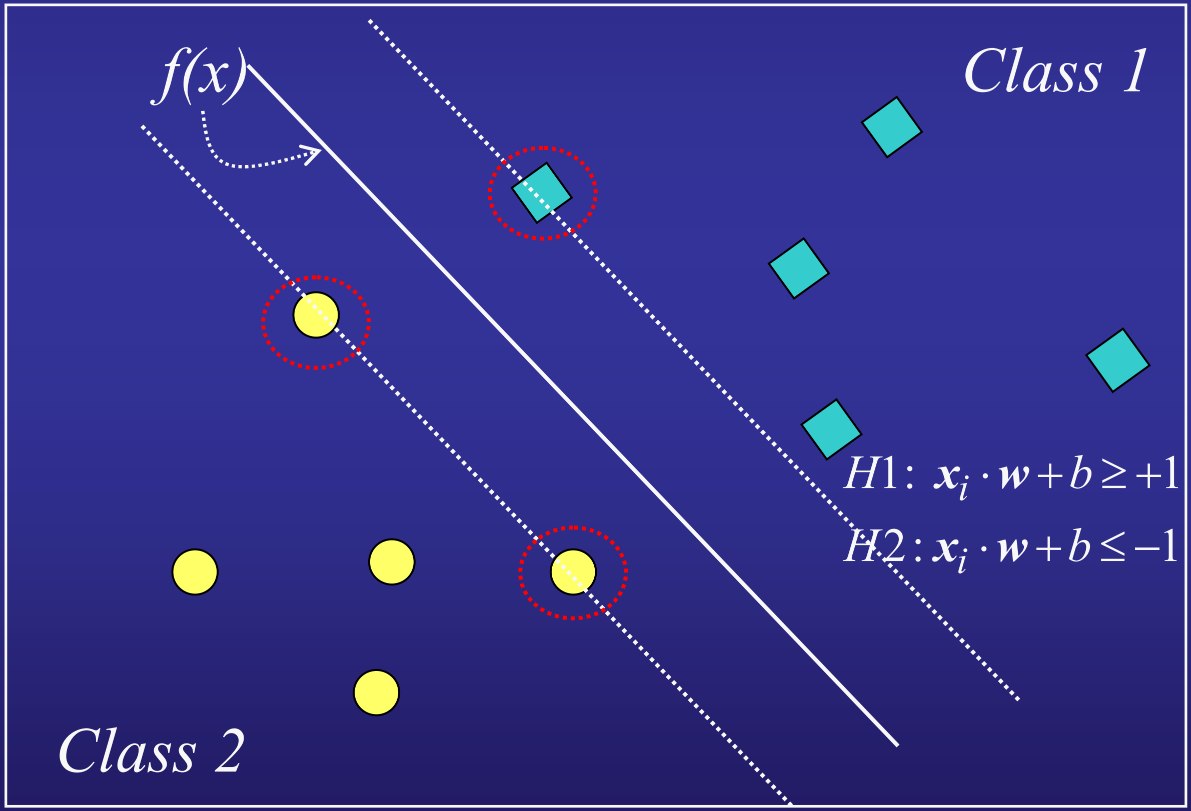


Only the points nearest to the hyperplane have positive weight !

They are called Support Vectors !

Remind the Complementarity Conditions

$$\alpha_i(y_i(\mathbf{x_i} \cdot \mathbf{w} + b) - 1) = 0 \quad \forall \mathrm{i} \quad \text{also note that } \boldsymbol{w} = \sum_i^l \alpha_i y_i \boldsymbol{x_i}$$



$$H1: \ \boldsymbol{x}_i \cdot \boldsymbol{w} + b \geq +1 \quad \text{for } \mathrm{y_i} = +1$$

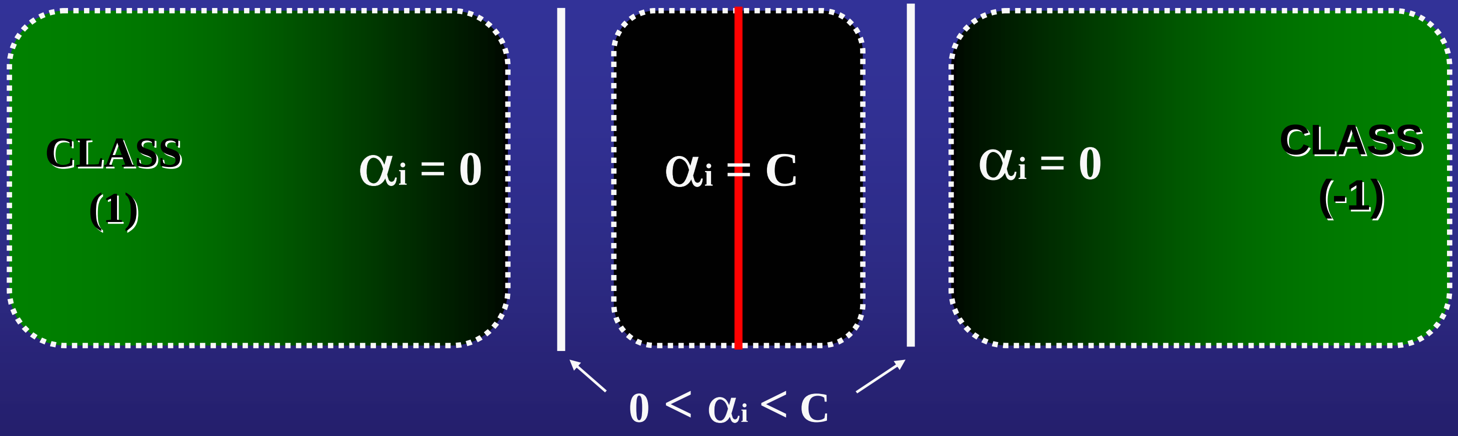
$$H2: \boldsymbol{x}_i \cdot \boldsymbol{w} + b \leq -1 \quad \text{for } \mathrm{y_i} = -1$$

SVs are distributed around decision boundary !

(1) Patterns OUT OF THE MARGIN $\alpha_i = 0$

(2) Patterns ON THE MARGIN (SVs) $0 < \alpha_i < C$

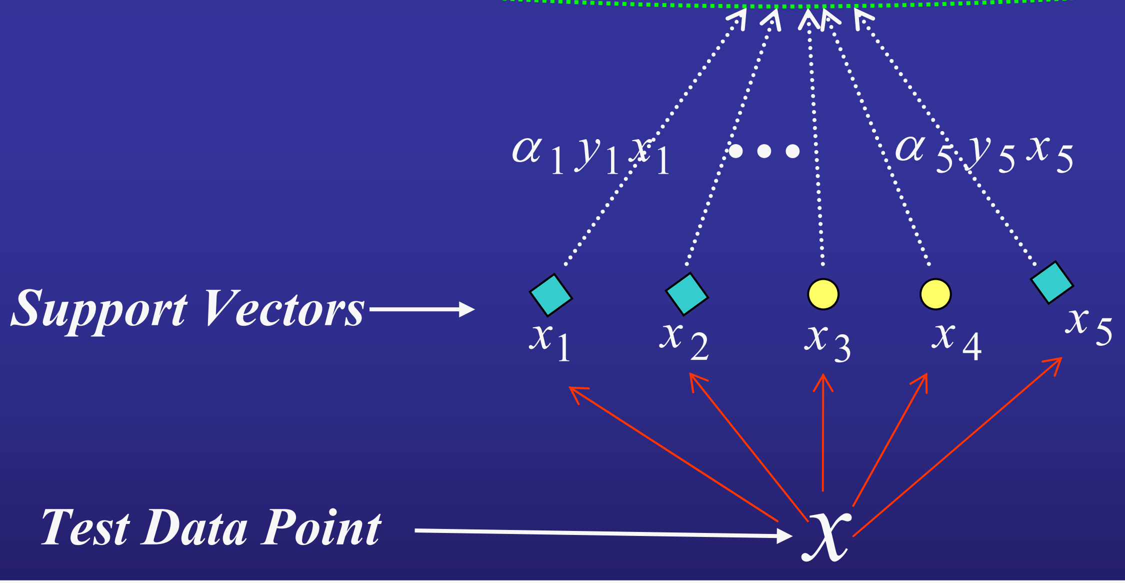
(3) Patterns BETWEEN THE MARGINS (SVs) $\alpha_i = C$



SVM Decision Function

$$f(x) = \begin{cases} +1 \ (\text{class } 1) & \textit{if } f(x) \geq 0 \\ -1 \ (\text{class } 2) & \textit{if } f(x) < 0 \end{cases}$$

$$f(x) = sign\ (w \cdot x + b) = sign\ \left(\sum_{i=1}^{l} \alpha_i y_i x_i \cdot x + b \right)$$



where $\boldsymbol{w} = \sum_{i}^{l} \alpha_i y_i \boldsymbol{x_i}$

Wrap-up

SVM QP Problem:
(Non-linear & Non-Separable)

$$\min . \ \frac{1}{2} \| \vec{w} \|^2 + C \sum_{i=1}^{M} \xi_i$$

$$s.t. \quad y_i (\vec{w} \cdot \Phi (\vec{x}_i) + b) \geqslant 1 - \xi_i \ ,$$

$$i = 1, \dots , M$$

SVM Decision Function: $f(\vec{x}) = sign \left(\sum_{i \in SV} y_i \alpha_i \Phi(\vec{x}_i) \cdot \Phi(\vec{x}) + b \right)$

Kernels:

$$\Phi(\vec{x}) \cdot \Phi(\vec{x}') = k(\vec{x}, \vec{x}') = \begin{cases} \exp \left(- \| \vec{x} - \vec{x}' \|^2 / 2\sigma^2 \right) \\ \tanh(\kappa (\vec{x} \cdot \vec{x}') + \Theta) \\ (\vec{x} \cdot \vec{x}' + 1)^P \end{cases}$$

Wrap-up

$$\min_{0\le\alpha_i\le C} W(\alpha_i,b) = \frac{1}{2}\sum_{i,j=1}^{M}\alpha_i\alpha_j y_i y_j K(\vec{x}_i,\vec{x}_j) - \sum_{i=1}^{M}\alpha_i + b\sum_{i=1}^{M} y_i\alpha_i$$

KKT

$$\frac{\partial W(\alpha_i,b)}{\partial\alpha_i} = \sum_{j=1}^{M} y_i y_j K(\vec{x}_i,\vec{x}_j)\alpha_j + y_i b - 1 = y_i \bar{f}(x_j) - 1$$

$$\frac{\partial W(\alpha_i,b)}{\partial b} = \sum_{j=1}^{M} y_j\alpha_j = 0$$

where $\bar{f}(\vec{x}) = \sum_{i=1}^{M} y_i\alpha_i K(\vec{x}_i,\vec{x}) + b$

Application I

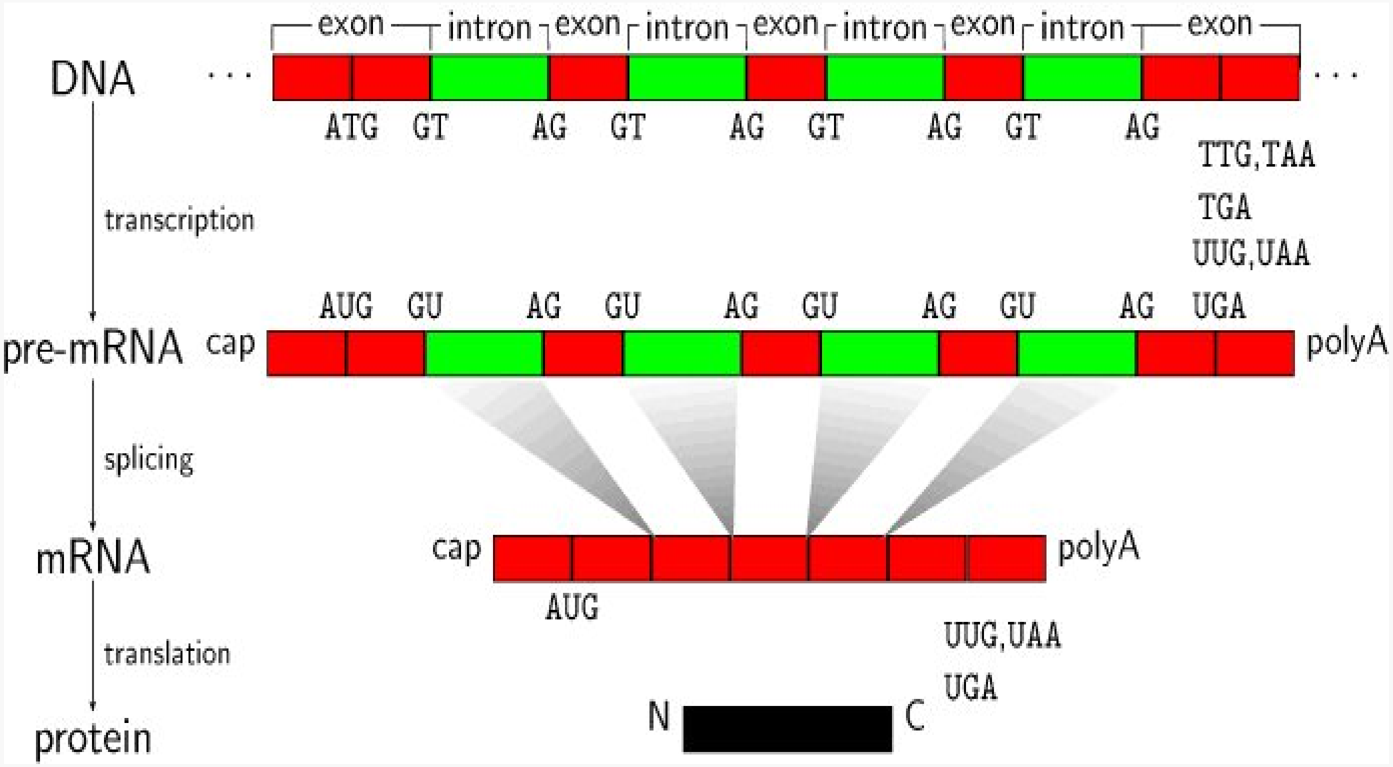
Recognition of Alternatively Spliced Exons in C.elegans

Task : Classification

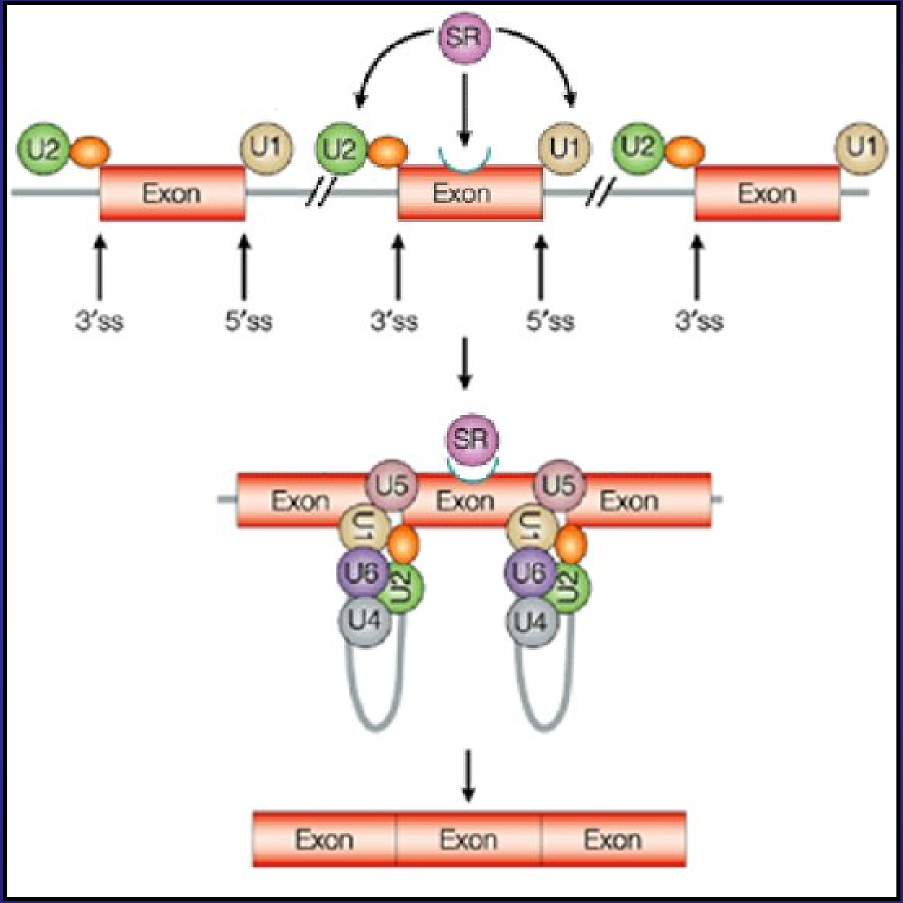
Model : Support Vector Machines

Application: C.elegans Genes - Alternative Splicing

Splicing


exon
intron
exon
intron
exon
intron
exon
intron
exon
DNA
ATG
GT
AG
GT
AG
GT
AG
GT
AG
TTG,TAA
TGA
transcription
UUG,UAA
AUG
GU
AG
GU
AG
GU
AG
GU
AG
UGA
pre-mRNA
cap
polyA
splicing
mRNA
cap
polyA
AUG
translation
UUG,UAA
UGA
protein
N
C

Splicing



Splice sites are

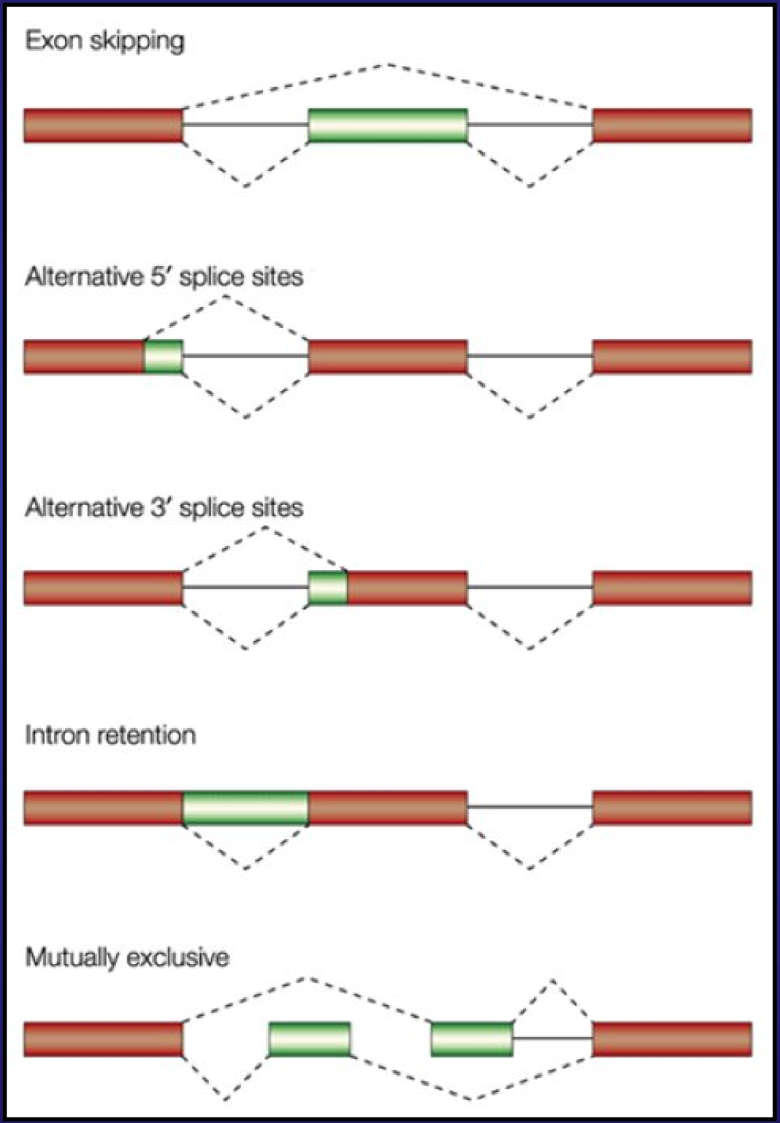
- *the exon/intron boundaries*
- *recognized by five snRNAs*
- *assembled in snRNPs*
- *flanked by regulatory elements*

Spliceosomal Proteins

- *interact with snRNPs and mRNA*
- *regulate recognition of splice sites*
- *can lead to alternative transcripts*

One gene may correspond to several transcripts/proteins !!

Alternative Splicing



Alternative Splicing

Alternative Splicing (AS) ..

- can produce several mRNA transcript per gene

(sometimes leading to more than 100 slightly different proteins)

- greatly increases the proteome diversity in eukaryotes

(about 70% of human genes are alternatively spliced!)

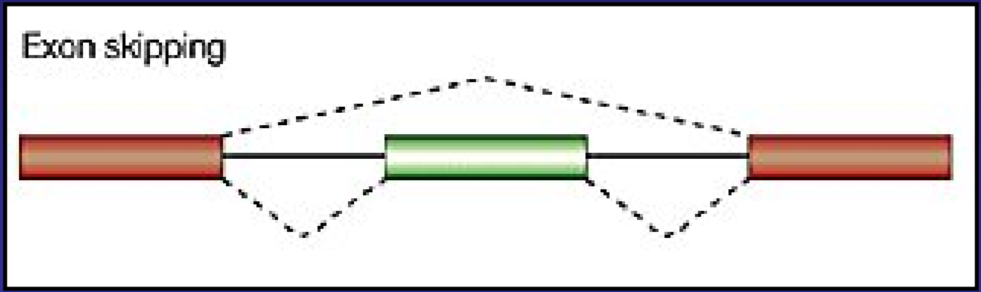
Alternative Splicing

Methods for identifying alternative splicing …

- usually need many EST sequences or
- exploit conservation between several organisms

Novel AS prediction method only using the pre-mRNA

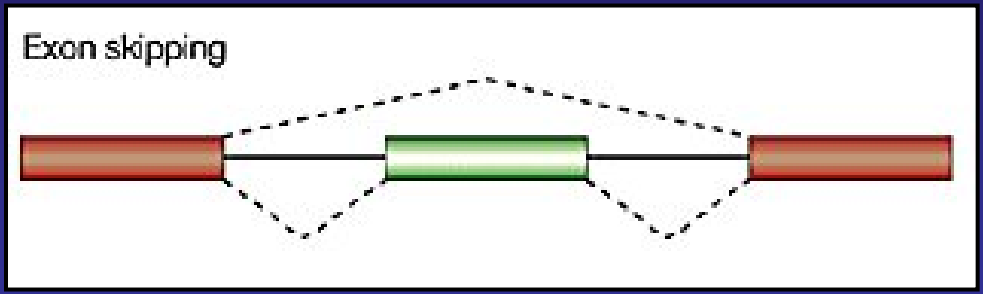
Alternatively Spliced Exons



Idea: Use Machine Learning to

- understand differences between alternative and constitutive splicing
- exploit and identify regulative elements
- predict unknown alternative splicing events

Alternatively Spliced Exons



Previous work

Analysis of conserved alternatively spliced exons (Sorek et al., Yeo et al. and others)

- *consider conserved alternative spliced exons (ACE)*
- *exploit that ACE and flanking introns are more conserved between mouse and human*

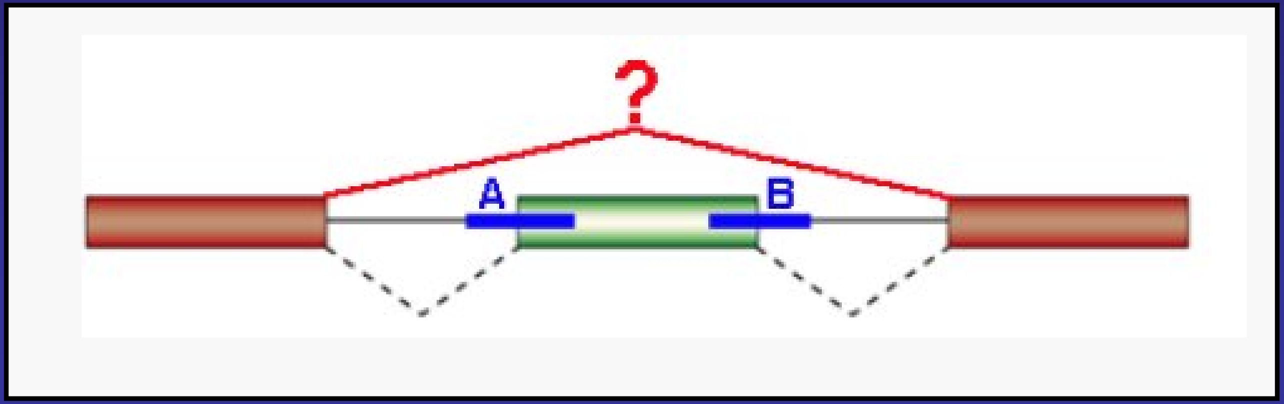
Problem

only works for conserved exons

Derive the features from the "pre-mRNA" in order to find "novel" exons !!

Task Formulation

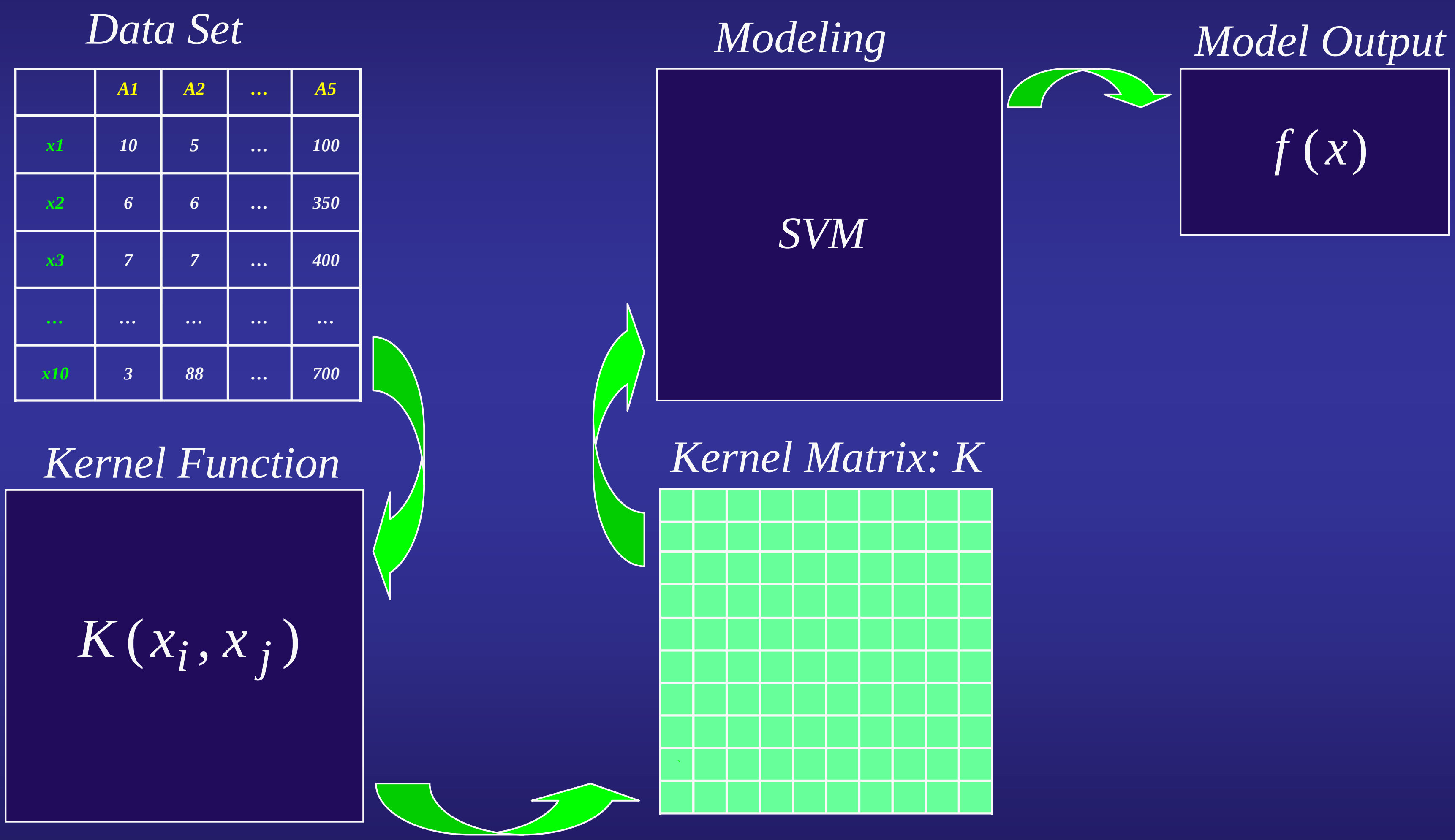
Two-class Classification Problem



A (or B) is true splice site or not?

Use Support Vector Machines!

Remind the Procedure of Kernel Methods !



Data Set

	A1	A2	…	A5
x1	10	5	…	100
x2	6	6	…	350
x3	7	7	…	400
…	…	…	…	…
x10	3	88	…	700

Kernel Function

$K(x_i, x_j)$

Kernel Matrix: K

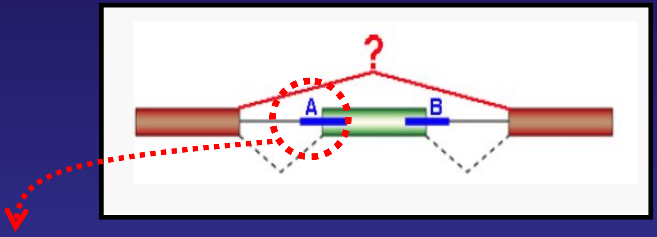
Modeling

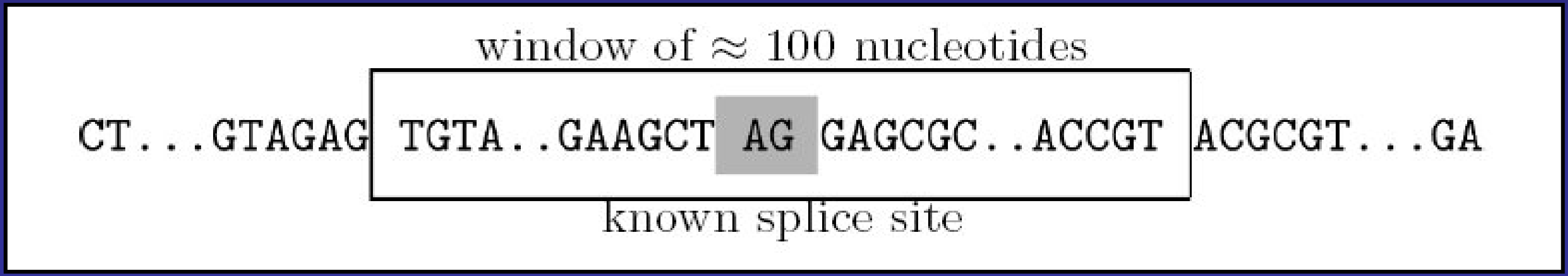
SVM

Model Output

$f(x)$

Procedure - Data Set



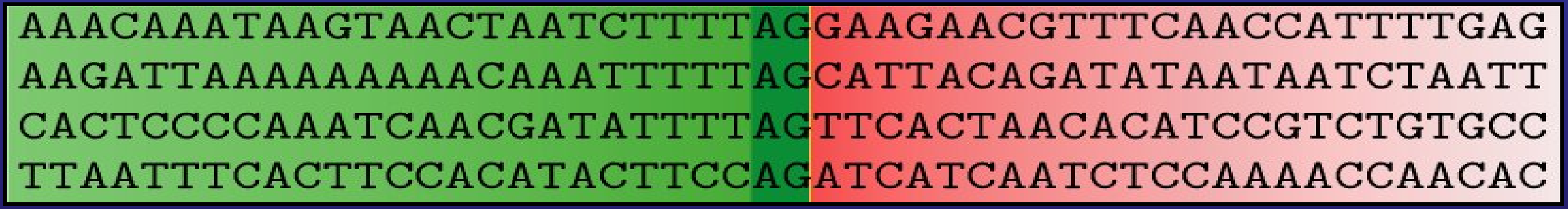

window of ≈ 100 nucleotides

```
CT...GTAGAG TGTA..GAAGCT AG GAGCGC..ACCGT ACGCGT...GA
```

known splice site

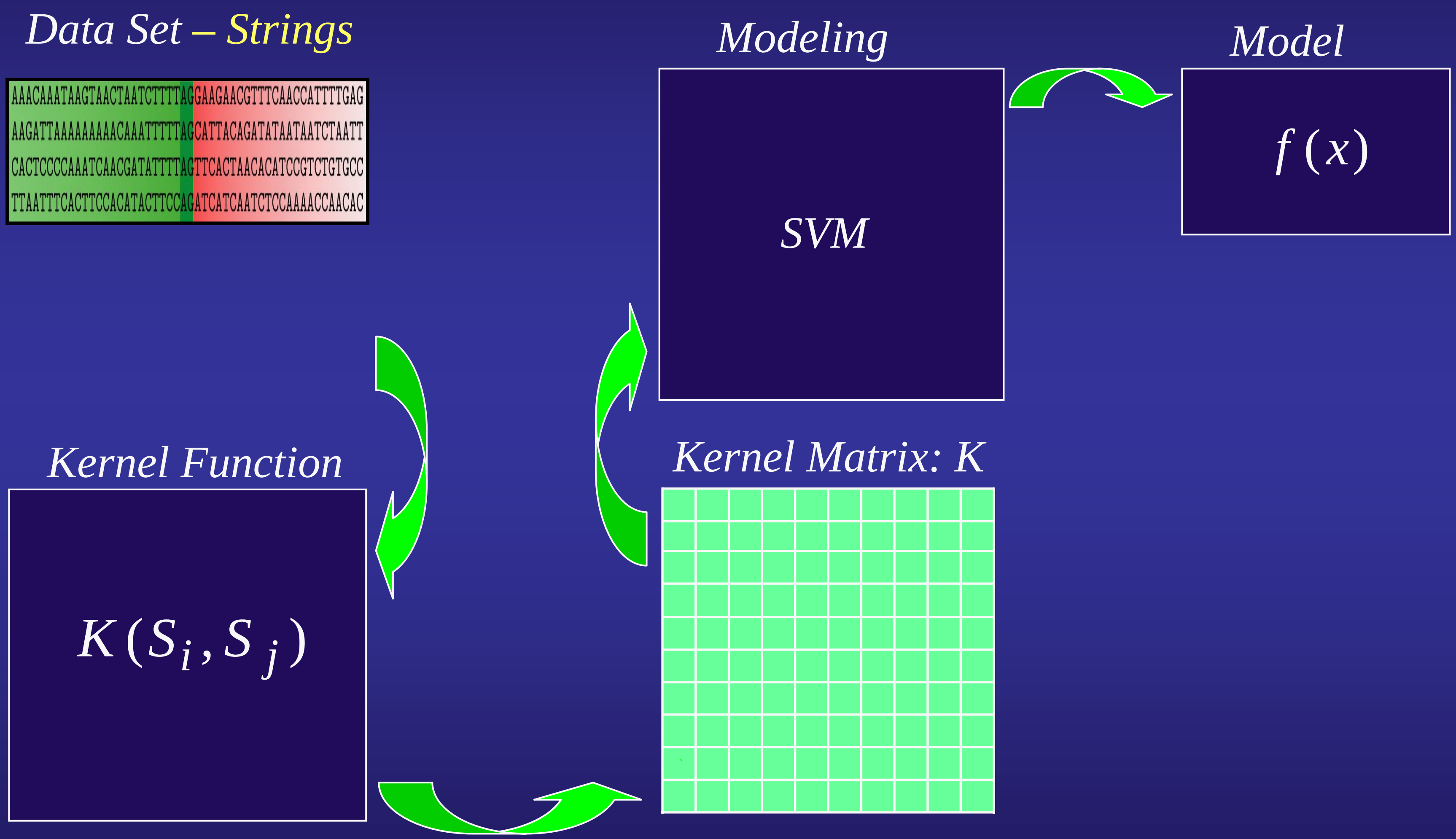
True sites (y=1): fixed window around a true splice site

Decoys sites (y=-1): generated by shifting the window



```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
```

Procedure - Data Set



Data Set – Strings

AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC

Kernel Function

$K(S_i, S_j)$

Kernel Matrix: K

Modeling

SVM

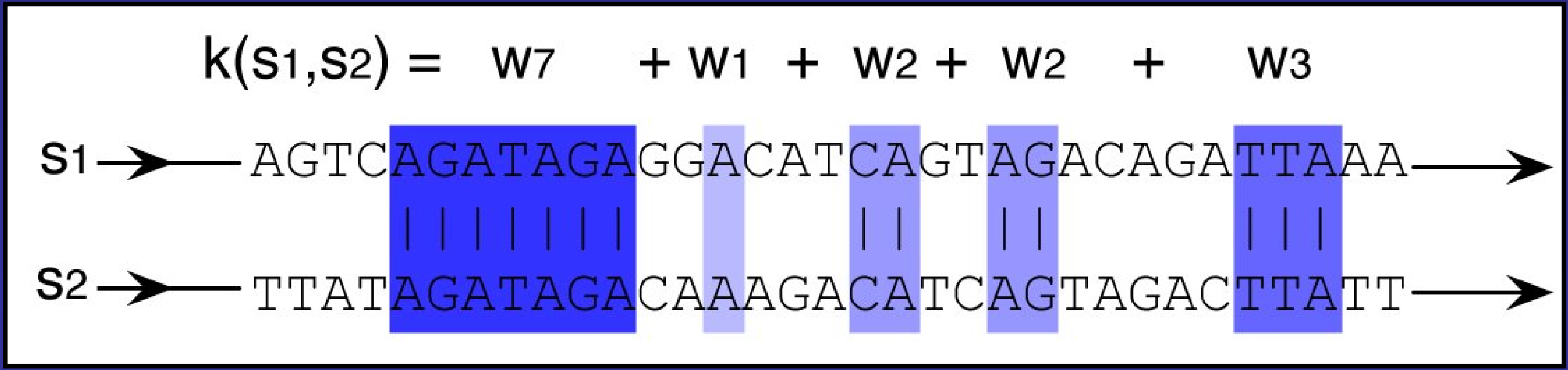
Model

$f(x)$

Procedure - Kernel Function (Matrix)

Kernels measure similarities between sequences

Weighted Degree Kernel (Sonnenburg et al., 2002)

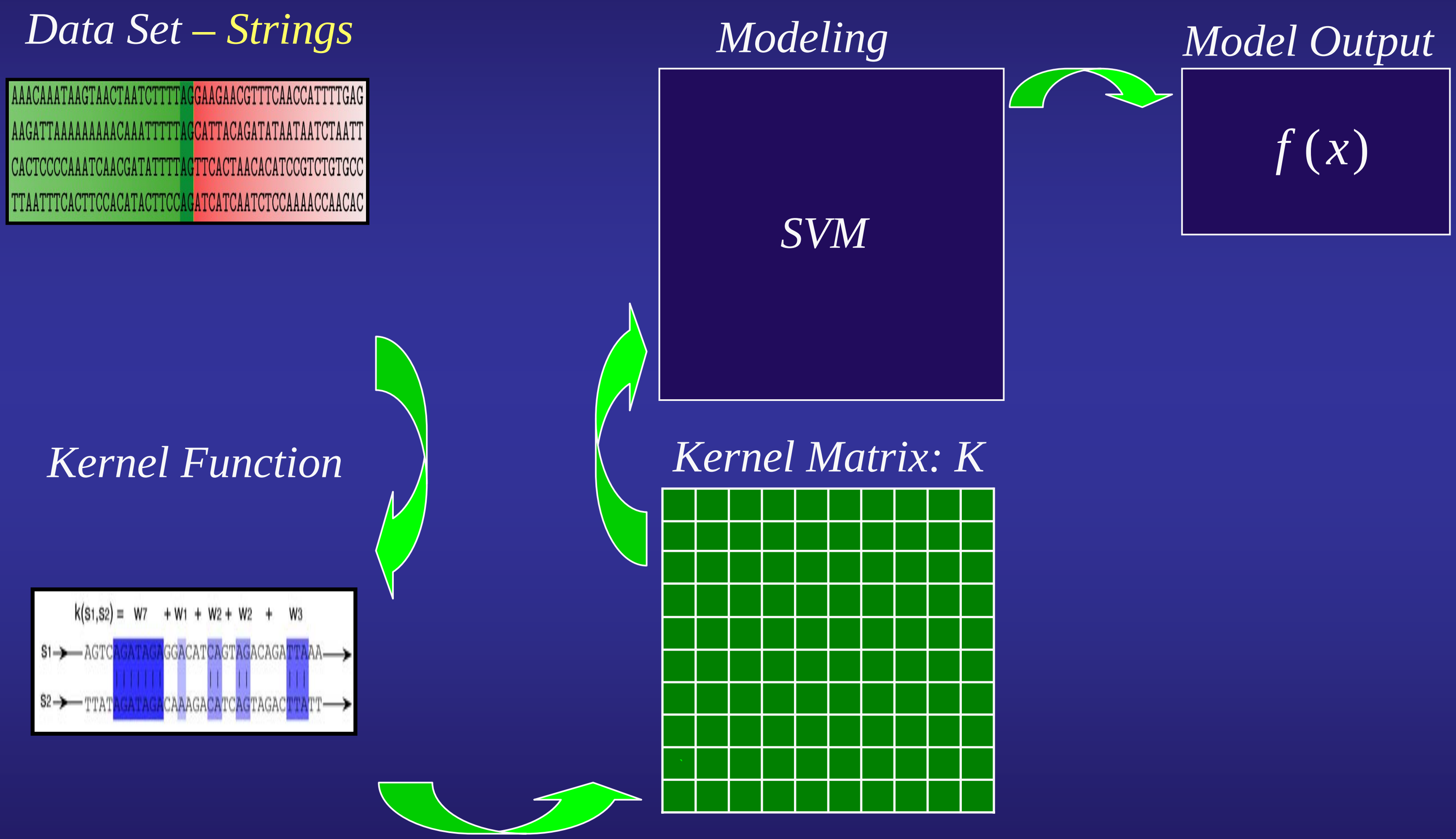


k(s1,s2) = W7 + W1 + W2 + W2 + W3
S1
AGTCAGATAGAGGACATCAGTAGACAGATTAAA
S2
TTATAGATAGACAAAGACATCAGTAGACTTATT

Given two sequences S_1 and S_2 of equal length, the kernel consists of a weighted sum to which each match in the sequences makes a contribution.

The longer matches contribute more significantly.

Procedure - Kernel Function (Matrix)



Data Set – Strings

AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC

Kernel Function

Modeling

SVM

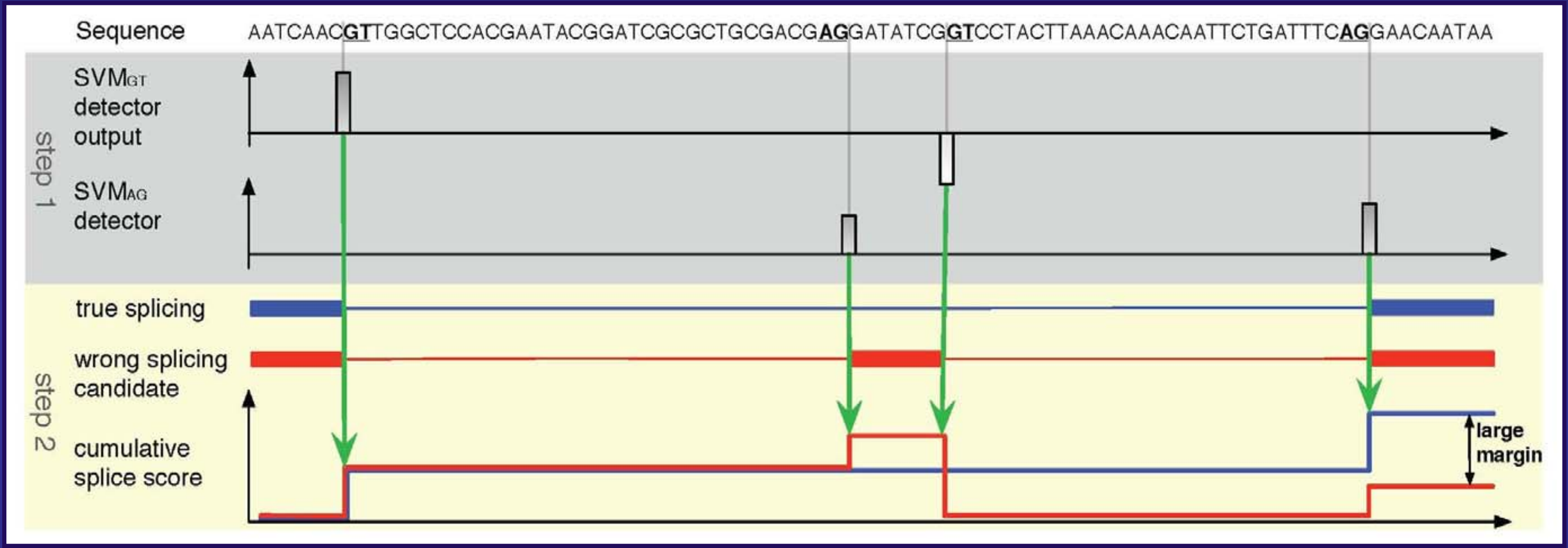
Model Output

$f(x)$

Kernel Matrix: K

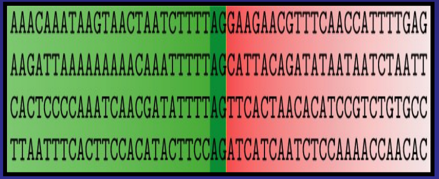
k(S1,S2) = W7 + W1 + W2 + W2 + W3
S1
AGTCAGATAGAGGACATCAGTAGACAGATTAAA
S2
TTATAGATAGACAAAGACATCAGTAGACTTATT

Procedure - Modeling & Output



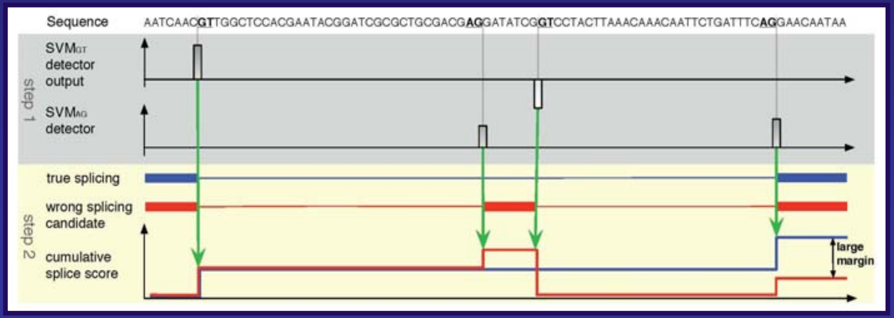
Procedure - *Modeling & Output*

Data Set – Strings

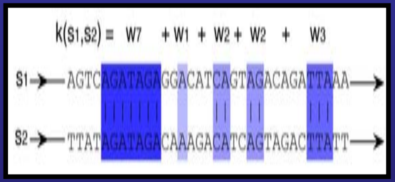


AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC

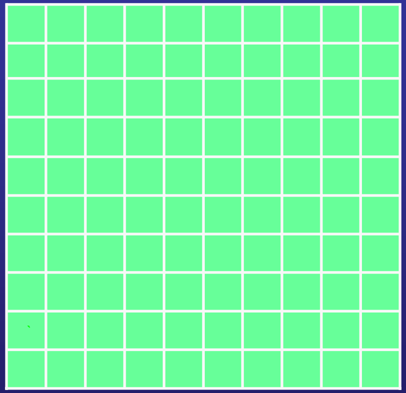
Modeling -- SVM Model



Kernel Function

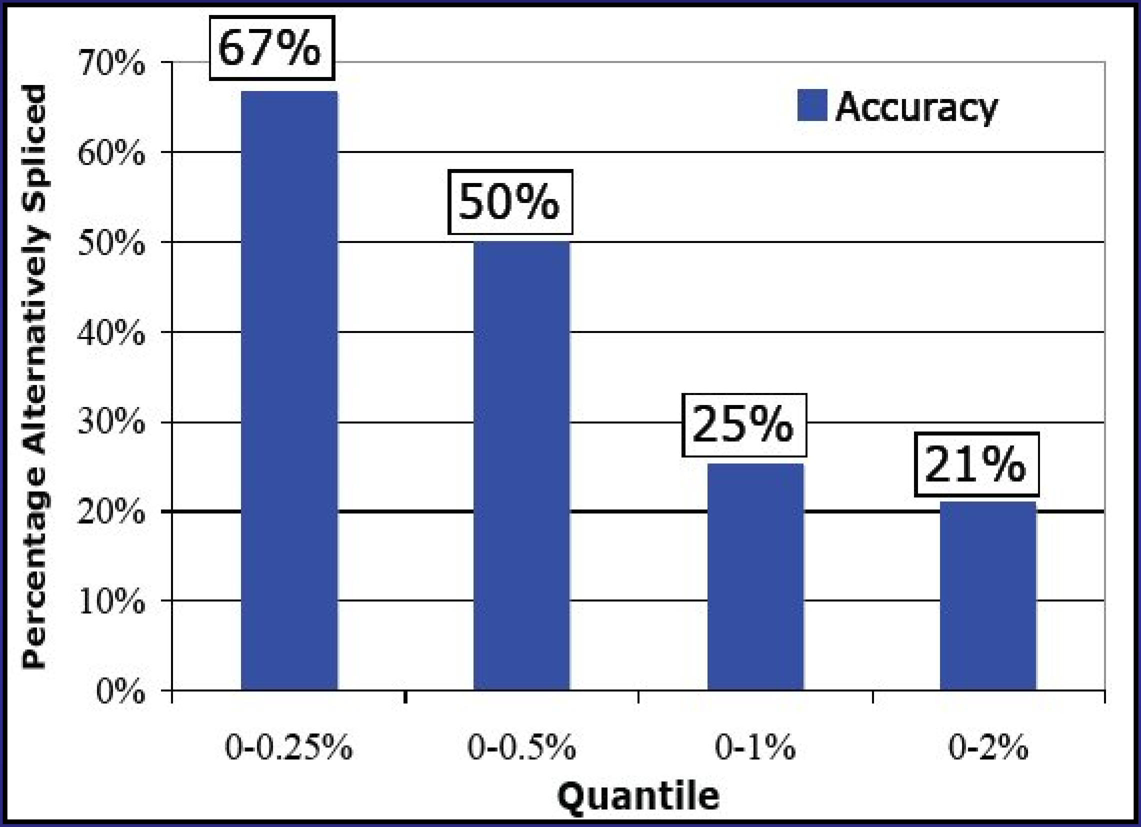

k(S1,S2) = W7 + W1 + W2 + W2 + W3
S1
AGTCAGATAGAGGACATCAGTAGACAGATTAAA
S2
TTATAGATAGACAAAGACATCAGTAGACTTATT

Kernel Matrix: K



Results

Exons Known



- 21,000 exons and 28,000 introns (single EST confirmed)

Results

280 AS spliced exons (total)

- *~ 1% of known exons are alternatively spliced (AS)*
- *~ 0.25% of AS exons are yet completely unknown*

RT-PCR with primers in flanking exons

(25 random exons & introns from 1-2% top ranks)

- *13 confirmed by RT-PCR*

Additional 80 AS exons can be found with less than 200 additional RT-PCRs

The Most Up-To-Date Models

Kernel Methods

Support Vector Machines (SVM), kPCA, kCCA, kICA, etc

Semi-Supervised Learning Methods

Graph-based SSL, Transductive Inference Methods

** Note that, currently, the term of "Semi-Supervised Learning" has been used as a name of "model" as well as the concept of "learning scheme"*

Semi-Supervised Learning Methods: Graph-Based SSL

Semi-Supervised Learning

Semi-Supervised Learning utilizes every possible information in hand (known + unknown), therefore enhances prediction accuracy of a model

Supervised

Known

Unknown

	A_1	A_2	A_3	…	A_{10}	y
x_1	10	5	red	…	1000	1
x_2	6	6	blue	…	3500	20
x_3	7	7	yellow	…	400	45
…	…	…	…	…	…	…
x_{18}	3	56	red	…	0	30
x_{19}	15	62	red	…	500	100
x_{20}	3	88	blue	…	700	3
x_{21}	5	42	red	…	560	?
…	…	…	…	…	…	?
x_{50}	25	56	blue	…	600	?

vs.

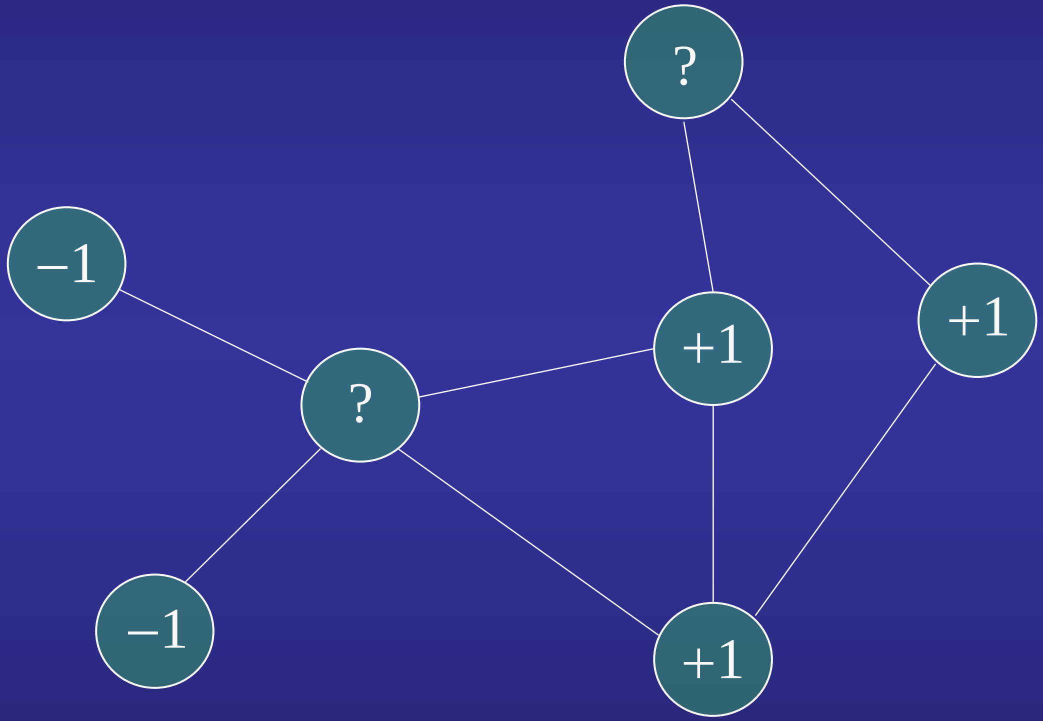
Semi-Supervised

Known

Unknown

	A_1	A_2	A_3	…	A_{10}	y
x_1	10	5	red	…	1000	1
x_2	6	6	blue	…	3500	20
x_3	7	7	yellow	…	400	45
…	…	…	…	…	…	…
x_{18}	3	56	red	…	0	30
x_{19}	15	62	red	…	500	100
x_{20}	3	88	blue	…	700	3
x_{21}	5	42	red	…	560	?
…	…	…	…	…	…	?
x_{50}	25	56	blue	…	600	?

Semi-Supervised Learning with a Single Graph

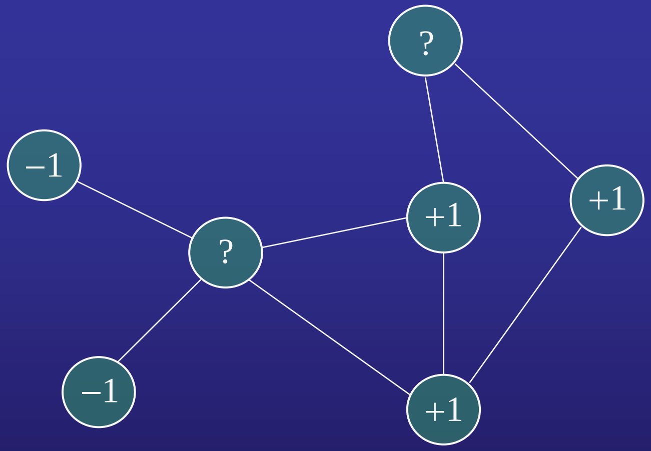


Semi-Supervised Learning with a Single Graph

- *Adjacency (similarity) matrix of the network: W*
- *Known Labels :* $y_1,\ldots,y_l \in \{-1, 1\}$
- *Unknown Labels :* $y_{l+1},\ldots,y_n \in \{0\}$
- *Predicted outputs :* $f_1,\ldots,f_n$

f_i *should be close to those of adjacent nodes,* f_j*'s where i~j.*

f_i *should be close to the given label* y_i *at training nodes*



Semi-Supervised Learning with a Single Graph

Learning Problem

$$\min \quad \mu \sum_{i \sim j} w_{ij}(f_i - f_j)^2 + \sum_i (f_i - y_i)^2$$

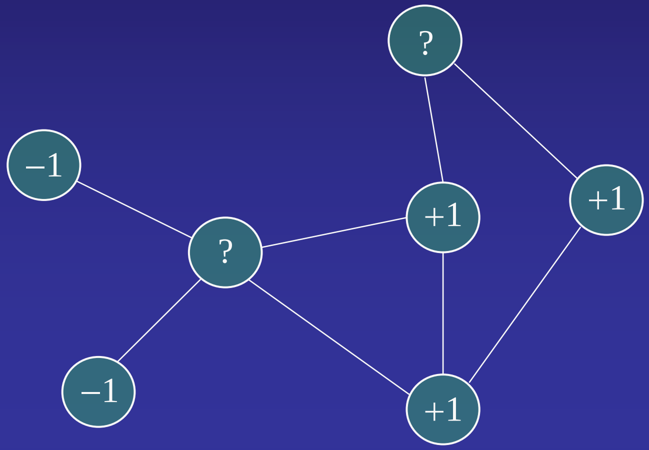
Equivalent Vector Form

$$\min_{\boldsymbol{f}} \quad \mu\, \boldsymbol{f}^T L\, \boldsymbol{f} + (\boldsymbol{f} - \boldsymbol{y})^T (\boldsymbol{f} - \boldsymbol{y})$$

L is called the graph Laplacian matrix where

$$L = D - W, \qquad D = diag(d_i), \quad d_i = \sum_j w_{ij}$$

Semi-Supervised Learning with a Single Graph



Objective Function

$$\min_{\boldsymbol{f}} \quad \mu\, \boldsymbol{f}^T L\, \boldsymbol{f} + (\boldsymbol{f} - \boldsymbol{y})^T (\boldsymbol{f} - \boldsymbol{y})$$

Solution

$$\boldsymbol{f} = \{\, \boldsymbol{I} + \mu\, L \,\}^{-1} \boldsymbol{y}$$

Application II

Functional Class Prediction with Multiple Networks

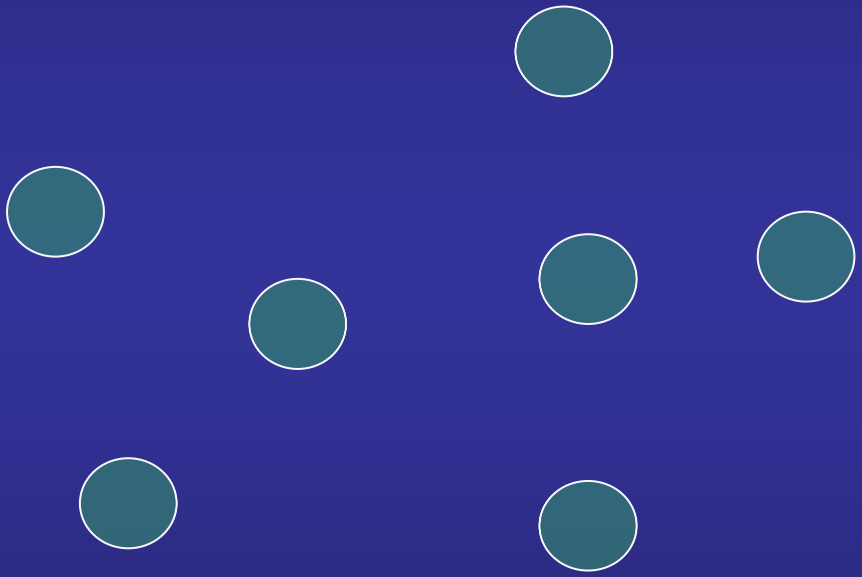
Task : Classification, Data Integration

Model : Semi-Supervised Learning

Application: Yeast Protein : Protein Function Prediction

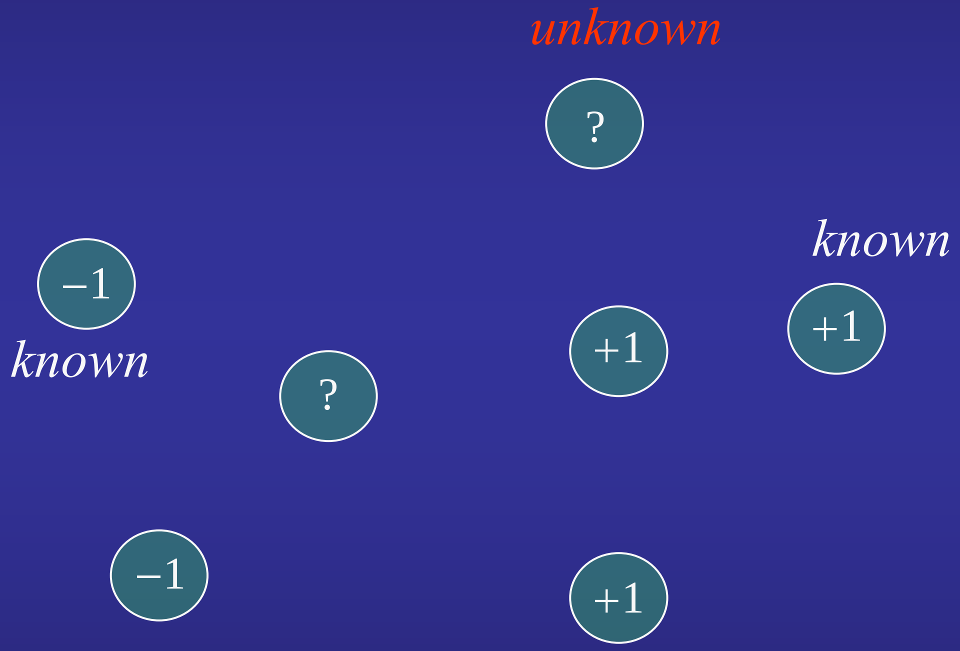
Functional Class Prediction on a Protein Network

Proteins : Nodes



Functional Class Prediction on a Protein Network

Functional Classes of Proteins : *Labeled / Unlabeled Nodes*

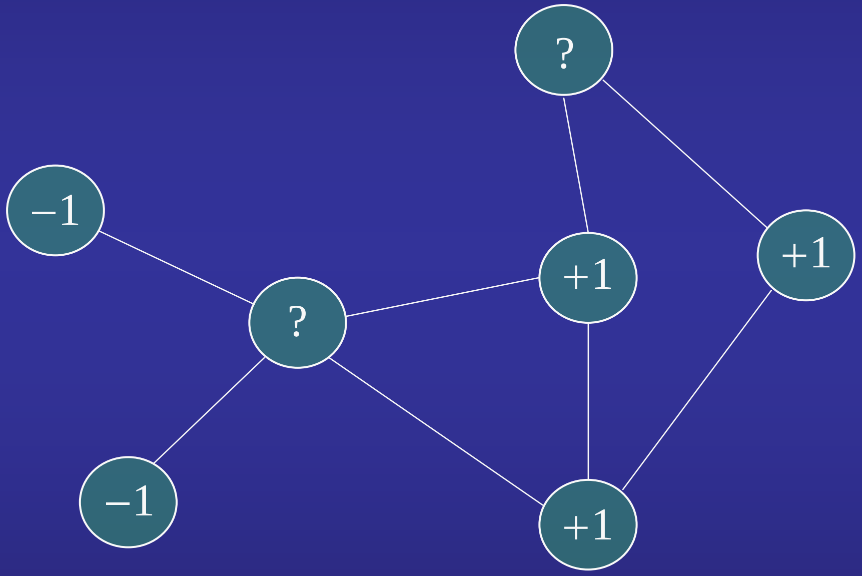


+1/-1 : Labeled proteins with/without a specific function

? : Unlabeled proteins

Functional Class Prediction on a Protein Network

Similarities between Proteins *: Edges*



- Edges in Physical Interaction Network: Two proteins physically interact (e.g., docking)
- Edges in Metabolic Network of Enzymes: Two enzymes catalyzing successive reactions

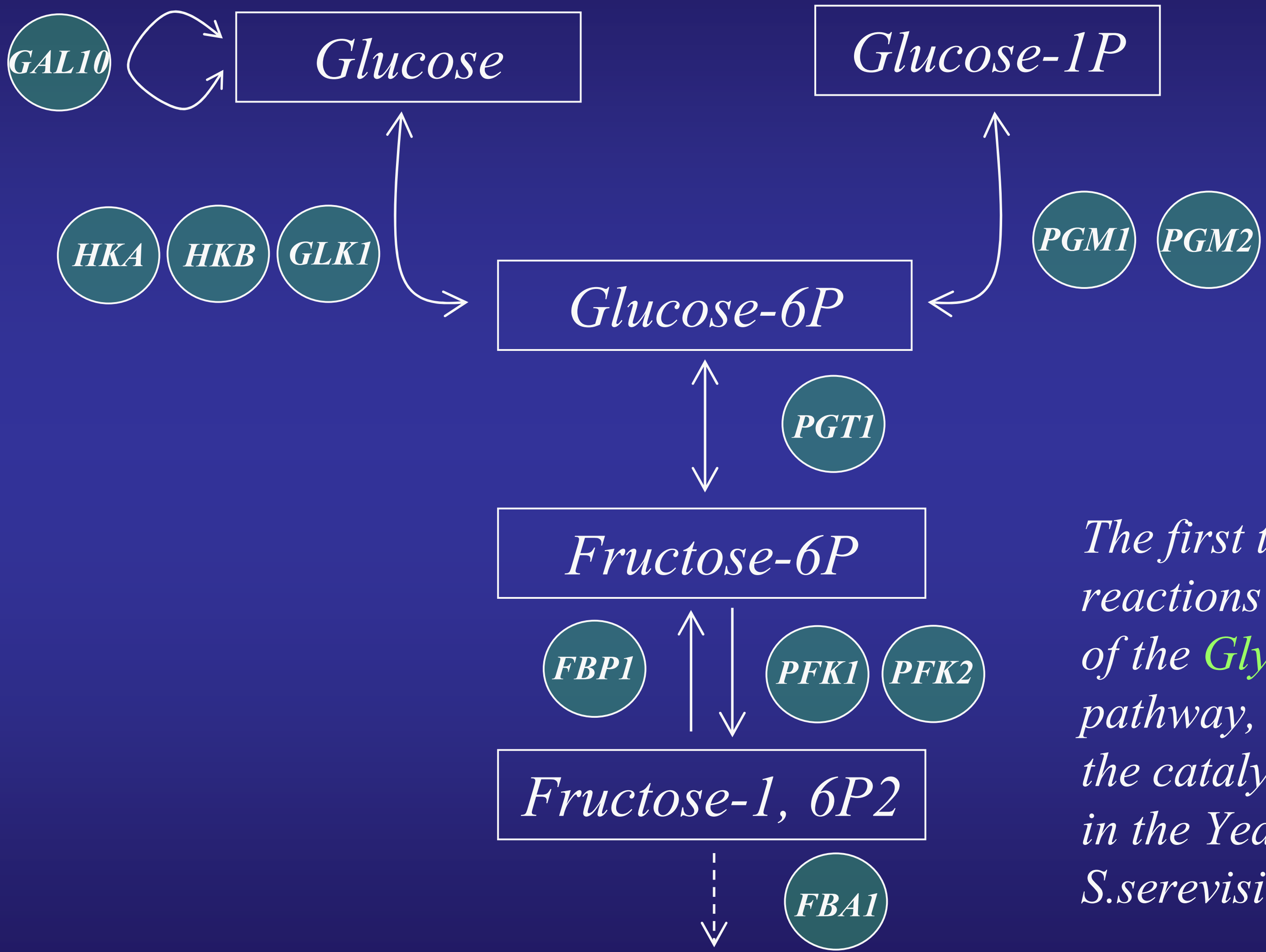
Functional Class Prediction on a Protein Network

The task is to predict labels of unlabeled proteins using similarities.

Graph Representation on Biological Networks

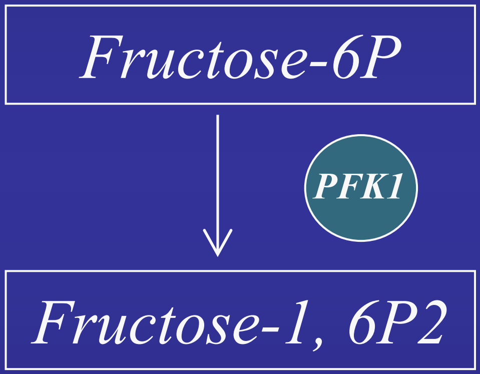
Example: Metabolic Gene Network

Graph Representation on Biological Networks



The first three reactions of the Glycolysis pathway, together with the catalyzing enzymes in the Yeast S.serevisiae.

Graph Representation on Biological Networks

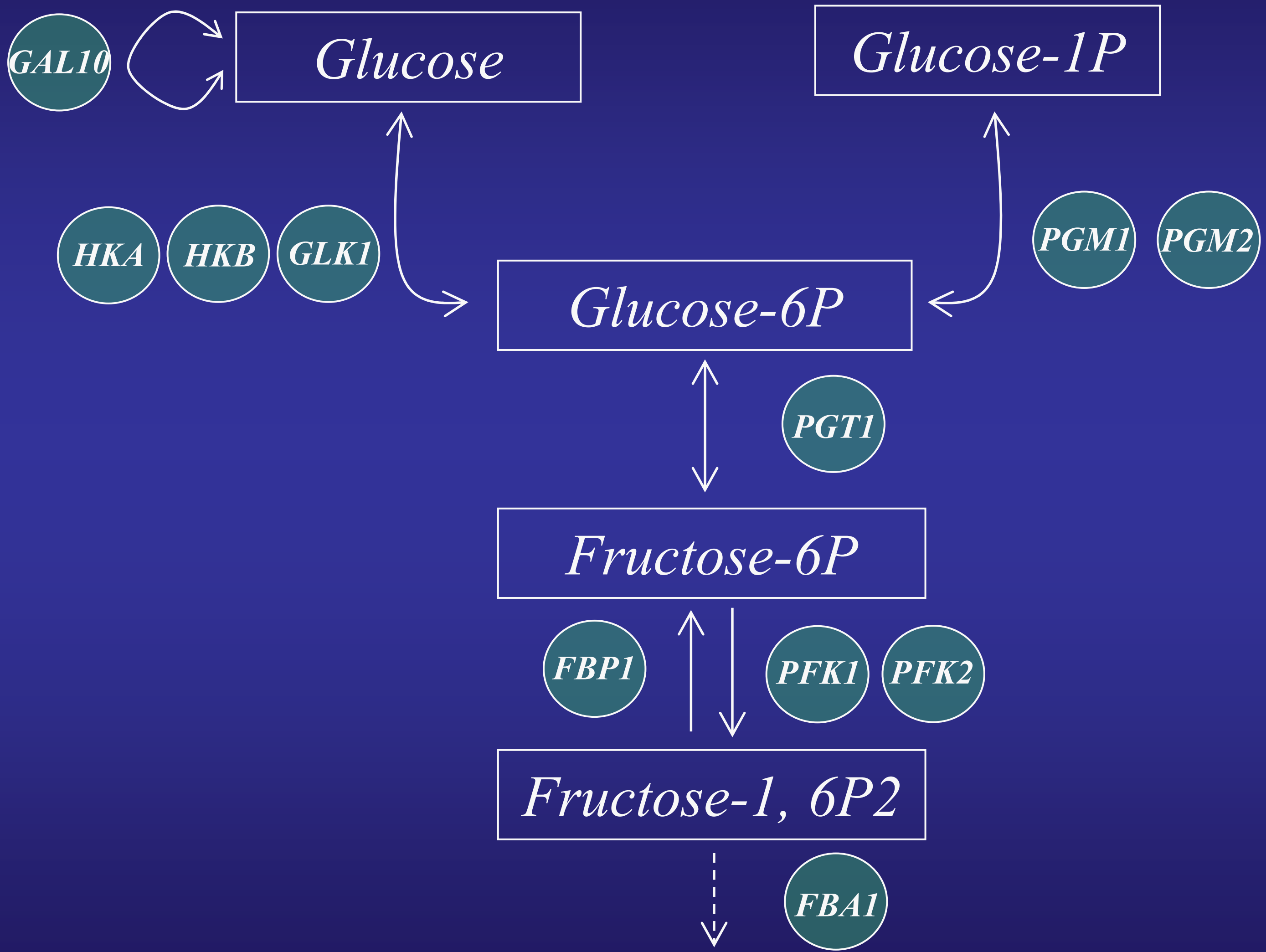


Substrate of PFK1

Enzyme or Protein, PFK1

Product of PFK1

Graph Representation on Biological Networks


Glucose

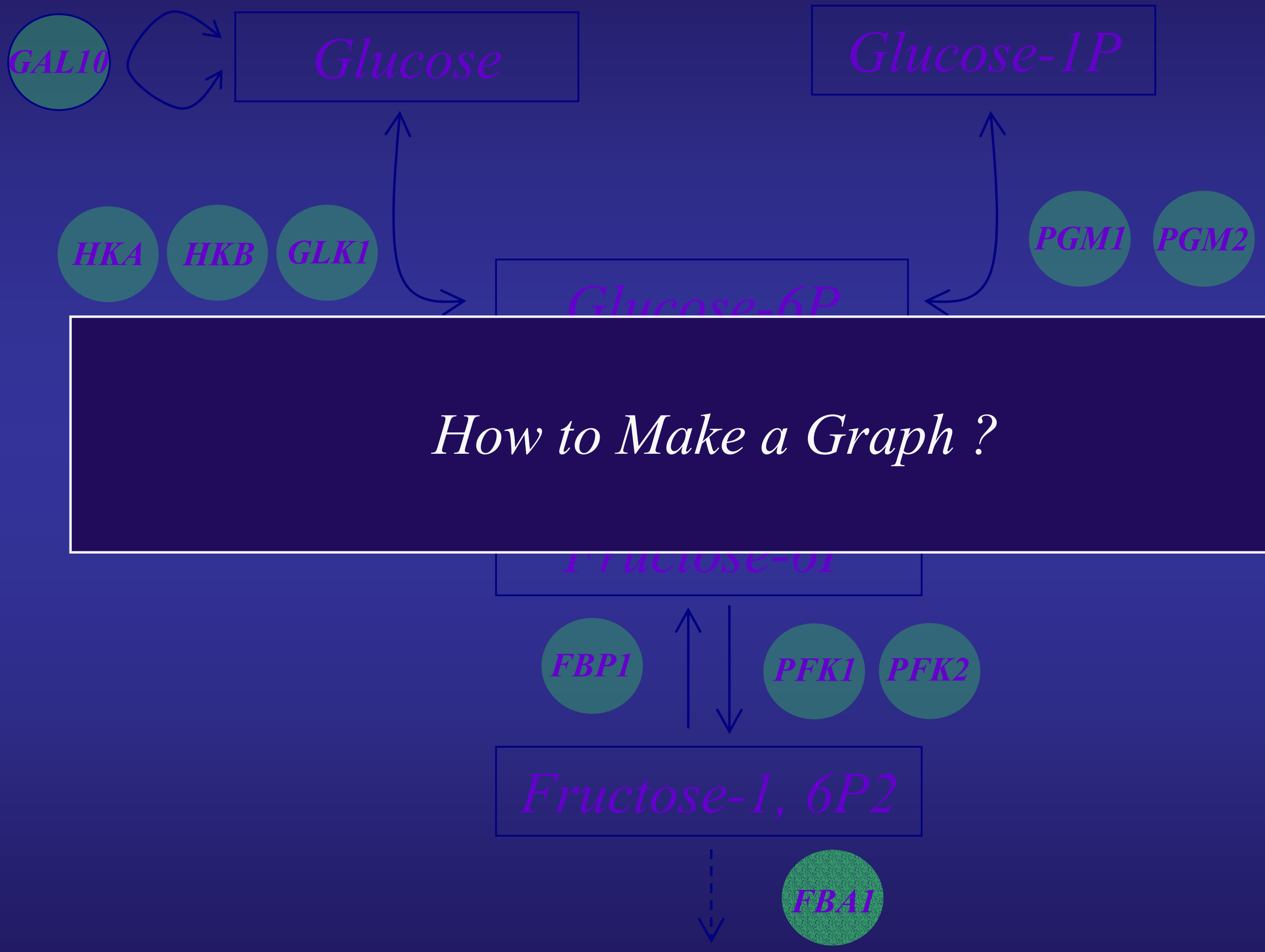
Glucose-1P

Glucose-6P

Fructose-6P

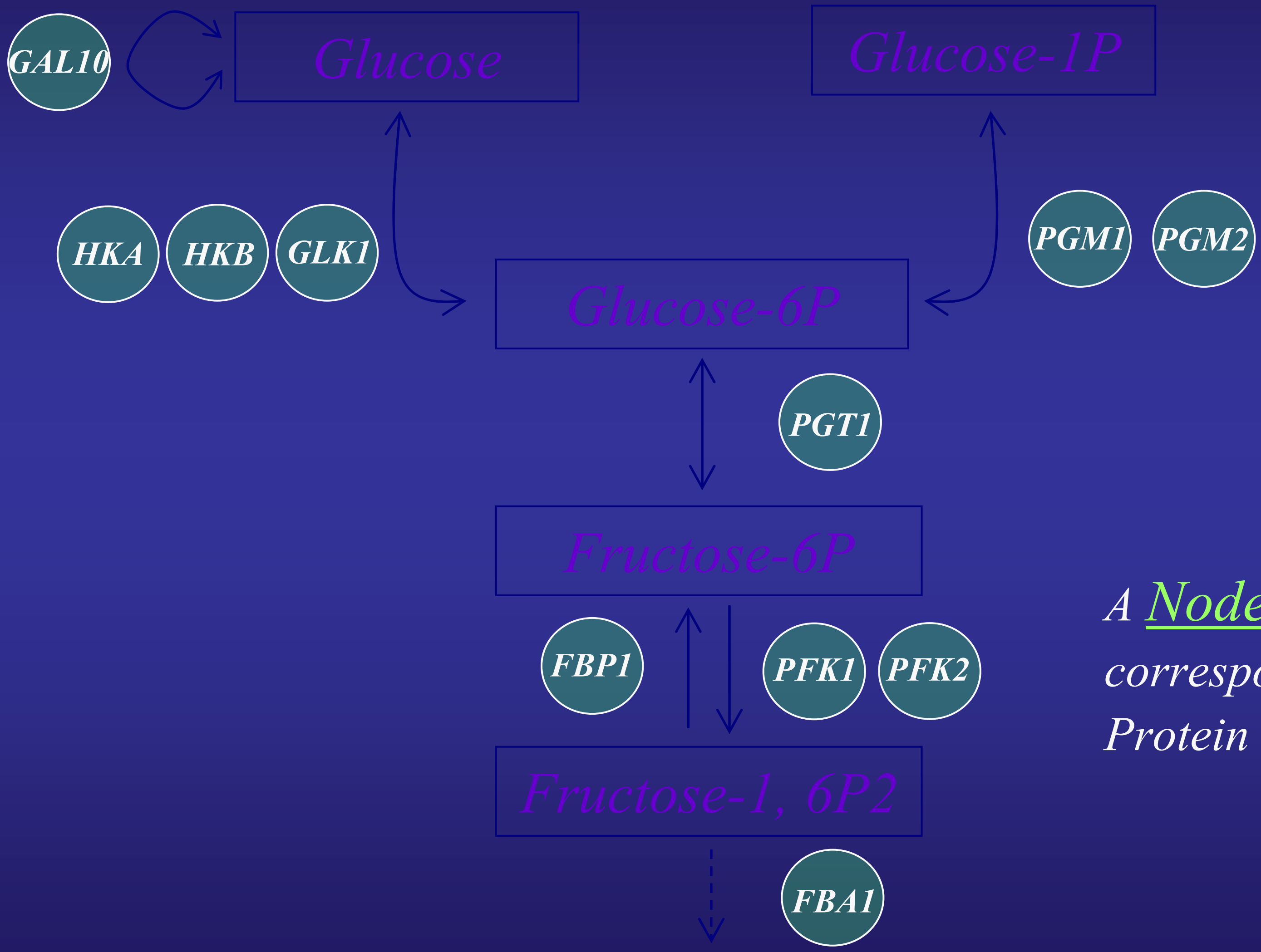
Fructose-1, 6P2

Graph Representation on Biological Networks



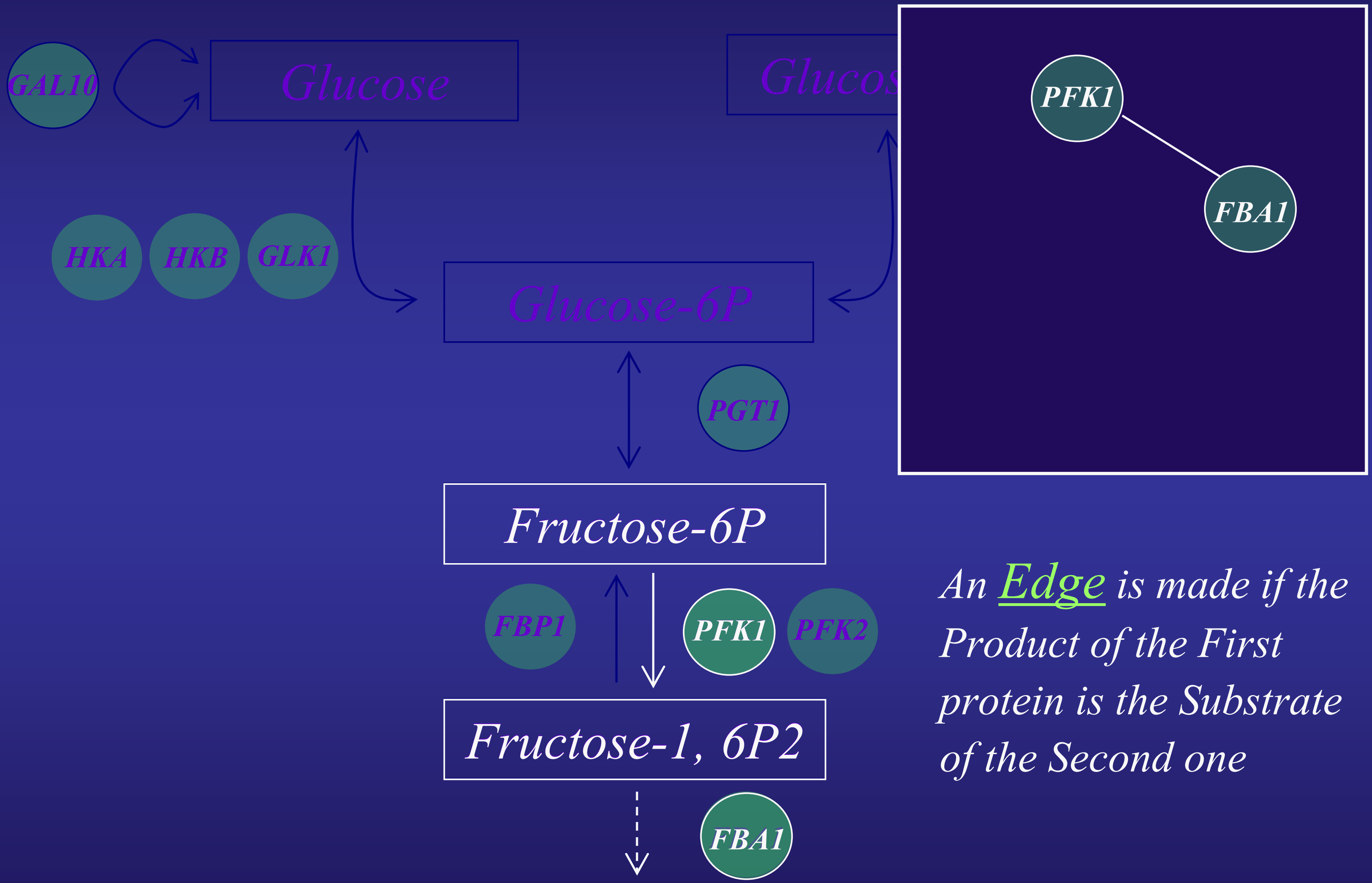
How to Make a Graph ?

Graph Representation on Biological Networks



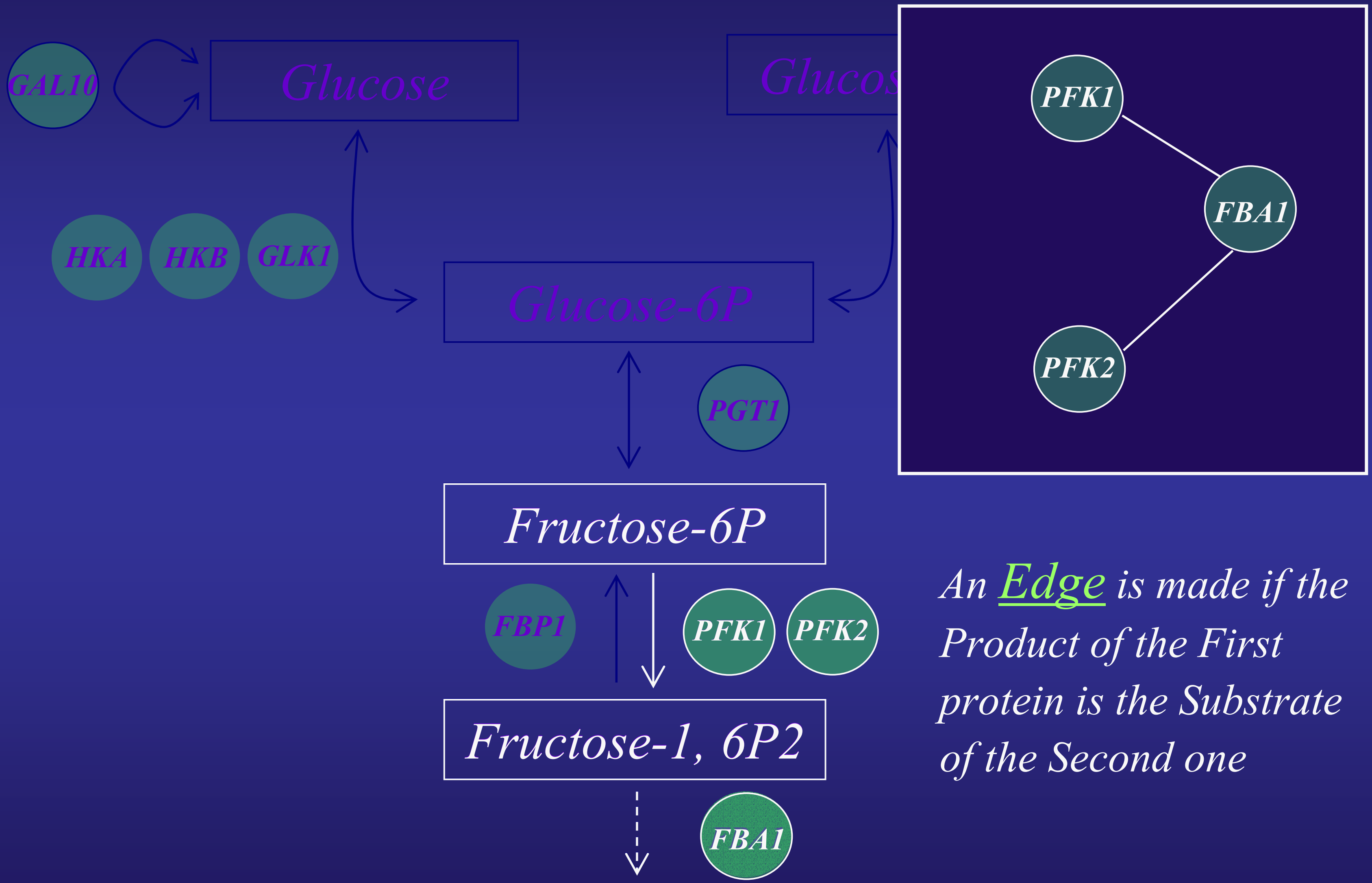
A ***Node*** *corresponds to a Protein*

Graph Representation on Biological Networks



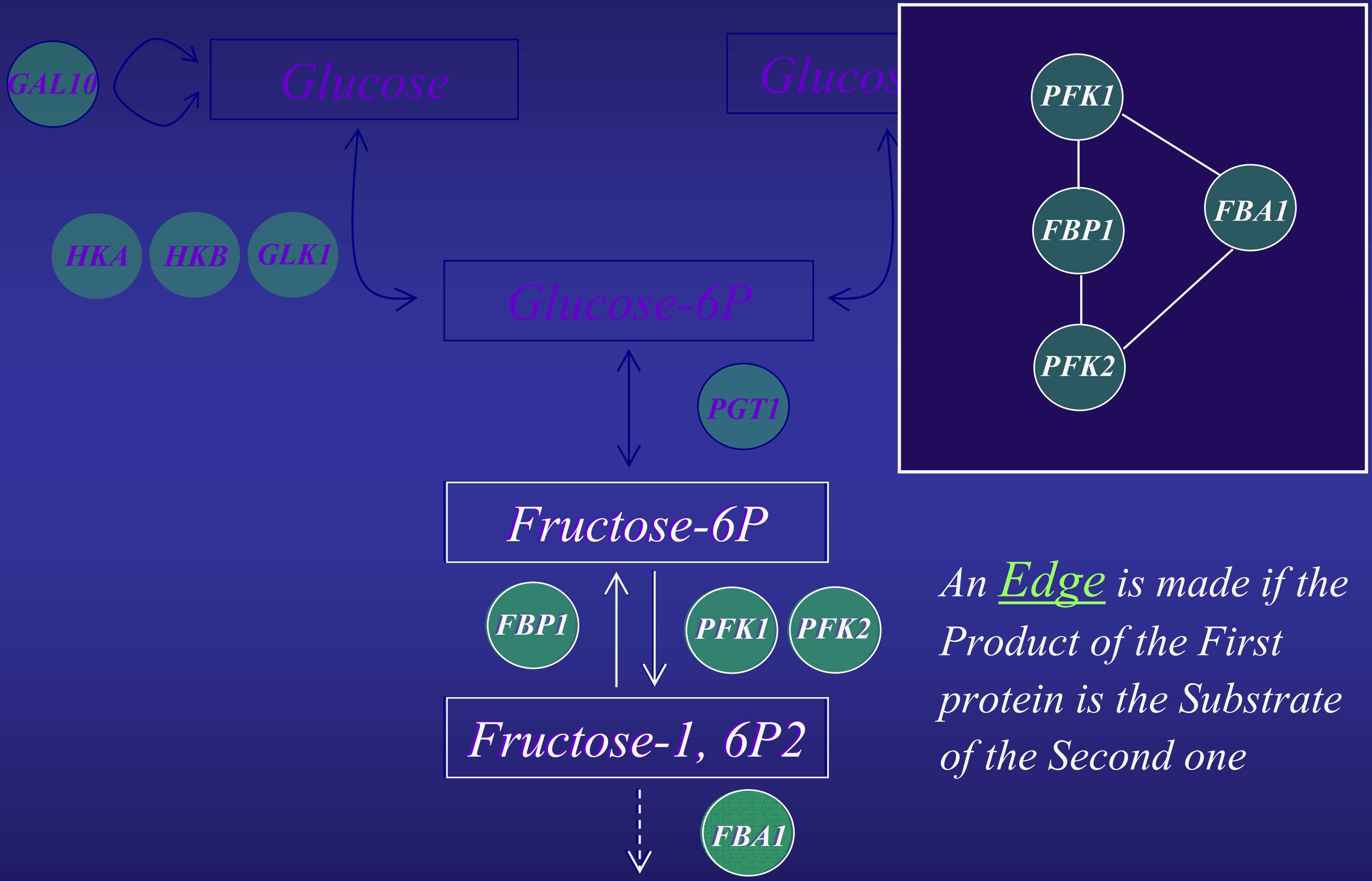
An *Edge* is made if the Product of the First protein is the Substrate of the Second one

Graph Representation on Biological Networks



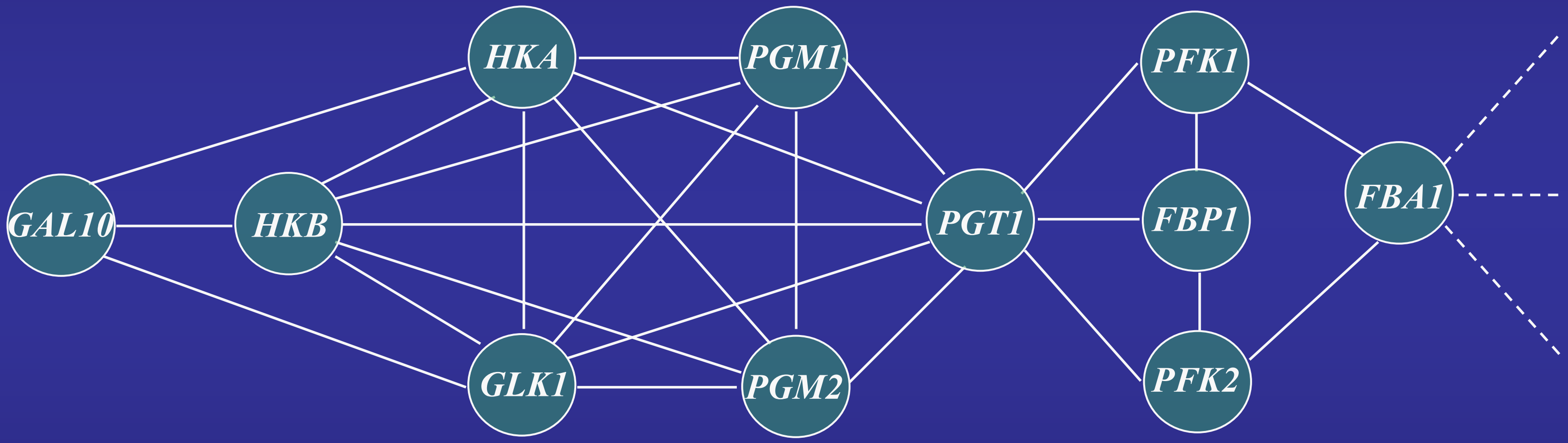
An Edge is made if the Product of the First protein is the Substrate of the Second one

Graph Representation on Biological Networks

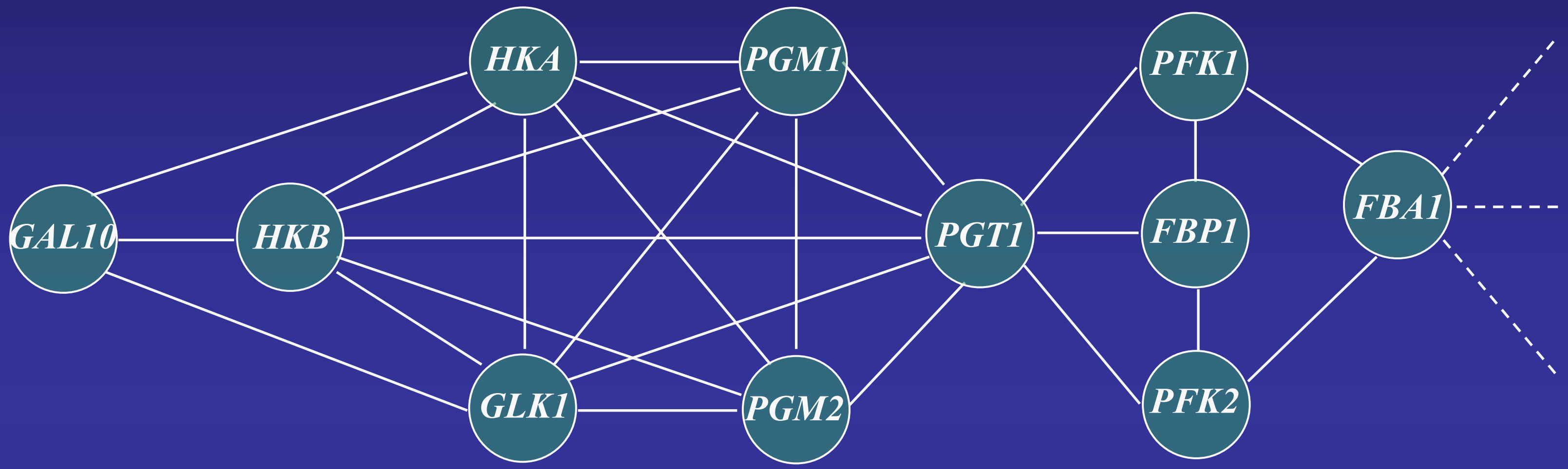


An *Edge* is made if the Product of the First protein is the Substrate of the Second one

Graph Representation on Biological Networks



Semi-Supervised Learning with a Single Graph



Objective Function

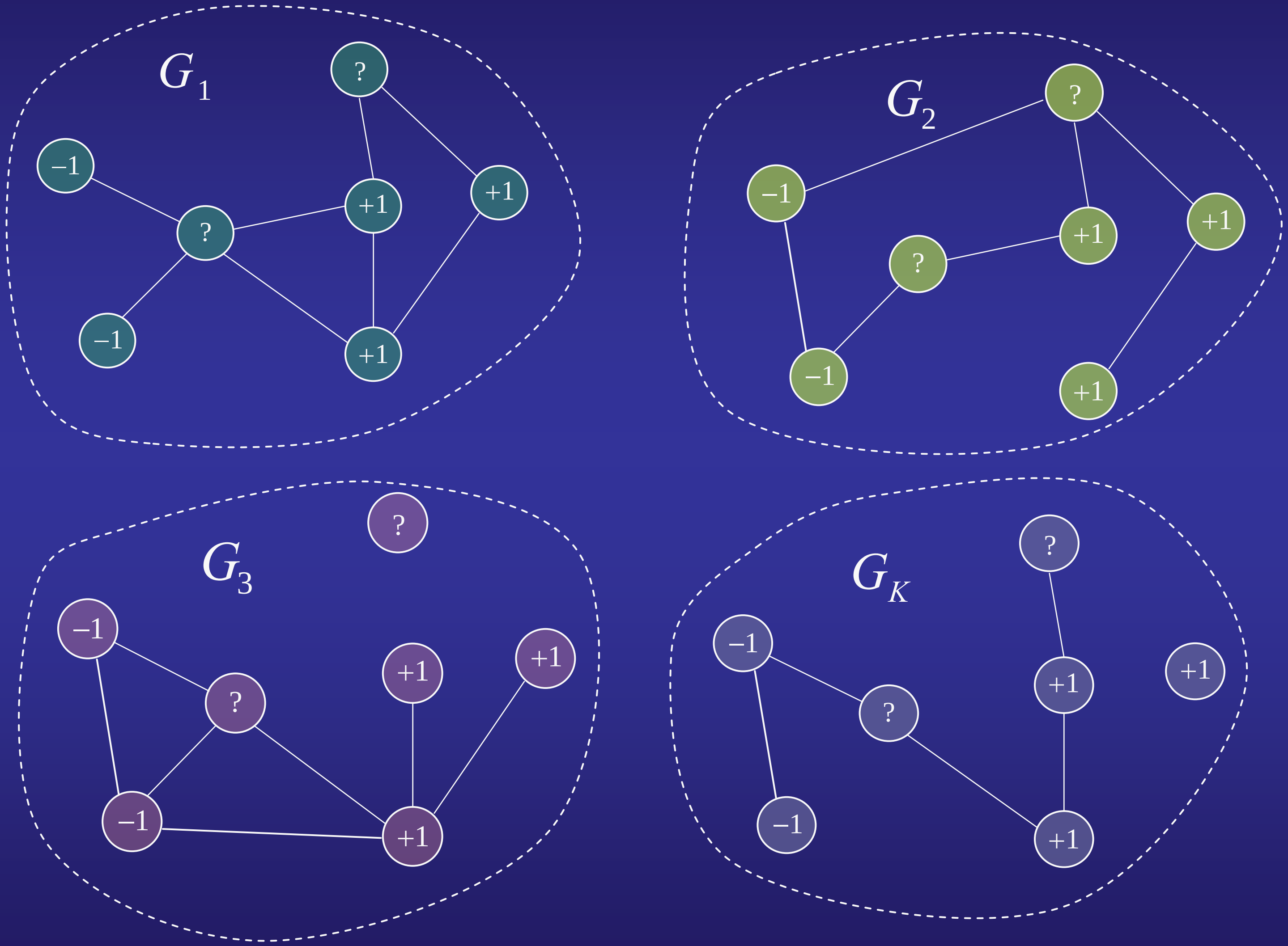
$$\min_{\boldsymbol{f}} \quad \mu\, \boldsymbol{f}^T L\, \boldsymbol{f} + (\boldsymbol{f} - \boldsymbol{y})^T (\boldsymbol{f} - \boldsymbol{y})$$

Solution

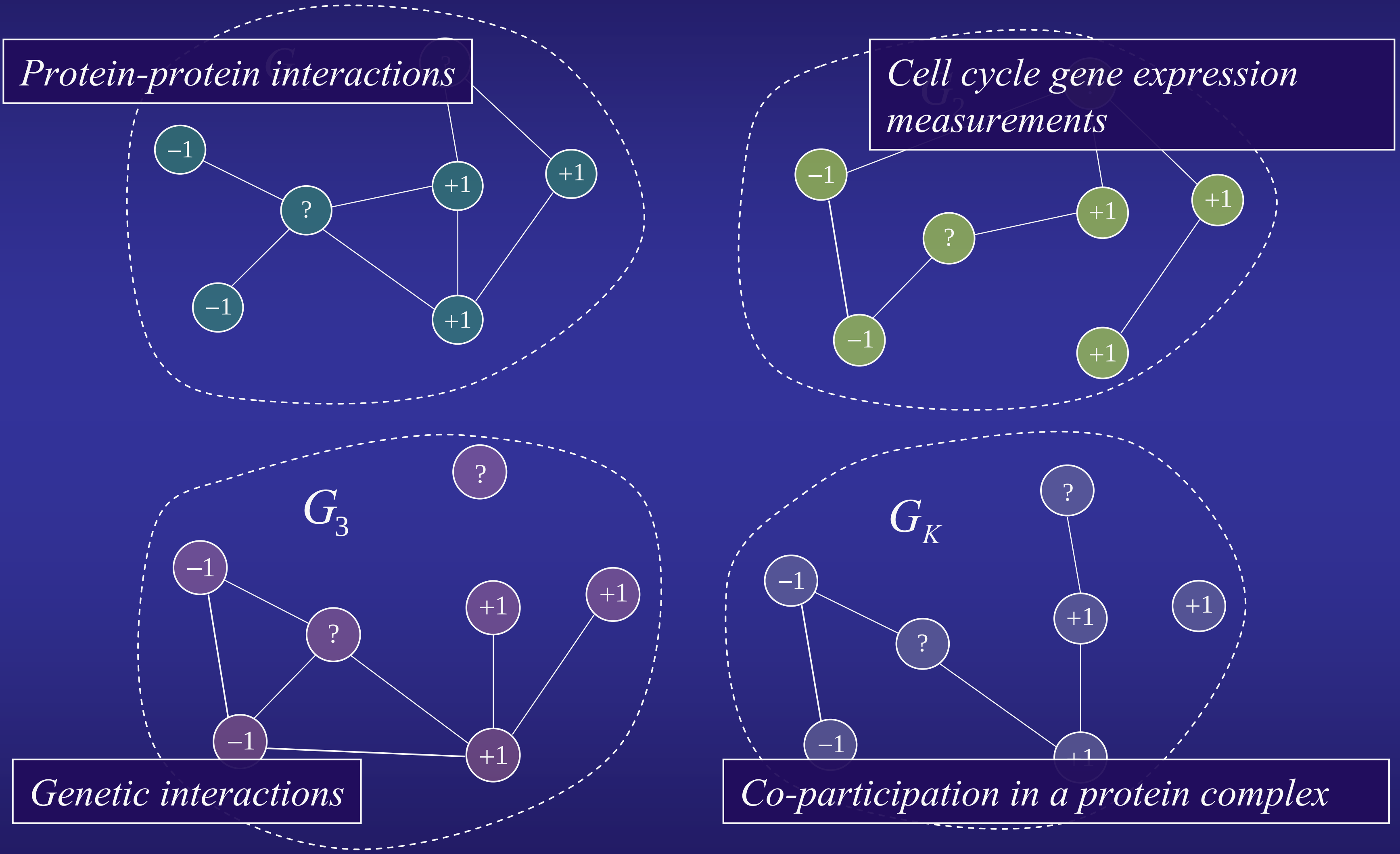
$$\boldsymbol{f} = \{\ \boldsymbol{I} + \mu\, L\ \}^{-1} \boldsymbol{y}$$

If Multiple Graphs are Given ?

If Multiple Graphs are Given?



If Multiple Graphs are Given?



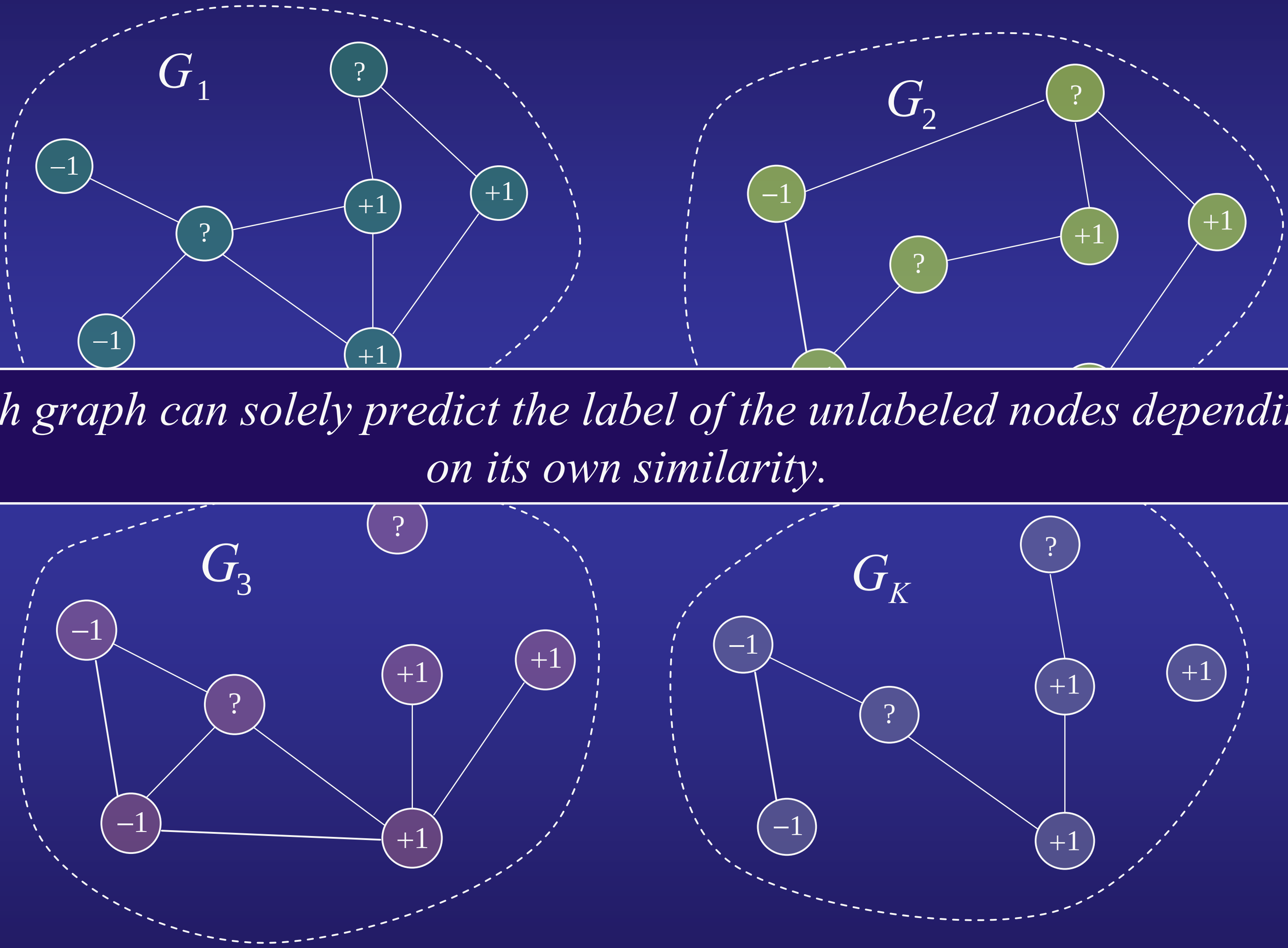
Protein-protein interactions

Cell cycle gene expression measurements

Genetic interactions

Co-participation in a protein complex

If Multiple Graphs are Given?



Each graph can solely predict the label of the unlabeled nodes depending on its own similarity.

If Multiple Graphs are Given?

Since different graphs contain
partly independent and partly complementary
pieces of information about the problem at hand,

one thus can enhance the total information about the problem
by combining those graphs.

If Multiple Graphs are Given?

Example: Multiple Graph Sources on Proteins

Physical interactions of the proteins
[Schwikowski,et al., 2000, Uetz et al., 2000, von Mering et al., 2002]

Gene regulatory relationships
[Lee et al., 2002, Ihmels et al., 2002, Segal et al., 2003]

Edges in a metabolic pathway [Kanehisa et al., 2004]

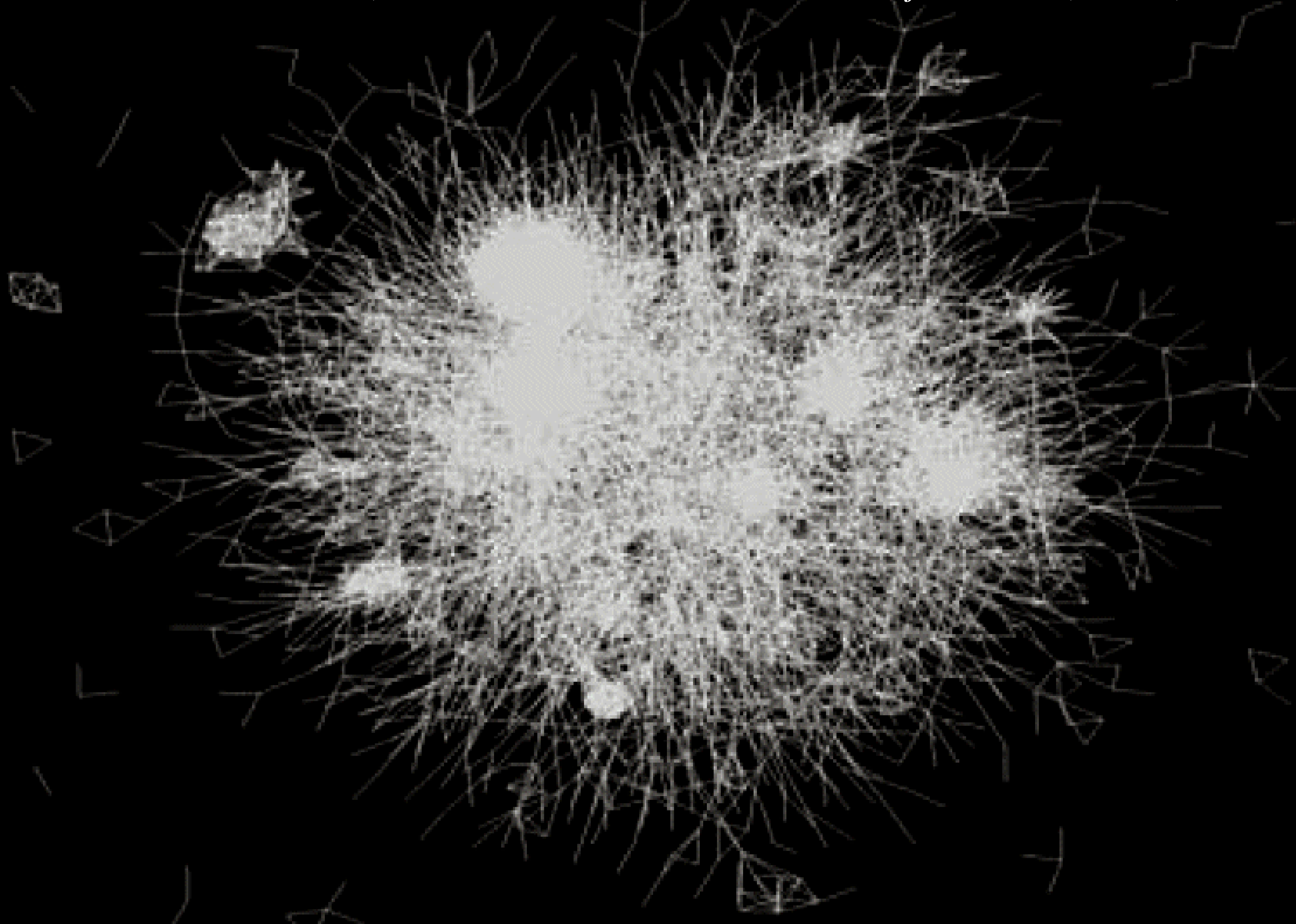
Similarities between protein sequences [Yona et al., 1999]

etc.

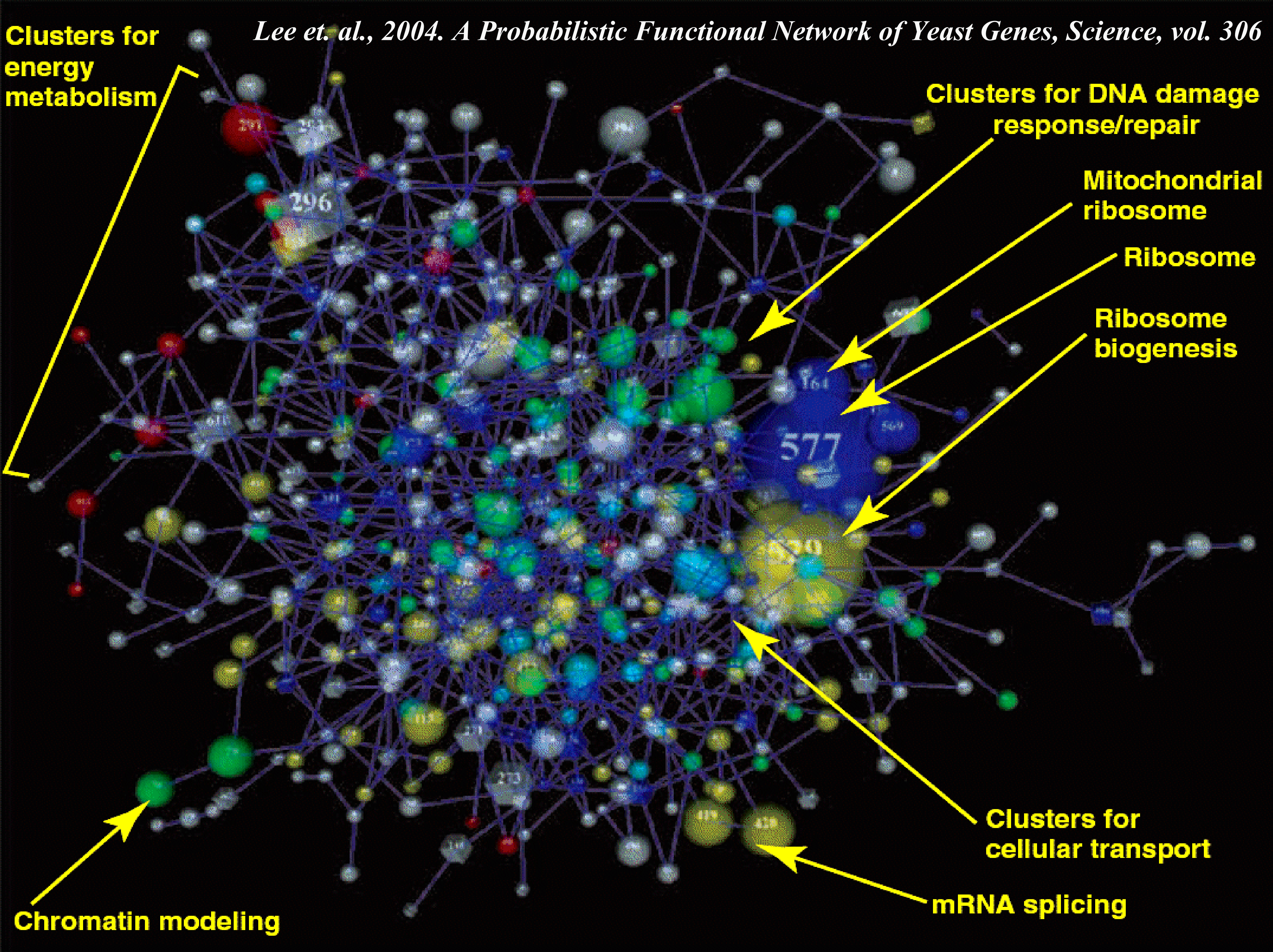
If Multiple Graphs are Given?

Lee et. al., 2004.
A Probabilistic Functional Network of Yeast Genes,
Science, vol. 306

Lee et. al., 2004. A Probabilistic Functional Network of Yeast Genes, Science, vol. 306



Lee et. al., 2004. A Probabilistic Functional Network of Yeast Genes, Science, vol. 306



Clusters for energy metabolism

Clusters for DNA damage response/repair

Mitochondrial ribosome

Ribosome

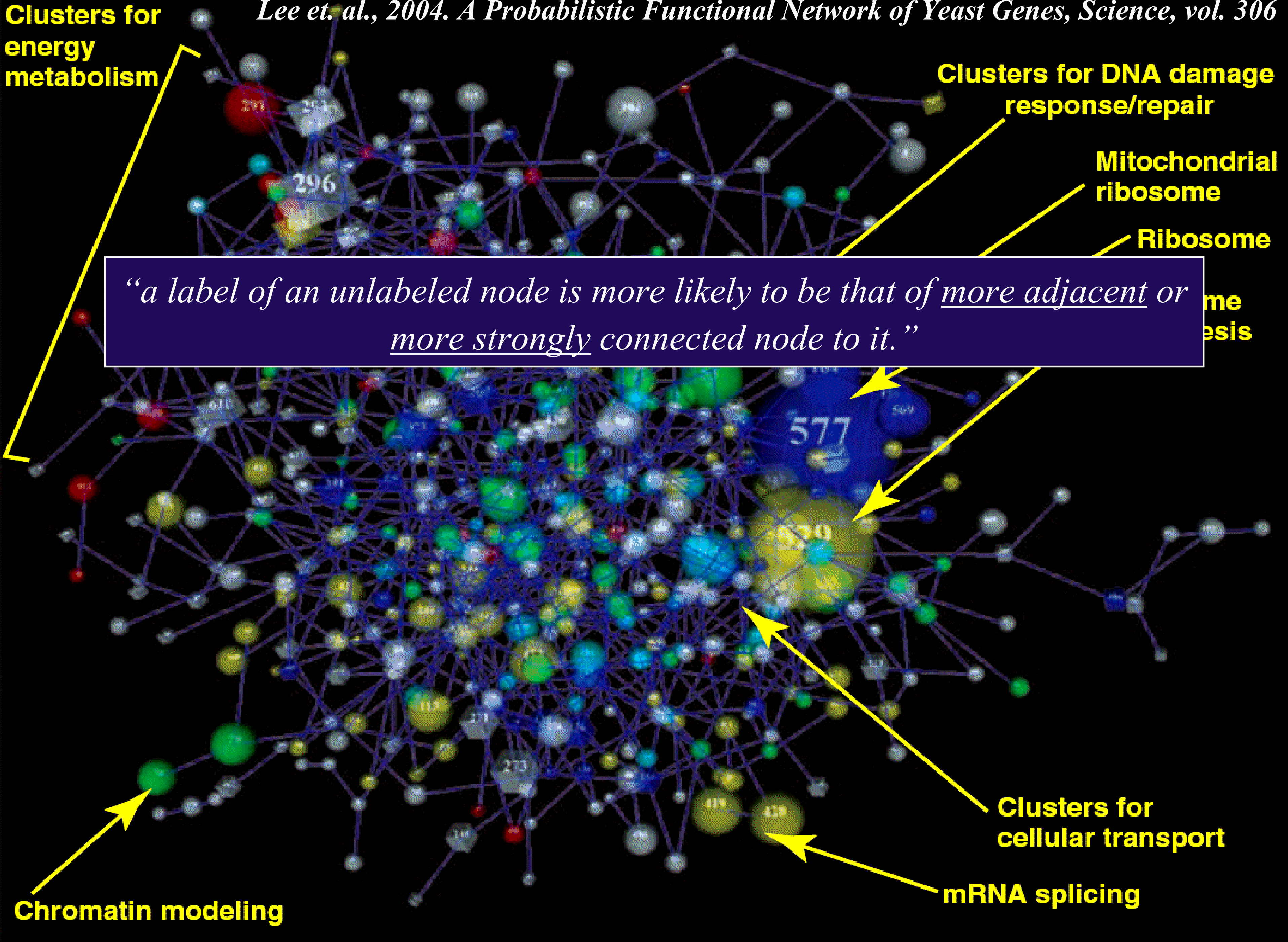
Ribosome biogenesis

Clusters for cellular transport

mRNA splicing

Chromatin modeling

Lee et. al., 2004. A Probabilistic Functional Network of Yeast Genes, Science, vol. 306



Clusters for energy metabolism

Clusters for DNA damage response/repair

Mitochondrial ribosome

Ribosome

me
esis

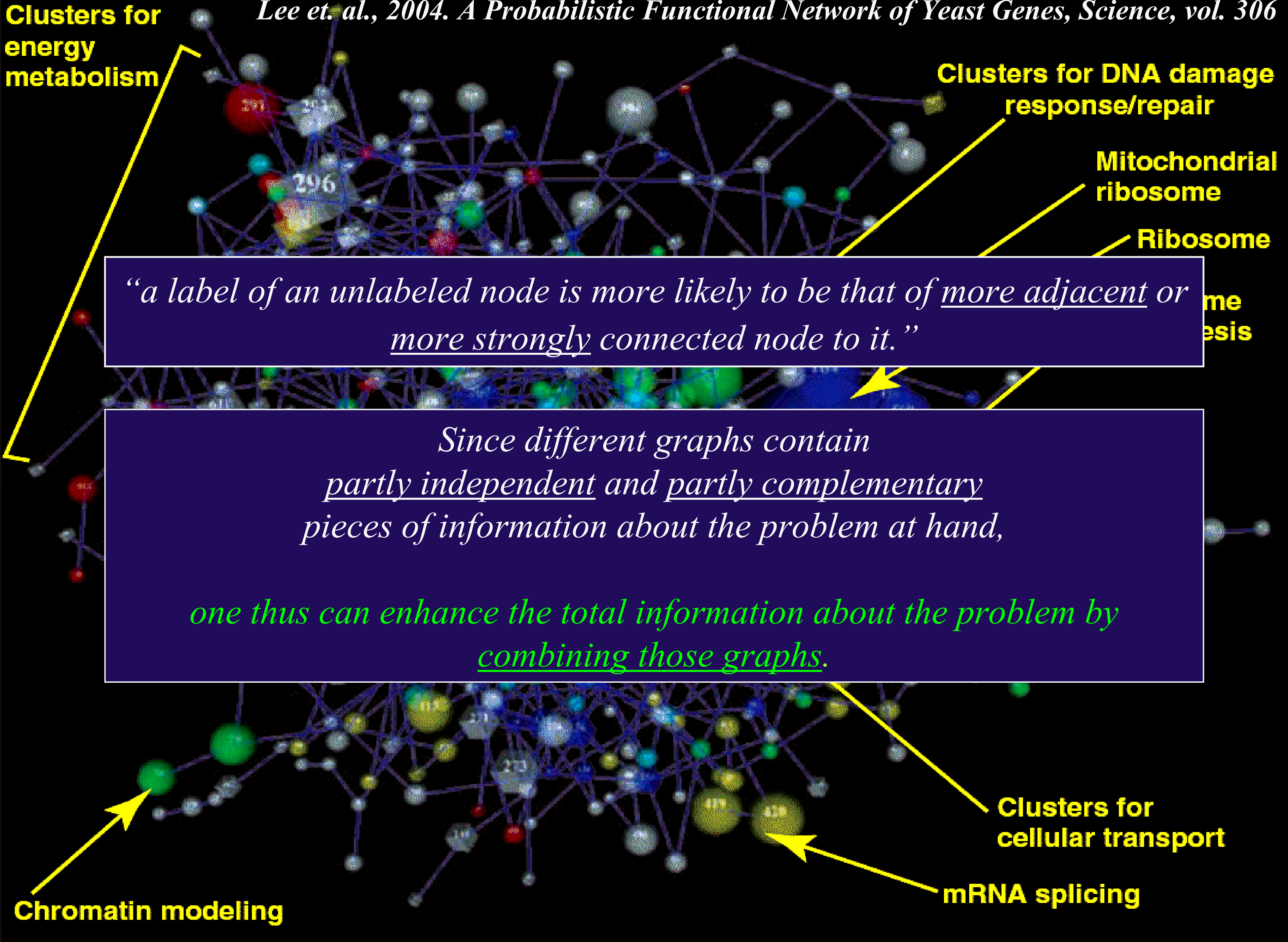
"a label of an unlabeled node is more likely to be that of <u>more adjacent</u> or <u>more strongly</u> connected node to it."

Clusters for cellular transport

mRNA splicing

Chromatin modeling

Lee et. al., 2004. A Probabilistic Functional Network of Yeast Genes, Science, vol. 306


Clusters for energy metabolism
Clusters for DNA damage response/repair
Mitochondrial ribosome
Ribosome
me
esis
Clusters for cellular transport
mRNA splicing
Chromatin modeling
296

"a label of an unlabeled node is more likely to be that of <u>more adjacent</u> or <u>more strongly</u> connected node to it."

Since different graphs contain
<u>partly independent</u> and <u>partly complementary</u>
pieces of information about the problem at hand,

one thus can enhance the total information about the problem by <u>combining those graphs</u>.

If Multiple Graphs are Given?

Previous Approach

SDP/SVM : Semi-Definite Programming based Support Vector Machine

[Lanckriet et al., Bioinfomatics, 2004]

SDP/SVM (Kernel Method)

Diffusion Kernel	*Each graph is converted to a kernel matrix*
SDP	*Kernel matrices are combined with weights which are automatically learned by Semi-Definite Programming*
SVM	*Labels are predicted based on the combined kernel matrix*

$$K(\mu) = \mu_1 \boxed{K_1} + \mu_2 \boxed{K_2} + \mu_3 \boxed{K_3} + \cdots\cdots + \mu_k \boxed{K_K}$$

SDP/SVM (Kernel Method)

Diffusion Kernel

Each graph is converted to a kernel matrix

SDP

Kernel matrices are combined with weights which are automatically learned by Semi-Definite Programming

SVM

Labels are predicted based on the combined kernel matrix

Good accuracy
which is much better than Markov Random Field

But
Very Slow

SDP/SVM (Kernel Method)

SDP/SVM : Semi-Definite Programming based SVM

In SDP/SVM, multiple kernel matrices corresponding to each of data sources are combined with weights obtained by solving an SDP.

However, when trying to apply SDP/SVM to large problems, the computational cost can become prohibitive, since both Converting the data to a kernel matrix *for the SVM and* Solving the SDP *are* *time and memory demanding*

SDP/SVM (Kernel Method)

Diffusion Kernel

[Kondor and Lafferty, 2002].

$$K_{\beta} = e^{\beta \boldsymbol{L}} = \lim_{s \to \infty} (I + \frac{\beta \boldsymbol{L}}{s})^{s} = I + \beta \boldsymbol{L} + \frac{\beta^{2}}{2} \boldsymbol{L}^{2} + \frac{\beta^{3}}{6} \boldsymbol{L}^{3} + \ldots$$

L : graph Laplacian.

β : diffusion rate

SDP/SVM (Kernel Method)

Semi-Definite Programming

[Vandenberg and Boyd, 1996]
[Boyd and Vandenberg, 2003]

$$\min_{\boldsymbol{u}} \quad \boldsymbol{c}^T \boldsymbol{u}$$

$$s.t. \quad F^j(\boldsymbol{u}) = F_o + \sum_{k=1}^{K} u_k F_k \geq 0, \qquad j = 1,\ldots,J.$$

where $\boldsymbol{c} \in R^K, \; F_k \in R^{n\times n}, \; F(\boldsymbol{u}) \in R^{n\times n}$: symmetric, positive semi - definite

Convex optimization problem since its objective and constrains are convex

SDP/SVM (Kernel Method)

SDP/SVM

[Lanckriet et al., 2004]

Single Kernel Case

SVM dual problem

$$\max_{\boldsymbol{\alpha}} \quad 2\boldsymbol{\alpha}^{\boldsymbol{T}}\boldsymbol{e} - \boldsymbol{\alpha}^{\boldsymbol{T}}(G(K)+\tau I)\boldsymbol{\alpha}$$

$$s.t. \quad 0 \le \boldsymbol{\alpha} \le C, \quad \boldsymbol{\alpha}^{\boldsymbol{T}} y = 0,$$

$$trace(K) = c.$$

$$where \quad G_{ij}(K) = k(x_i, x_j) y_i y_j$$

SVM cast as an SDP

$$\min_{K, t, \lambda, \boldsymbol{v}, \boldsymbol{\delta}} \quad t$$

$$s.t. \quad trace(K) = c,$$

$$\begin{pmatrix} G(K_{tr}) + \tau I_{ntr} & \boldsymbol{e}+\boldsymbol{v}-\boldsymbol{\delta}+\lambda\boldsymbol{y} \\ (\boldsymbol{e}+\boldsymbol{v}-\boldsymbol{\delta}+\lambda\boldsymbol{y})^T & t - 2C\boldsymbol{\delta}^T\boldsymbol{e} \end{pmatrix} \succeq 0,$$

$$\boldsymbol{v} \ge 0,$$

$$\boldsymbol{\delta} \ge 0.$$

SDP/SVM (Kernel Method)

SDP/SVM

[Lanckriet et al., 2004]

Multiple Kernels

SVM cast as an SDP

$$\min_{K,t,\lambda,\boldsymbol{v},\boldsymbol{\delta}} \quad t$$

$$s.t. \quad trace\left(\sum_{i=1}^{m}\mu_i K_i\right) = c,$$

$$\sum_{i=1}^{m}\mu_i K_i \geq 0,$$

$$\begin{pmatrix} G\left(\sum_{i=1}^{m}\mu_i K_{i,tr}\right) + \tau I_{ntr} & \boldsymbol{e}+\boldsymbol{v}-\boldsymbol{\delta}+\lambda\boldsymbol{y} \\ (\boldsymbol{e}+\boldsymbol{v}-\boldsymbol{\delta}+\lambda\boldsymbol{y})^T & t-2C\boldsymbol{\delta}^T\boldsymbol{e} \end{pmatrix} \geq 0,$$

$$\boldsymbol{v} \geq 0,$$

$$\boldsymbol{\delta} \geq 0.$$

SDP/SVM (Kernel Method)

Calculating a Diffusion Kernel from a Graph

$O(n^3)$, *A dense matrix of n x n*

Solving SDP

$O((m+n)^2 n^{2.5})$

m: the number of kernel matrices
n: number of nodes (data)

Computationally Expensive both in Time and Memory

Why not use a more direct approach for combining graphs based on significant progress of graph-based semi-supervised learning methods ?

- *Zhou et al., 2004*
- *Belkin and Niyogi, 2003*
- *Zhu et al., 2003*
- *Chapelle et al., 2003*

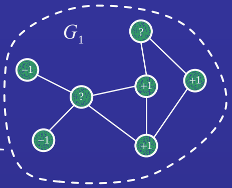
Semi-Supervised Learning Extension to Multiple Graphs

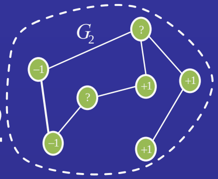
- *Combining weights are automatically assigned to Graphs*
- *Comparable Accuracy to SDP/SVM*
- *Very Fast*

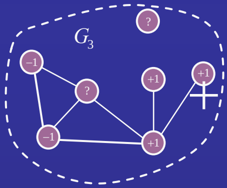
Extension to Multiple Graphs

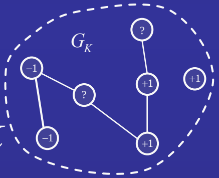
Linear Combination of Laplacians

$$L(\beta) = \sum_{k=1}^{K} \beta_k L_k$$

$L(\beta) = \beta_1$ 

$+ \beta_2$ 

$+ \beta_3$ 

$+ \cdots\cdots + \beta_k$ 

How to Find Combining Weights ?

Extension to Multiple Graphs

Single Graph

$$\min_{\boldsymbol{f}} \quad (\boldsymbol{f}-\boldsymbol{y})^T(\boldsymbol{f}-\boldsymbol{y}) + \mathrm{c}\, \boldsymbol{f}^T L\, \boldsymbol{f}$$

Without loss of generality, the problem is rewritten
by penalizing the upper-bound

$$\min_{\boldsymbol{f},\gamma} \quad (\boldsymbol{f}-\boldsymbol{y})^T(\boldsymbol{f}-\boldsymbol{y}) + c\gamma, \qquad \boldsymbol{f}^T L\, \boldsymbol{f} \le \gamma.$$

Extension to Multiple Graphs

Multiple Graphs

$$\min_{\boldsymbol{f}} \quad (\boldsymbol{f}-\boldsymbol{y})^T(\boldsymbol{f}-\boldsymbol{y}) + \mathrm{c}\{\beta_1 \boldsymbol{f}^T L_1 \boldsymbol{f} + \beta_2 \boldsymbol{f}^T L_2 \boldsymbol{f} + \ldots + \beta_\mathrm{k} \boldsymbol{f}^T L_k \boldsymbol{f}\}$$

Without loss of generality, the problem is rewritten
by penalizing the upper-bound

$$\min_{f,\gamma} \quad (\boldsymbol{f}-\boldsymbol{y})^T(\boldsymbol{f}-\boldsymbol{y}) + c\gamma, \quad \boldsymbol{f}^T L_k \boldsymbol{f} \le \gamma, \quad k = 1,\ldots K.$$

Extension to Multiple Graphs: Optimization

Primal

$$\min_{f,\gamma} \quad (\boldsymbol{f}-\boldsymbol{y})^T(\boldsymbol{f}-\boldsymbol{y})+c\gamma,$$

$$\text{s.t.} \quad \boldsymbol{f}^T L_k \boldsymbol{f} \le \gamma,$$

$$\gamma \ge 0, \quad k = 1,\ldots K.$$

Dual

$$\min_{\boldsymbol{\beta}} \; d(\boldsymbol{\beta}) \equiv \boldsymbol{y}^T(\boldsymbol{I}+\sum_{k=1}^{K} \beta_k \; L_k)^{-1}\boldsymbol{y},$$

$$\text{s.t.} \quad \sum_{k=1}^{K} \beta_k \le c$$

β_k : *Weight for Network k, Lagrange Multiplier.*

Extension to Multiple Graphs: Solution

Solution

$$\boldsymbol{f} = \left\{ \boldsymbol{I} + \sum_{k=1}^{K} \beta_k L_k \right\}^{-1} \boldsymbol{y}$$

Matrix Inversion

β_k : *Weight for Network k, Lagrange Multiplier.*

Sparse Linear System

$$\boldsymbol{y} = \left\{ \boldsymbol{I} + \sum_{k=1}^{K} \beta_k L_k \right\} \boldsymbol{f}$$

Linear Systems

Extension to Multiple Graphs: Meaning of Weights

By KKT complementarity condition, we have the following relationship at the optimal solution,

$$\beta_k(\boldsymbol{f}^T L_k \boldsymbol{f} - \delta) = 0$$

$$\beta_k = 0 \quad \text{iff} \quad \boldsymbol{f}^T L_k \boldsymbol{f} < \delta$$

*The score vector **f** would not be changed much with those graphs, thus those are considered as redundant*

$$\beta_k > 0 \quad \text{iff} \quad \boldsymbol{f}^T L_k \boldsymbol{f} = \delta$$

Those graphs are considered important.

Extension to Multiple Graphs

Computational Efficiency

1. *Repetition of an Identical Form of Inverse Matrix*
2. *Implicit Calculation of Matrix Inversion*

Extension to Multiple Graphs

Computational Efficiency

1. Repetition of an Identical Form of Inverse Matrix:
in the objective function and the derivative, (and the network output).

Objective Function

$$\min_{\boldsymbol{\beta}} \; d(\boldsymbol{\beta}) \equiv \boldsymbol{y}^T \boxed{(\boldsymbol{I} + \sum_{k=1}^{K} \beta_k L_k)^{-1} \boldsymbol{y}}$$

Solution Update

$$\frac{\partial d}{\partial \beta_k} = -\boxed{\boldsymbol{y}^T (I + \sum_{j=1}^{K} \beta_j L_j)^{-1}} L_k \boxed{(I + \sum_{j=1}^{K} \beta_j L_j)^{-1} \boldsymbol{y}}$$

Network Output

$$\boldsymbol{f} = \boxed{\left\{ \boldsymbol{I} + \sum_{k=1}^{K} \beta_k L_k \right\}^{-1} \boldsymbol{y}}$$

Extension to Multiple Graphs

Computational Efficiency

2. Implicit Calculation of Matrix Inversion:

The solution can be obtained by solving the "sparse linear systems."

Therefore, computational cost is nearly linear in the number of non-zero entries of $\sum_{k=1}^{K} \beta_k L_k$ – (Spielman and Teng, 2004).

Matrix Inversion

$$\boldsymbol{f} = \left\{ \boldsymbol{I} + \sum_{k=1}^{K} \beta_k L_k \right\}^{-1} \boldsymbol{y}$$

Linear Systems

$$\boldsymbol{y} = \left\{ \boldsymbol{I} + \sum_{k=1}^{K} \beta_k L_k \right\} \boldsymbol{f}$$

Function Prediction Experiments

MIPS Comprehensive Yeast Genome Database (CYGD-mips.gsf.de/proj/yeast).

Data	*3588 yeast proteins*
Output	*13 functional categories* *Binary classification for each category*
Input	*5 networks*
Setting	*5 fold cross validation* *5 times repetition*

Protein Functional Categories

MIPS Comprehensive Yeast Genome Database (CYGD-mips.gsf.de/proj/yeast).

13 CYGD functional Classes

1. metabolism
2. energy
3. cell cycle and DNA processing
4. transcription
5. protein synthesis
6. protein fate
7. cellular transportation and transportation mechanism
8. cell rescue, defense and virulence
9. interaction with cell environment
10. cell fate
11. control of cell organization
12. transport facilitation
13. others

Inputs (5 networks)

W_1 Network created from **Pfam domain structure**. A protein is represented by a 4950-dimensional binary vector, in which each bit represents the presence or absence of one Pfam domain. An edge is created if the inner product between two vectors exceeds 0.06. The edge weight corresponds to the inner product.

W_2 **Co-participation in a protein complex** (determined by tandem affinity purification, TAP). An edge is created if there is a bait-prey relationship between two proteins.

W_3 **Protein-protein interactions** (MIPS physical interactions)

W_4 **Genetic interactions** (MIPS genetic interactions)

W_5 Network created from the **cell cycle gene expression measurements** [Spellman et al., 1998]. An edge is created if the Pearson coefficient of two profiles exceeds 0.8. The edge weight is set to 1. This is identical with the network used in [Deng et al., 2003]

Inputs (5 networks)

W_1 Network created from **Pfam domain structure**. A protein is represented by a 4950-dimensional binary vector, in which each bit represents the presence or absence of one Pfam domain. An edge is created if the inner product between two vectors exceeds 0.06. The edge weight corresponds to the inner product.

Inputs (5 networks)

W_2 **Co-participation in a protein complex** (determined by tandem affinity purification, TAP). An edge is created if there is a bait-prey relationship between two proteins.

Inputs (5 networks)

W_3 **Protein-protein interactions** (MIPS physical interactions)

Inputs (5 networks)

W_4 **Genetic interactions** (MIPS genetic interactions)

Inputs (5 networks)

W_5 Network created from the **cell cycle gene expression measurements** [Spellman et al., 1998]. An edge is created if the Pearson coefficient of two profiles exceeds 0.8. The edge weight is set to 1. This is identical with the network used in [Deng et al., 2003]

Inputs (5 networks)

Density of Laplacians (%)

W_1 Network created from Pfam domain structure. A protein is represented by a 4950-dimensional binary vector, in which each bit represents the presence or absence of one Pfam domain. An edge is created if the inner product between two vectors exceeds 0.06. The edge weight corresponds to the inner product.

0.7805

W_2 Co-participation in a protein complex (determined by tandem affinity purification, TAP). An edge is created if there is a bait-prey relationship between two proteins.

0.0570

W_3 Protein-protein interactions (MIPS physical interactions)

0.0565

W_4 Genetic interactions (MIPS genetic interactions)

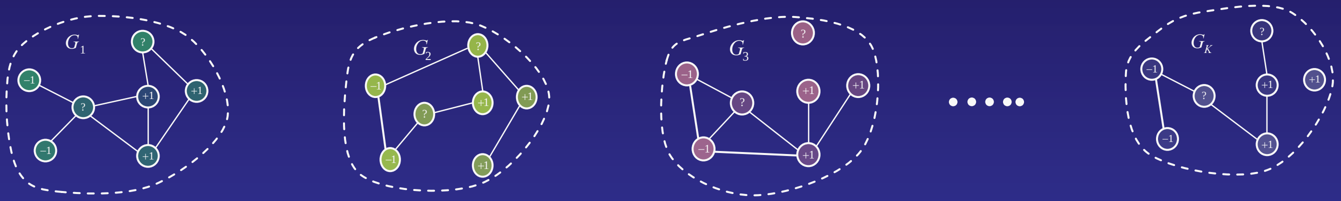
0.0435

W_5 Network created from the cell cycle gene expression measurements [Spellman et al., 1998]. An edge is created if the Pearson coefficient of two profiles exceeds 0.8. The edge weight is set to 1. This is identical with the network used in [Deng et al., 2003]

0.0919

Density of Working Matrices

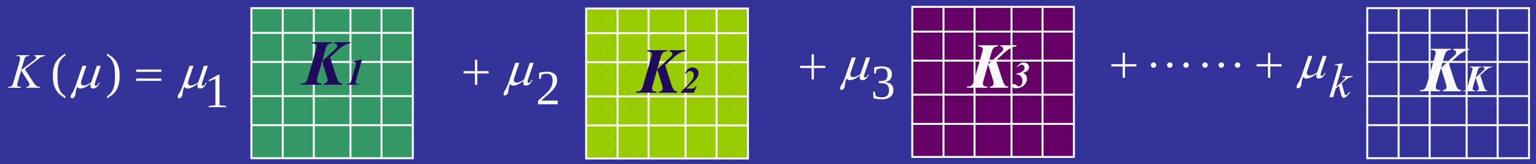
Given Graphs



SDP/SVM

Kernel matrix

Dense

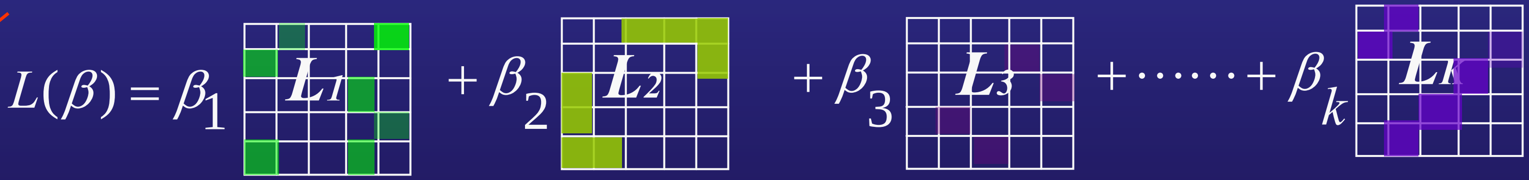


$$K(\mu) = \mu_1 K_1 + \mu_2 K_2 + \mu_3 K_3 + \cdots\cdots + \mu_k K_K$$

SSL

Laplacian matrix L (or Similarity matrix W)

Sparse



$$L(\beta) = \beta_1 L_1 + \beta_2 L_2 + \beta_3 L_3 + \cdots\cdots + \beta_k L_K$$

Inputs (5 networks)

Network created from Pfam domain structure. A protein is represented by a 4950-dimensional binary vector, in which each bit represents the presence or absence of one Pfam domain. An edge is created if the inner product between two vectors exceeds 0.06. The edge weight corresponds to the inner product.

Co-participation in a protein complex (determined by tandem affinity purification, TAP). An edge is created if there is a bait-prey relationship between two proteins.

Protein-protein interactions (MIPS physical interactions)

Genetic interactions (MIPS genetic interactions)

Network created from the cell cycle gene expression measurements [Spellman et al., 1998]. An edge is created if the Pearson coefficient of two profiles exceeds 0.7. The edge weight is set to 1. This is identical with the network used in [Deng et al., 2003]

	Density of Laplacians (%)	*Memory Saving Ratio (%) against Kernels*
W_1	**0.7805**	**1/0.0078=128**
W_2	**0.0570**	**1754**
W_3	**0.0565**	**1770**
W_4	**0.0435**	**2298**
W_5	**0.0919**	**1088**

Methods in Comparison

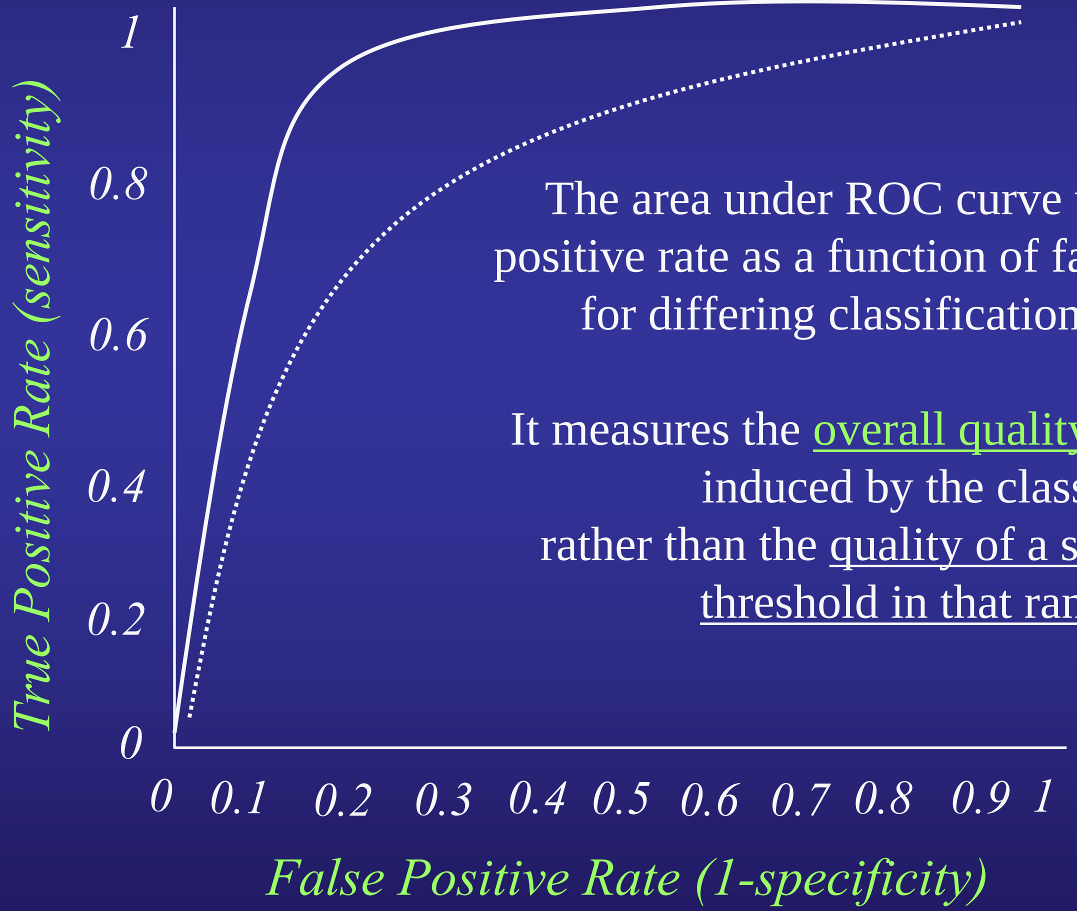
L_k	*Label propagation with an Individual Graphs (k=1…5)*
L_{opt}	*Laplacian of Combined Graph with Optimized Weights*
L_{fix}	*Label propagation with Equal Weights*
MRF	*Markov Random Field, proposed by Deng et al [2003]*
SDP/SVM	*Semi-definite Programming based Support Vector Machines, proposed by Lanckriet et al [2004]*

Measurements

- *ROC (receiver operating characteristic) score*
- *TP1FP , TP10FP*
- *Computational Time*

Measurements

ROC score

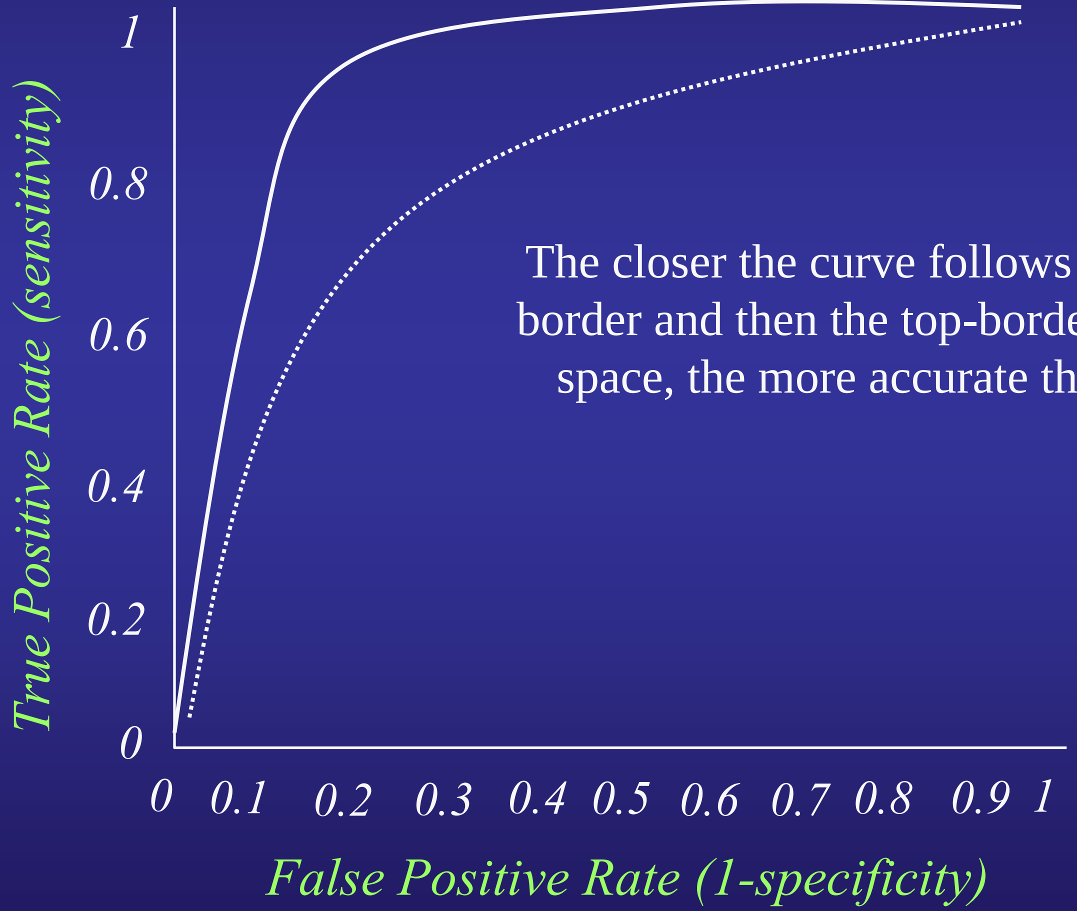


The area under ROC curve that plots true positive rate as a function of false positive rate for differing classification thresholds.

It measures the overall quality of the ranking induced by the classifier, rather than the quality of a single value of threshold in that ranking.

Measurements

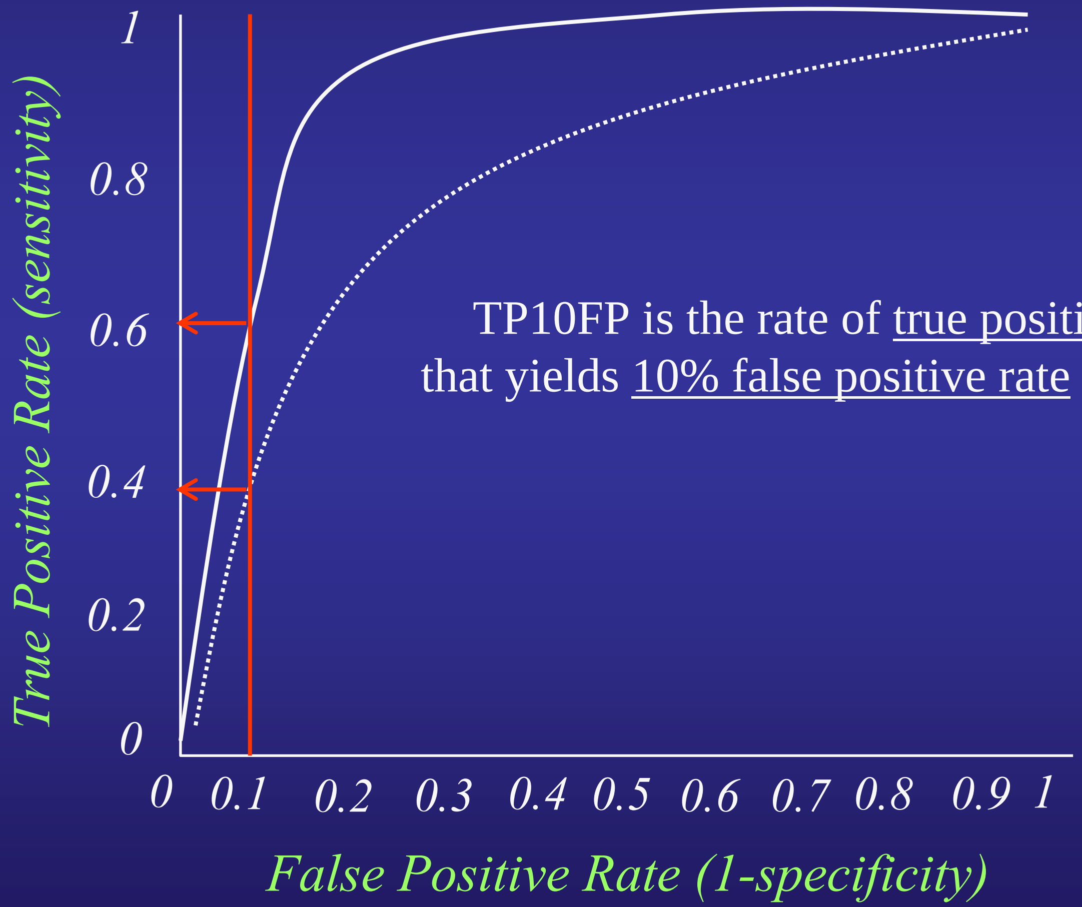
ROC score



The closer the curve follows the left-hand border and then the top-border of the ROC space, the more accurate the classifier.

Measurements

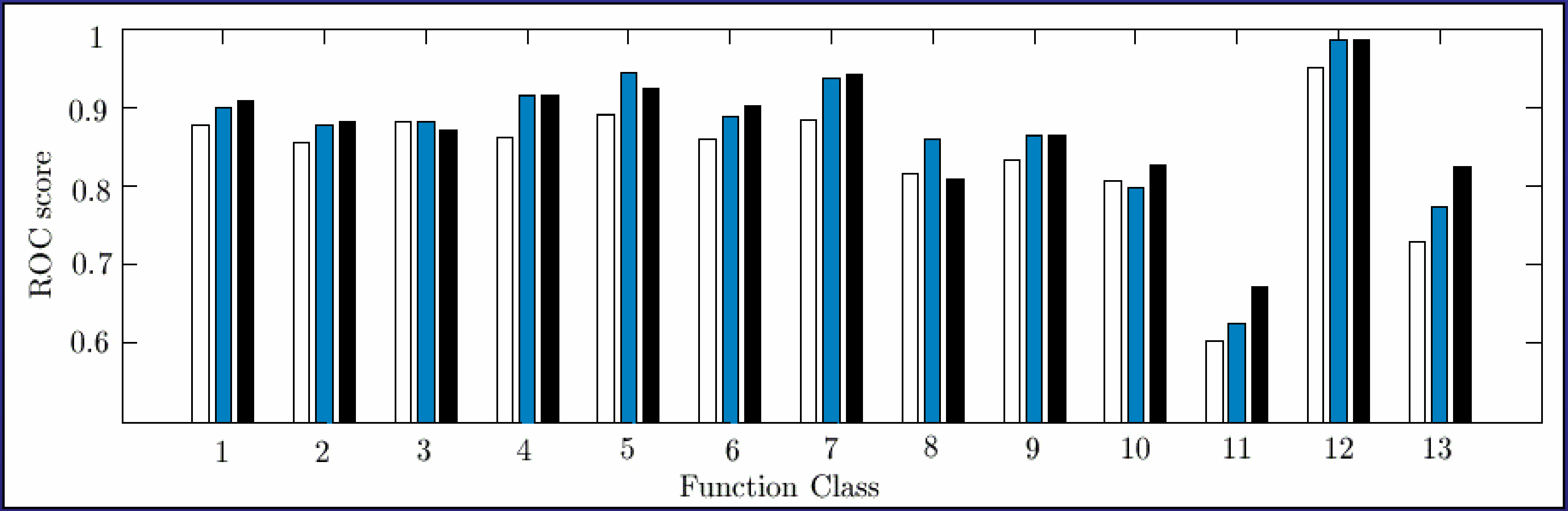
TP10FP



TP10FP is the rate of true positives at the point that yields 10% false positive rate on the ROC curve

Results : *ROC scores of L_{opt}, L_{fix} vs. the Best Performing Individual L_k*

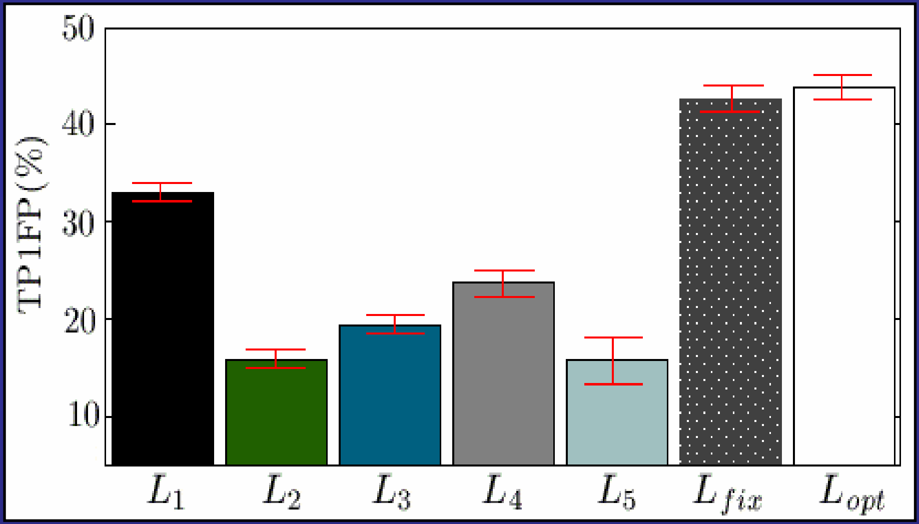
White: the best performing individual network
Blue: L_{fix}
Black: L_{opt}



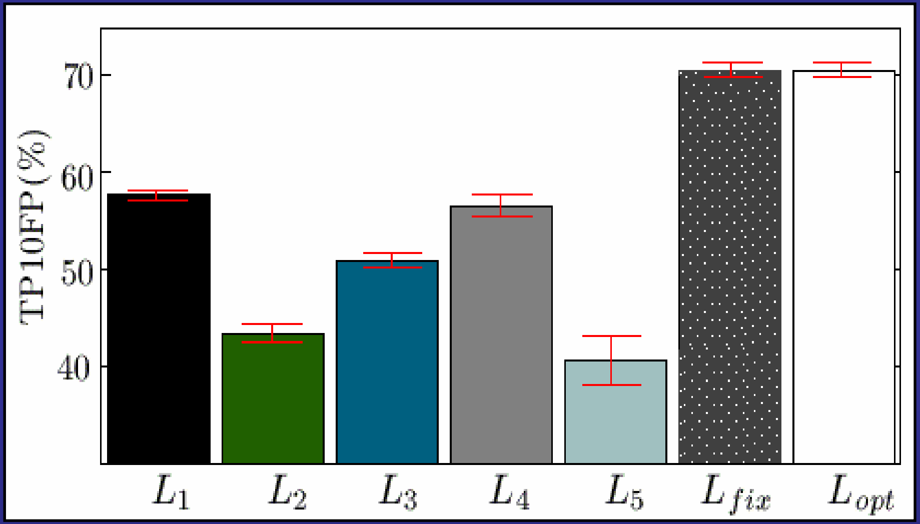
Across the 13 classes, L_{fix} or L_{opt} outperforms the best performing individual.

Results : TP1FP and TP10FP of L_{opt}, L_{fix} vs. Individual L_k's

TP1FP (%)

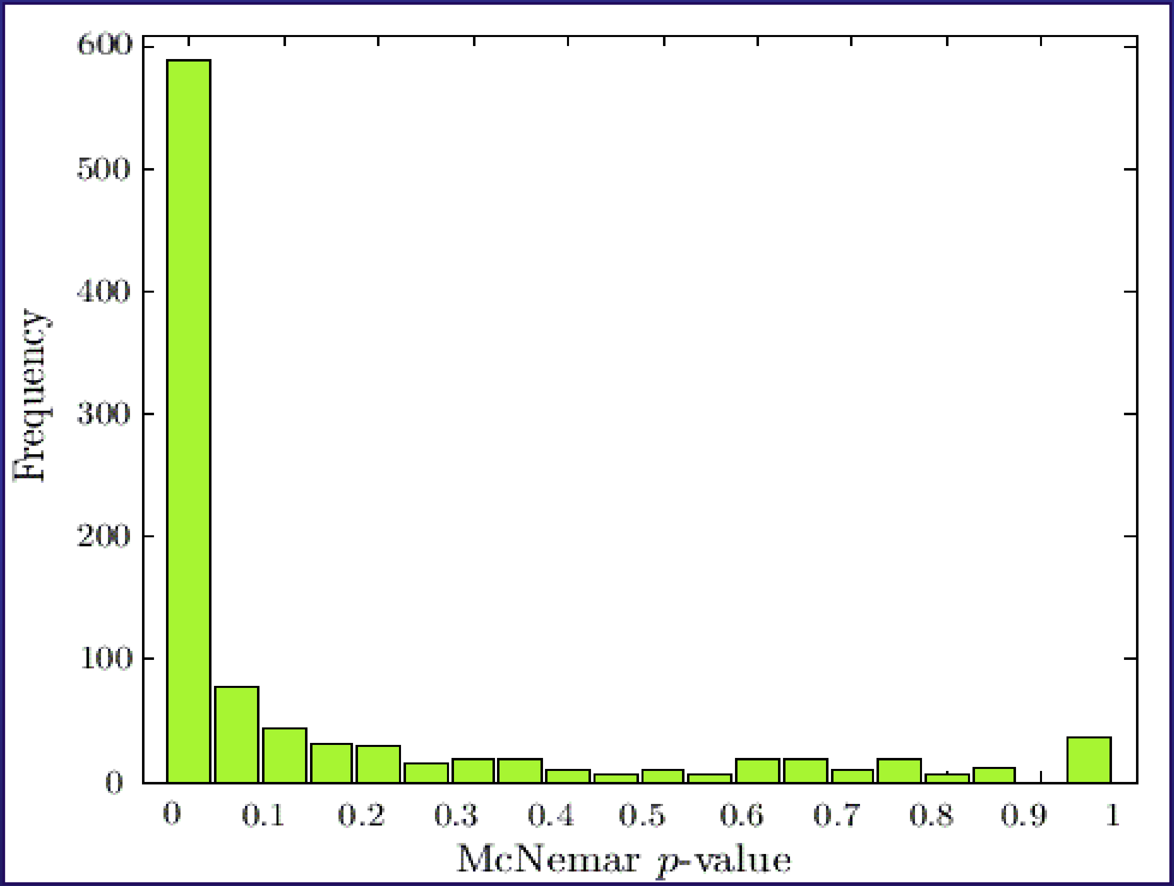


TP10FP (%)



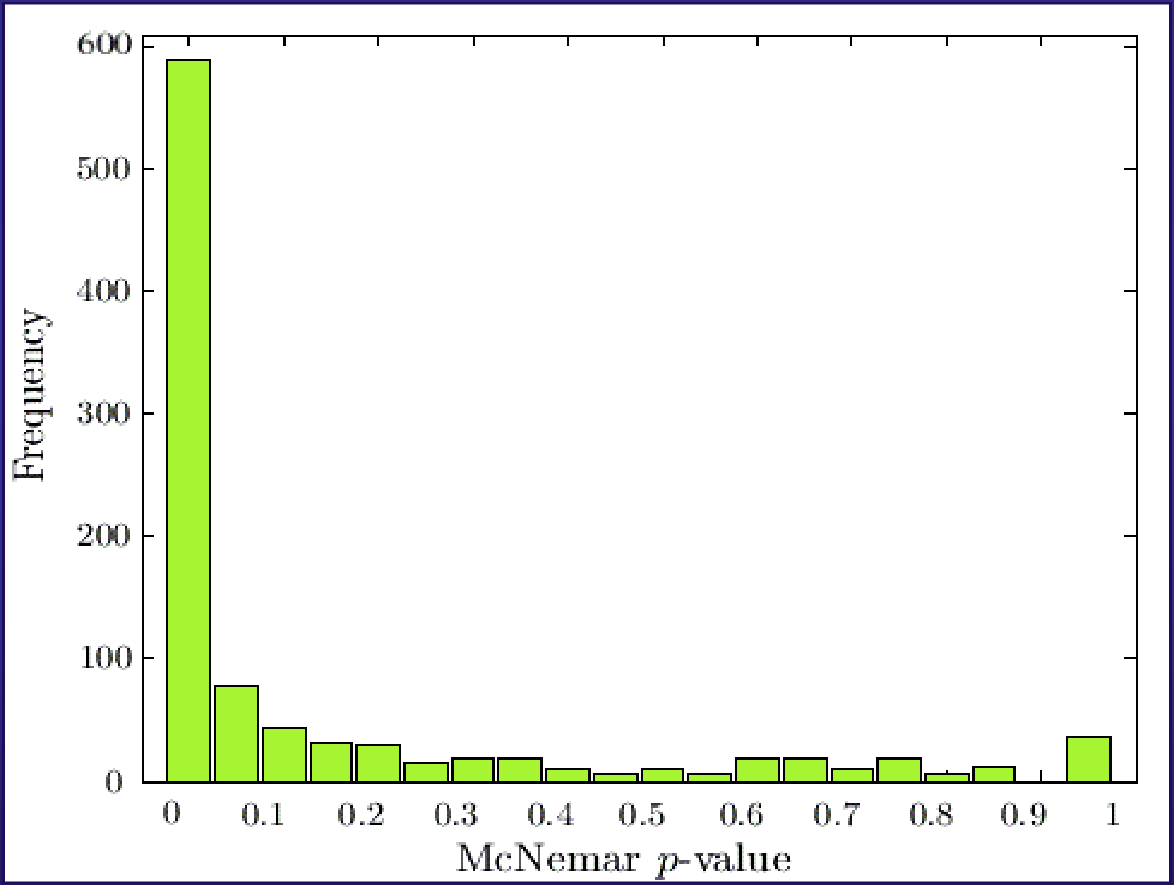
Results – McNemar's Test:

A pairwise test for ROC score difference:
the combined graph vs. individual graphs



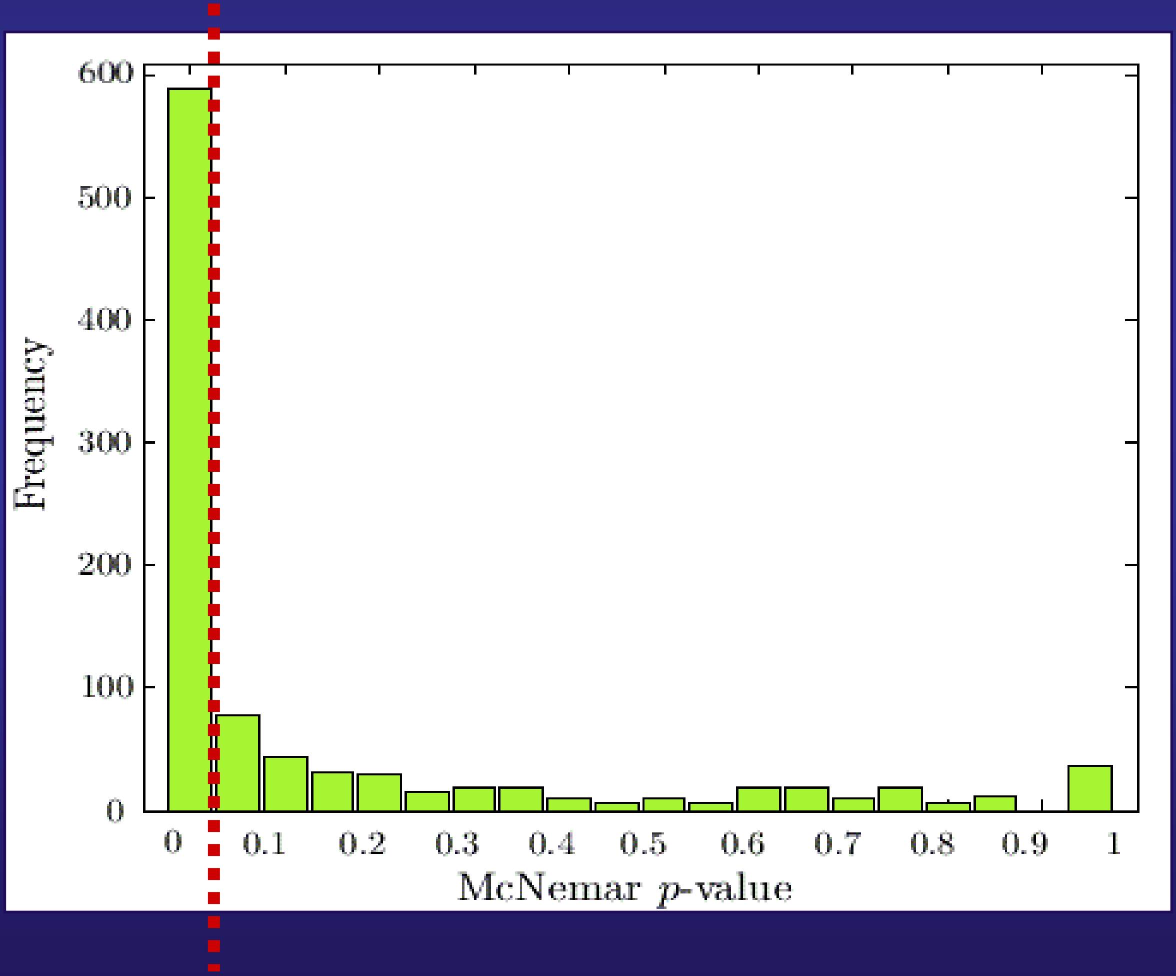
Results – McNemar's Test:

A smaller p-value indicates a more statistically significant difference

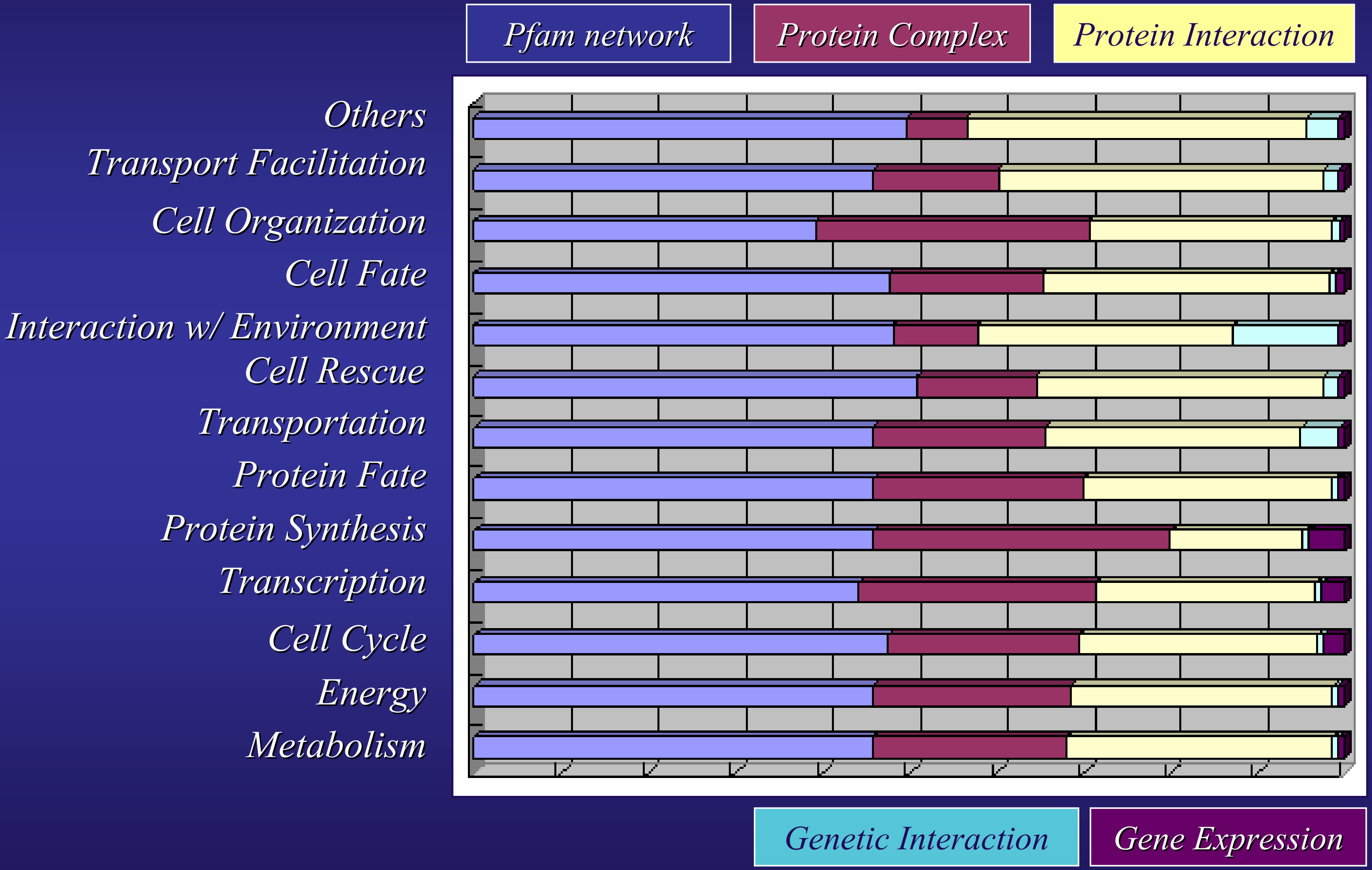


Results – McNemar's Test:

In 61% of the total number of trials, there is a statistically significant difference (at a significance level of alpha=0.05).

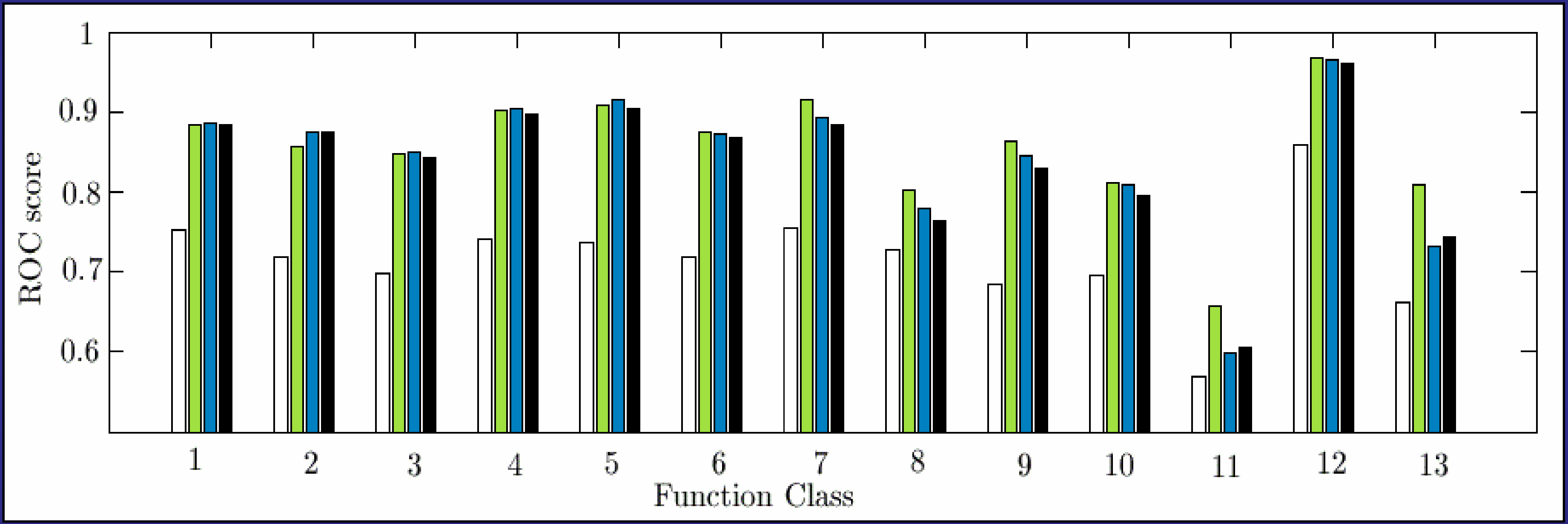


Results : *Obtained Weights*



Results : Comparison between Methods

White: MRF
Green: SDP/SVM
Blue: L_{fix}
Black: L_{opt}



*For most classes, the proposed method achieves high scores, which are similar to the SDP/SVM methods. In classes 11 and 13, the proposed method **performs poor** (but still better than the MRF method), However, taking into account the Simplicity and Efficiency the method shows the promising results*

Results : Computational Time

Average Computation Time

Combining Graphs		
	Fixed Weights :	*1.41 seconds (std. 0.013)*
	Optimized Weights :	*49.3 seconds (std. 14.8)*
SDP/SVM :		*Approx. Several CPU days* *(G. Lanckriet, personal communication)*

** Measured in a standard 2.2Ghz PC with 1GByte memory*

Results : Computational Time

Average Computation Time

Combining Graphs: *Nearly linearly proportional to the number of non-zero entries of sparse matrices*

SDP/SVM : $O(n^3) + O((m+n)^2 n^{2.5})$

Results : Summary

*Combining Graphs with Optimized Weights has **"MORE"***

Selectivity

When Compared with Combining Graphs with Fixed Weights

*Combining Graphs has **"MORE"***

Simplicity, Computational Efficiency, thus Scalablity

When Compared with SDP/SVM

Results : Summary

Combining Graphs with Optimized Weights is ***"LESS"***

Simple

When Compared with Combining Graphs with Fixed Weights

Combining Graphs is ***"LESS"***

Accurate

When Compared with SDP/SVM

Results : Summary

Semi-Supervised Learning with Multiple Networks

- *...is Fast and Scalable*
- *...provides Selectivity*
(redundant / irrelevant networks can be excluded)

For Further Information…

Multivariate Statistical Methods

R.A. Johnson & D.W. Wichern,
Applied multivariate statistical analysis,
Prentice-Hall. Inc, 1998.

B.F.J. Manly,
Multivariate statistical methods: A primer,
Chapman & Hall, 1997.

Kernel Methods and SVM

V. Vapnik,
Statistical learning theory,
Wiley, NY, 1998

For Further Information…

Kernel Methods and SVM

C.J.C. Burges,
A tutorial on support vector machines for pattern recognition,
Data Mining and Knowledge Discovery, 1998.

N. Cristianini and J. Shawe-Taylor,
An introduction to support vector machines,
Cambridge University Press, Cambridge, UK, 2000.

B. Scholkopf and A. J. Smola,
Learning with Kernels,
MIT press, MA, 2002.

For Further Information…

Kernel Methods and Bioinfomatics

B. Scholkopf, K. Tsuda and J-P. Vert,
Kernel Methods in Computational Biology,
MIT press, London, 2004.

Semi-supervised Learning

Olivier Chapelle, Bernhard Schoelkopf and Alexander Zien,
Semi-Supervised Learning,
MIT press, 2005

For Further Information…

Application I: Alternative Splicing

G. Rätsch, S. Sonnenburg and B. Schölkopf,
A RASE: Recognition of Alternatively Spliced Exons in C. elegans,
Bioinfomatics, 2004.
http://www.fml.tuebingen.mpg.de/raetsch/projects/RASE

Application II: Protein Function Classification

H. Shin and K. Tsuda,
Prediction of Protein Function from Networks,
In book: Semi-Supervised Learning, MIT press, London, 2006.
http://www.kyb.tuebingen.mpg.de/~shin
http://www.fml.tuebingen.mpg.de/~shin

K. Tsuda, H. Shin, and B. Schölkopf,
Fast Protein Classification with Multiple Networks,
Bioinformatics, 2005.