6.047 / 6.878 Computational Biology: Genomes, Networks, Evolution
Fall 2008

Classification

Two Different Approaches

- Generative
  - Bayesian Classification and Naïve Bayes
  - Example: Mitochondrial Protein Prediction
- Discriminative
  - Support Vector Machines
  - Example: Tumor Classification

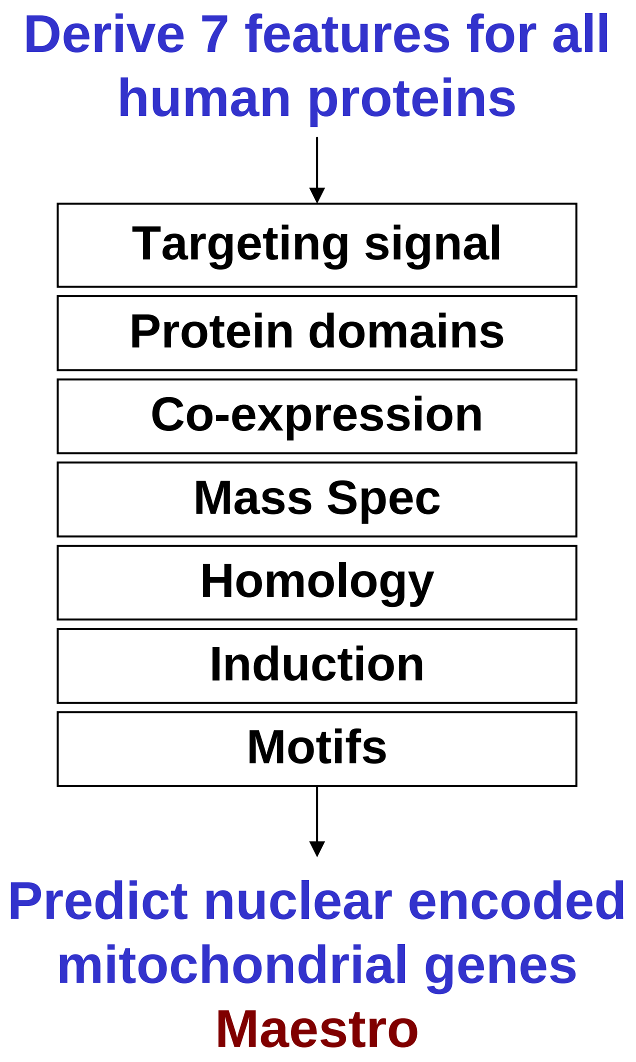
Bayesian Classification

We will pose the classification problem in probabilistic terms

Create models for how features are distributed for objects of different classes

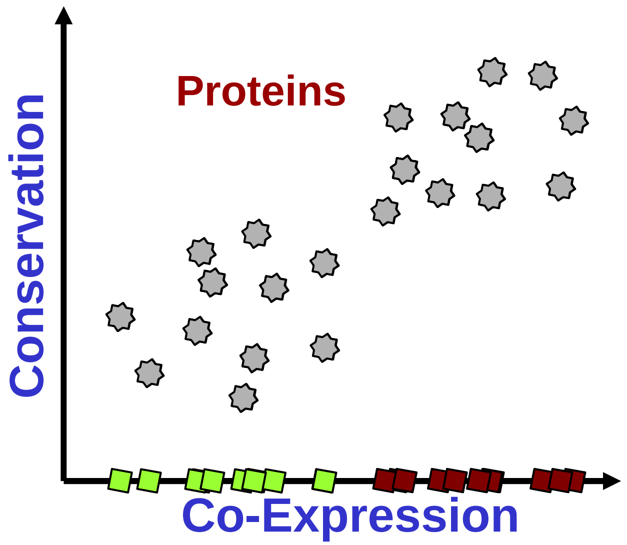
We will use probability calculus to make classification decisions

Classifying Mitochondrial Proteins


Derive 7 features for all human proteins
Targeting signal
Protein domains
Co-expression
Mass Spec
Homology
Induction
Motifs
Predict nuclear encoded mitochondrial genes
Maestro

Lets Look at Just One Feature

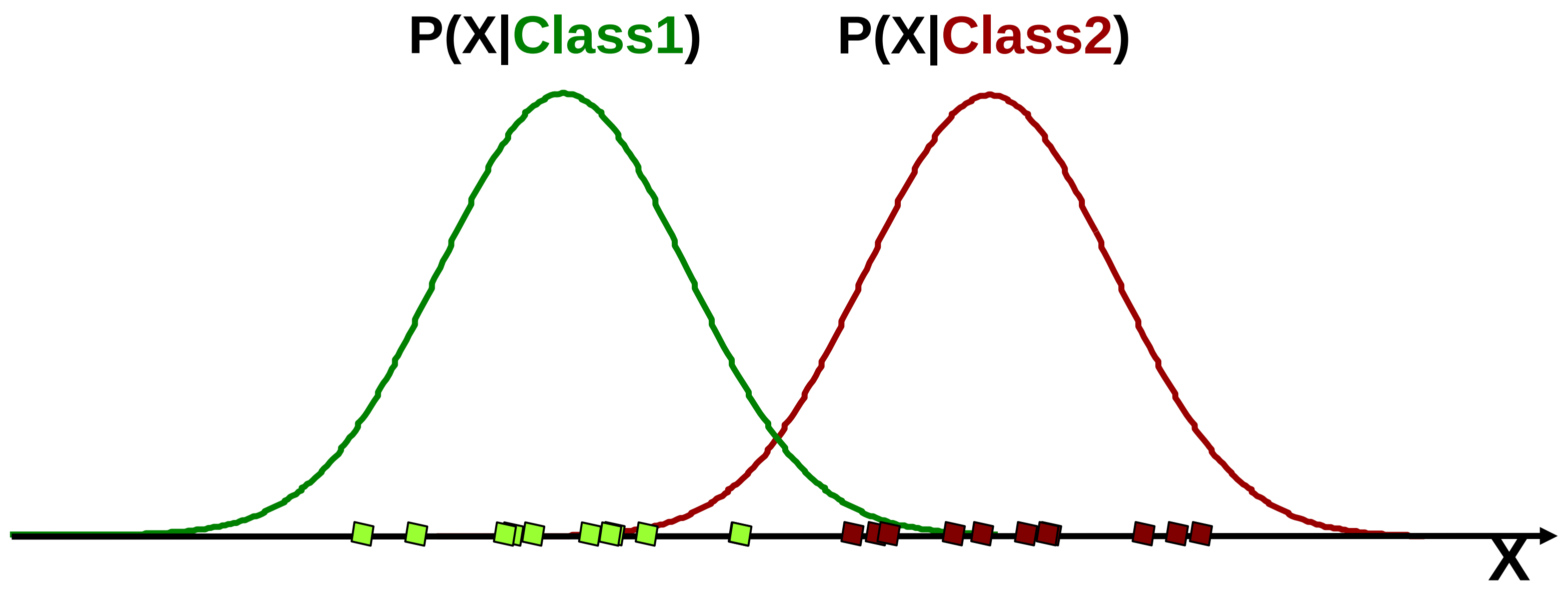
- Each object can be associated with multiple features
- We will look at the case of just one feature for now



We are going to define two key

The First Key Concept

Features for each class drawn from class-conditional probability distributions (CCPD)



Our first goal will be to *model* these distributions

The Second Key Concept

We model prior probabilities to quantify the expected *a priori* chance of seeing a class

P(Class2) & P(Class1)

P(mito) = how likely is the next protein to be a mitochondrial protein *before I see any features to help me decide*

We expect ~1500 mitochondrial genes out of ~21000 total, so

P(mito)=1500/21000
P(~mito)=19500/21000

But How Do We Classify?

- **So we have priors defining the *a priori* probability of a class**

P(Class1), P(Class2)

- **We also have models for the probability of a feature given each class**

P(X|Class1), P(X|Class2)

But we want the probability of the class given a feature
***How do we get* P(Class1|X)?**

Bayes Rule

Evaluate evidence

Belief before evidence

$$P(Class \mid Feature) = \frac{P(Feature \mid Class)P(Class)}{P(Feature)}$$

Belief after evidence

Evidence

Bayes, Thomas (1763) An essay towards solving a problem in the doctrine of chances. Philosophical Transactions of the Royal Society of London, **53:370-418**

Bayes Decision Rule

If we observe an object with feature X, how do decide if the object is from Class 1?

The Bayes Decision Rule is simply choose Class1 if:

$$P(Class1 \mid X) > P(Class2 \mid X)$$

$$\frac{P(X \mid Class1)P(L1)}{\underset{\nwarrow}{P(X)}} > \frac{P(X \mid Class2)P(L2)}{\underset{\nearrow}{P(X)}}$$

This is the same number on both sides!

Discriminant Function

We can create a convenient representation of the Bayes Decision Rule

$$P(X \mid Class1)P(Class1) > P(X \mid Class2)P(Class2)$$

$$\frac{P(X \mid Class1)P(Class1)}{P(X \mid Class2)P(Class2)} > 1$$

$$\boxed{G(X) = \log \frac{P(X \mid Class1)}{P(X \mid Class2)} \frac{P(Class1)}{P(Class2)} > 0}$$

If G(X) > 0, we classify as Class 1

Stepping back

What do we have so far?

We have defined the two components, class-conditional distributions and priors

P(X|Class1), P(X|Class2)   P(Class1), P(Class2)

We have used Bayes Rule to create a discriminant function for classification from these components

$$G(X) = \log \frac{P(X \mid Class1)}{P(X \mid Class2)} \frac{P(Class1)}{P(Class2)} > 0$$

Given a new feature, X, we plug it into this equation…

…and if G(X)> 0 we classify as Class1

Two Fundamental Tasks

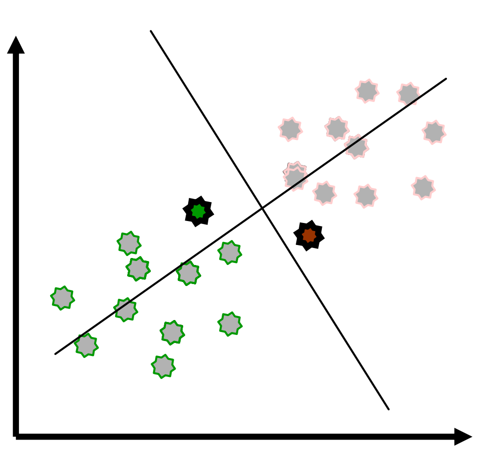
- We need to estimate the needed probability distributions
  - P(X|Mito) and P(x|~Mito)
  - P(Mito) and P(~Mito)

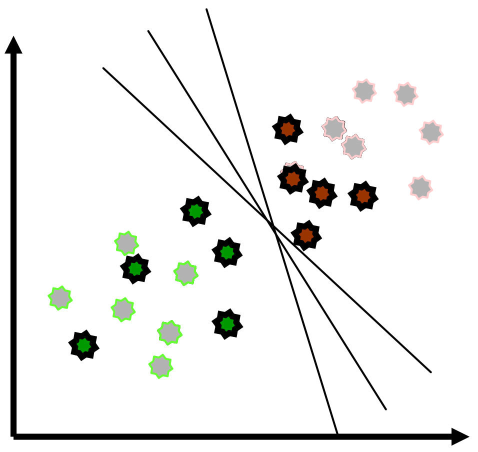
- We need to assess the accuracy of the classifier
  - How well does it classify new objects

The All Important Training Set

Building a classifier requires a set of labeled data points called the Training Set

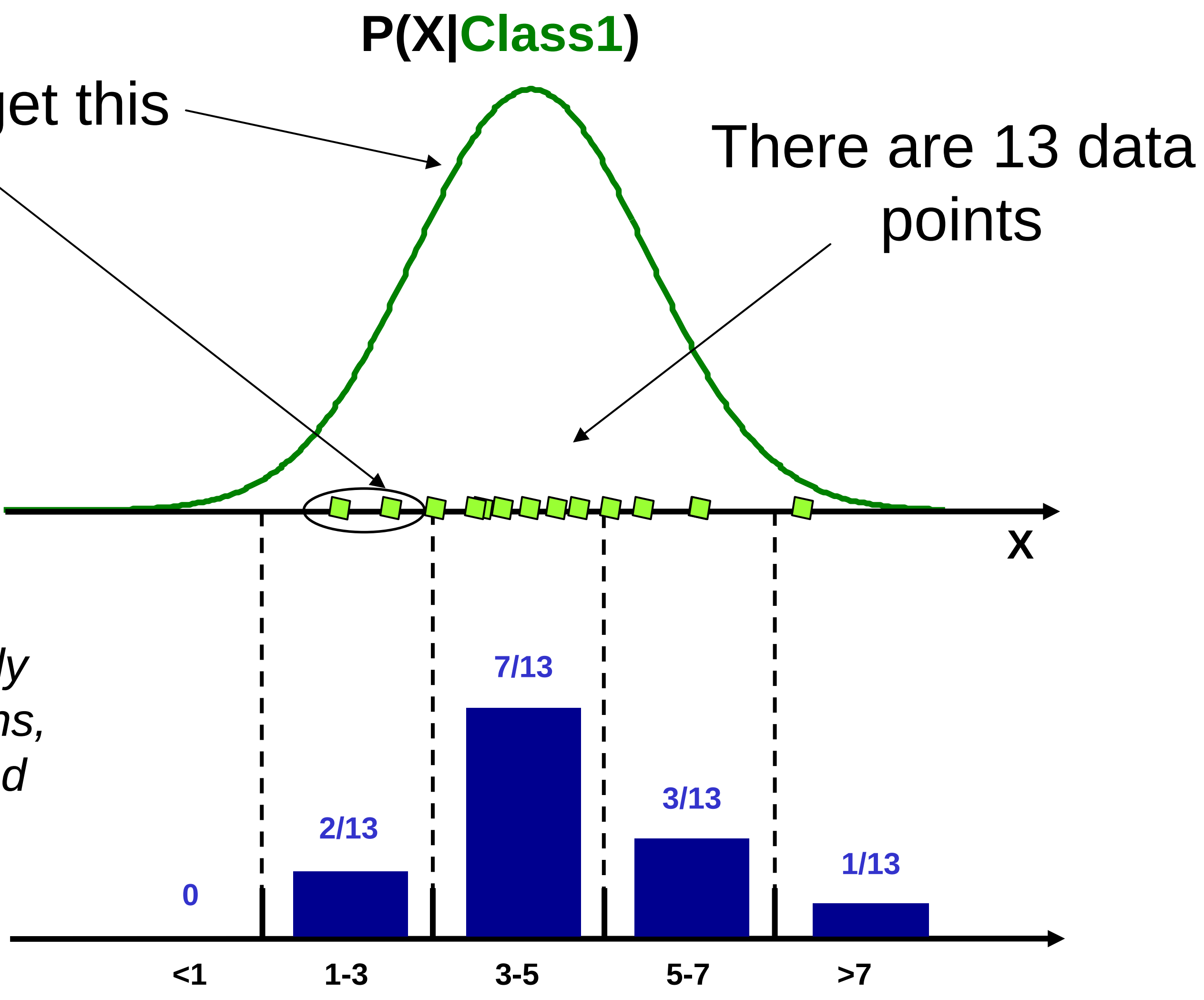
The quality of the classifier depends on the number of training set data points





How many data points you need depends on the problem
Need to build and test your classifier

Getting P(X|Class) from Training Set



One Simple Approach

Divide X values into bins

And then we simply count frequencies

In general, and especially for continuous distributions, this can be a complicated problem

Density Estimation

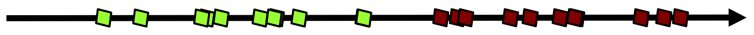
Getting Priors

Three general approaches

1. Estimate priors by counting fraction of classes in training set

P(Class1)=13/23

P(Class2)=10/23



13 Class1 **10 Class2**

2. Estimate from "expert" knowledge

Example
P(mito)=1500/21000
P(~mito)=19500/21000

3. We have no idea – use equal (uninformative) priors

P(Class1)=P(Class2)

But sometimes fractions in training set are not representative of world

We Are Just About There….

We have created the class-conditional distributions and priors

P(X|Class1), P(X|Class2) P(Class1), P(Class2)

And we are ready to plug these into our discriminant function

$$G(X) = \log \frac{P(X \mid Class1)}{P(X \mid Class2)} \frac{P(Class1)}{P(Class2)} > 0$$

But there is one more little complication…..

But What About Multiple Features?

- We have focused on a single feature for an object
- But mitochondrial protein prediction (for example) has 7 features

Targeting signal
Protein domains
Co-expression
Mass Spec
Homology
Induction
Motifs

So P(X|Class) become P(X1,X2,X3,…,X8|Class) and our discriminant function becomes

$$G(X) = \log \frac{P(X_1, X_2, \ldots, X_7 \mid Class1)}{P(X_1, X_2, \ldots, X_7 \mid Class2)} \frac{P(Class1)}{P(Class2)} > 0$$

Distributions Over Many Features

Estimating P(X1,X2,X3,…,X8|Class1) can be difficult

- Assume each feature binned into 5 possible values
- We have 5^8 combinations of values we need to count the frequency for

- Generally will not have enough data
  - We will have lots of nasty zeros

Naïve Bayes Classifier

We are going to make the following assumption:

All features are independent given the class

$$P(X_1, X_2, \ldots, X_n \mid Class) = P(X_1 \mid Class) P(X_2 \mid Class) \ldots P(X_n \mid Class)$$

$$= \prod_{i=1}^{n} P(X_i \mid Class)$$

We can thus estimate <u>individual distributions</u> for each feature and just <u>multiply</u> them together!

Naïve Bayes Discriminant Function

Thus, with the Naïve Bayes assumption, we can now rewrite, this:

$$G(X_1,\ldots,X_7) = \log \frac{P(X_1, X_2, \ldots, X_7 \mid Class1)}{P(X_1, X_2, \ldots, X_7 \mid Class2)} \frac{P(Class1)}{P(Class2)} > 0$$

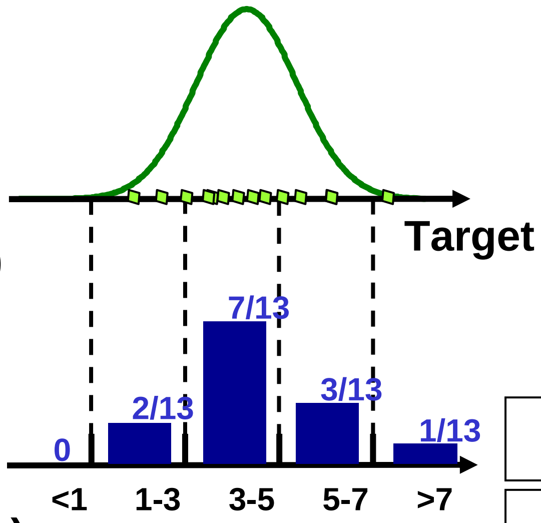
As this:

$$\boxed{G(X_1,\ldots,X_7) = \log \frac{\prod P(X_i \mid Class1)}{\prod P(X_i \mid Class2)} \frac{P(Class1)}{P(Class2)} > 0}$$

Individual Feature Distributions

Instead of a single big distribution, we have a smaller one for each feature (and class)

P(Target|Mito)	**P(Target|~Mito)**
P(Domain|Mito)	**P(Domain|~Mito)**
P(CE|Mito)	**P(CE|~Mito)**
P(Mass|Mito)	**P(Mass|~Mito)**
P(Homology|Mito)	**P(Homology|~Mito)**
P(Induc|Mito)	**P(Induc|~Mito)**
P(Motif|Mito)	**P(Motif|~Mito)**



Targeting signal

Protein domains

Co-expression

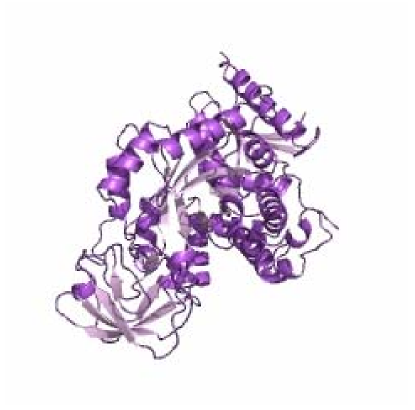
Mass Spec

Homology

Induction

Motifs

Classifying A New Protein



- Targeting signal
- Protein domains
- Co-expression
- Mass Spec
- Homology
- Induction
- Motifs

X_i

$P(X_i|\text{Mito})$

$P(X_i|\sim\text{Mito})$

(for all 8 features)

Plug these and priors into the discriminant function

$$G(X_1,\ldots,X_7) = \log \frac{\prod P(X_i \mid Mito)}{\prod P(X_i \mid \sim Mito)} \frac{P(Mito)}{P(\sim Mito)} > 0$$

IF G>0, we predict that the protein is from class Mito

Maestro Results

Apply Maestro to Human Proteome

Total predictions: 1,451 proteins

490 novel predictions

Slide Credit: S. Calvo

How Good is the Classifier?

The Rule

We *must* test our classifier on a different set from the training set: the labeled test set

The Task

We will classify each object in the test set and count the number of each type of error

Binary Classification Errors

	True (Mito)	False (~Mito)
Predicted True	🟩	🟥
Predicted False	🟨	🟫

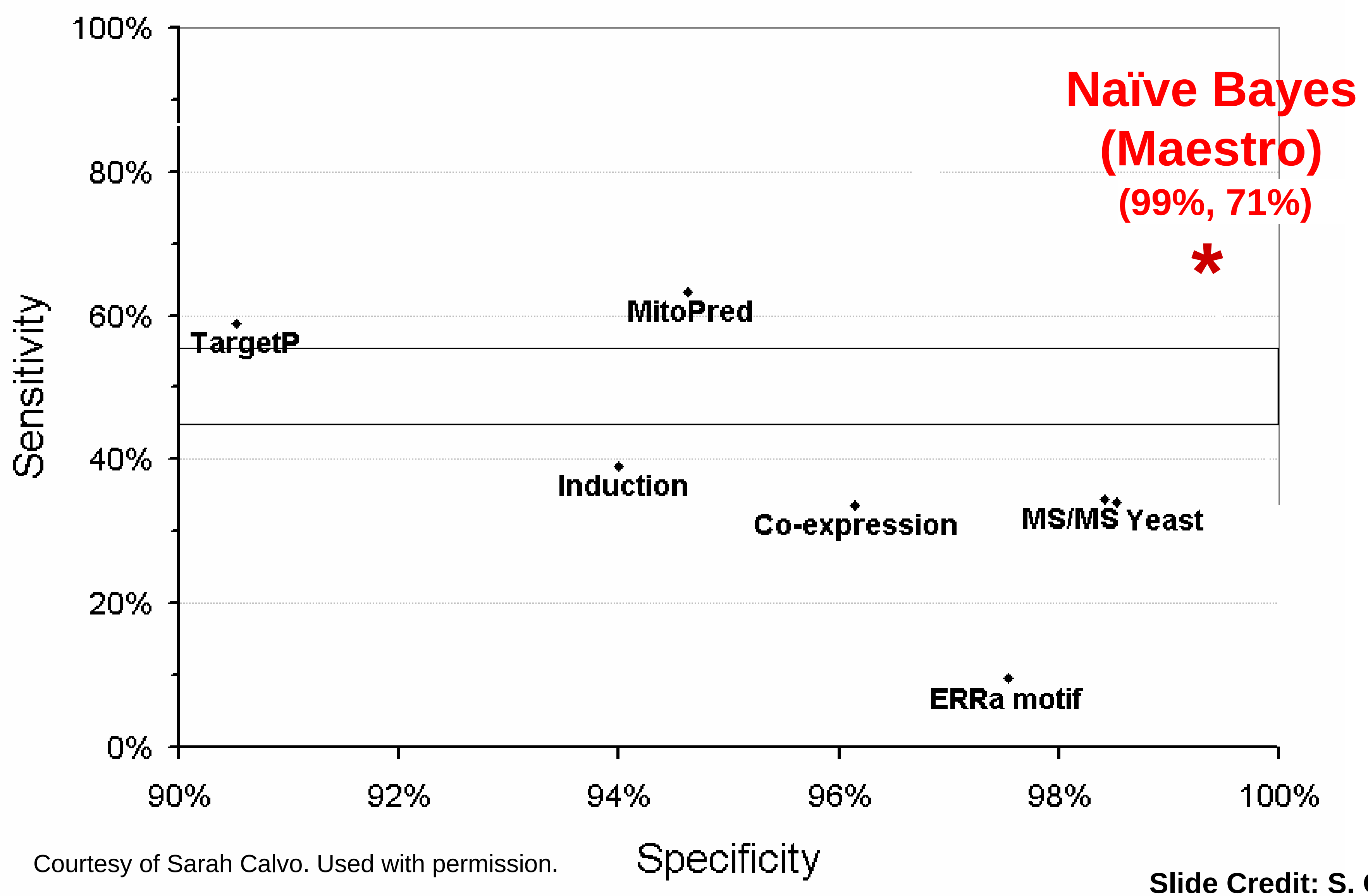
Sensitivity = TP/(TP+FN) **Specificity = TN/(TN+FP)**

- Sensitivity
  - Fraction of all Class1 (True) that we correctly predicted at Class 1
  - *How good are we at finding what we are looking for*

- Specificity
  - Fraction of all Class 2 (False) called Class 2
  - *How many of the Class 2 do we filter out of our Class 1 predictions*

In both cases, the higher the better

Maestro Outperforms Existing Classifiers



Slide Credit: S. Calvo

Support Vector Machines

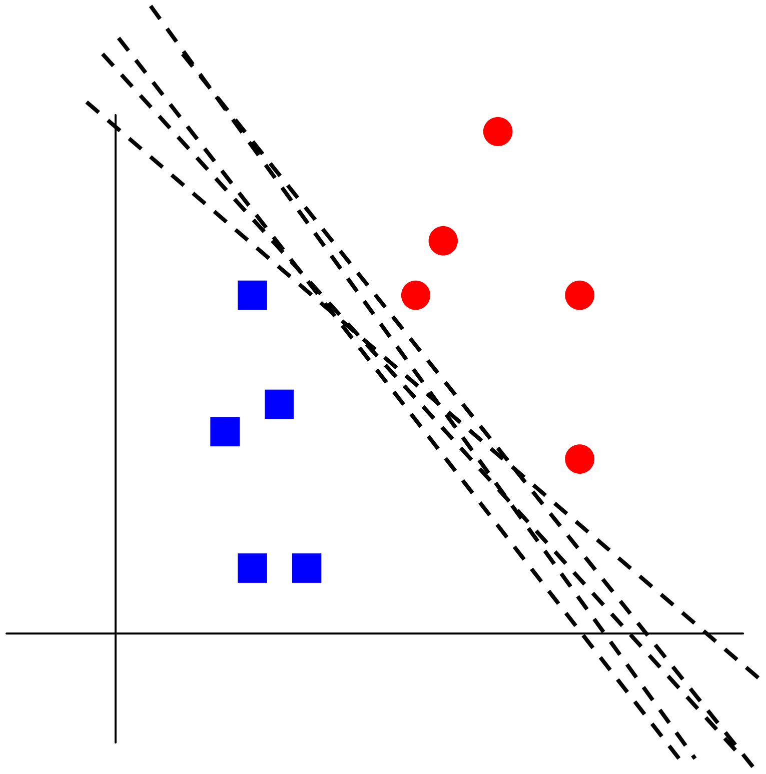
Discriminative Classification

Support Vector Machines (SVMs)

Easy to select a line

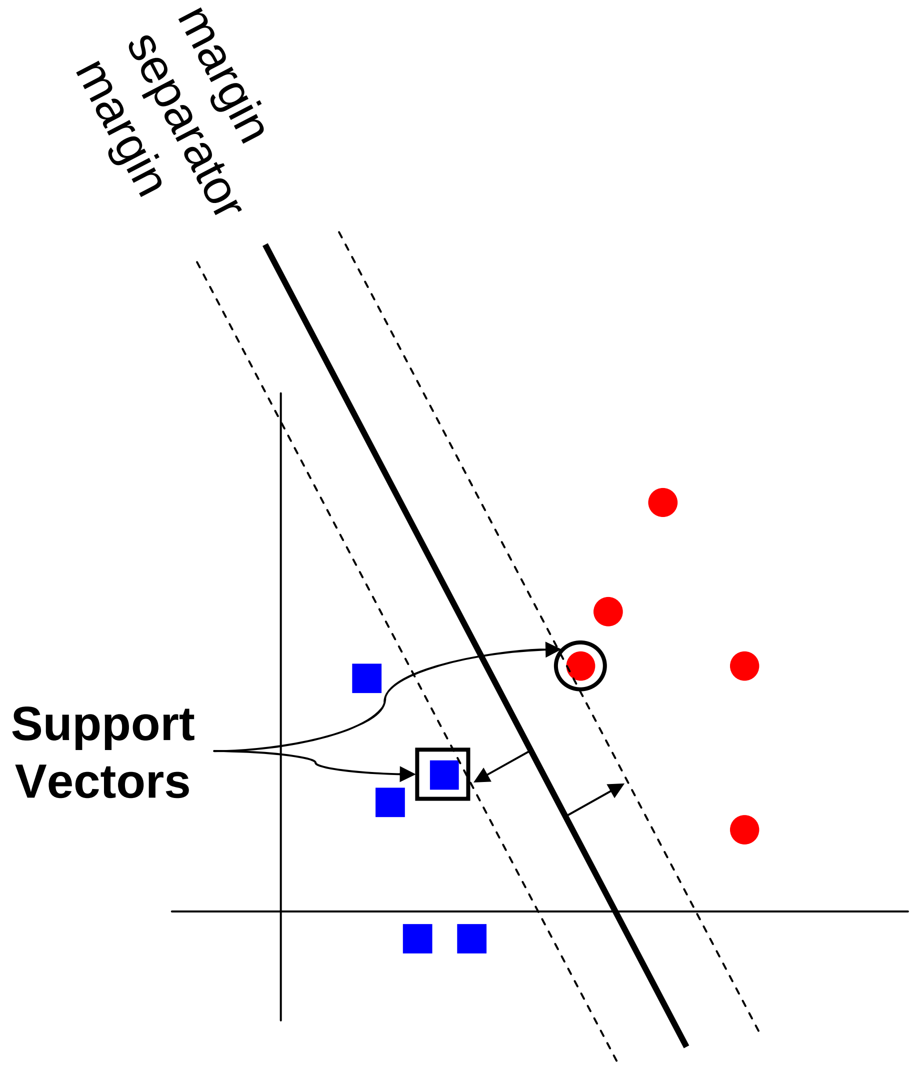
But many lines will separate these training data

What line should we choose?



Support Vector Machines (SVMs)

A sensible choice is to select a line that maximizes the *margin* between classes



SVM Formulation

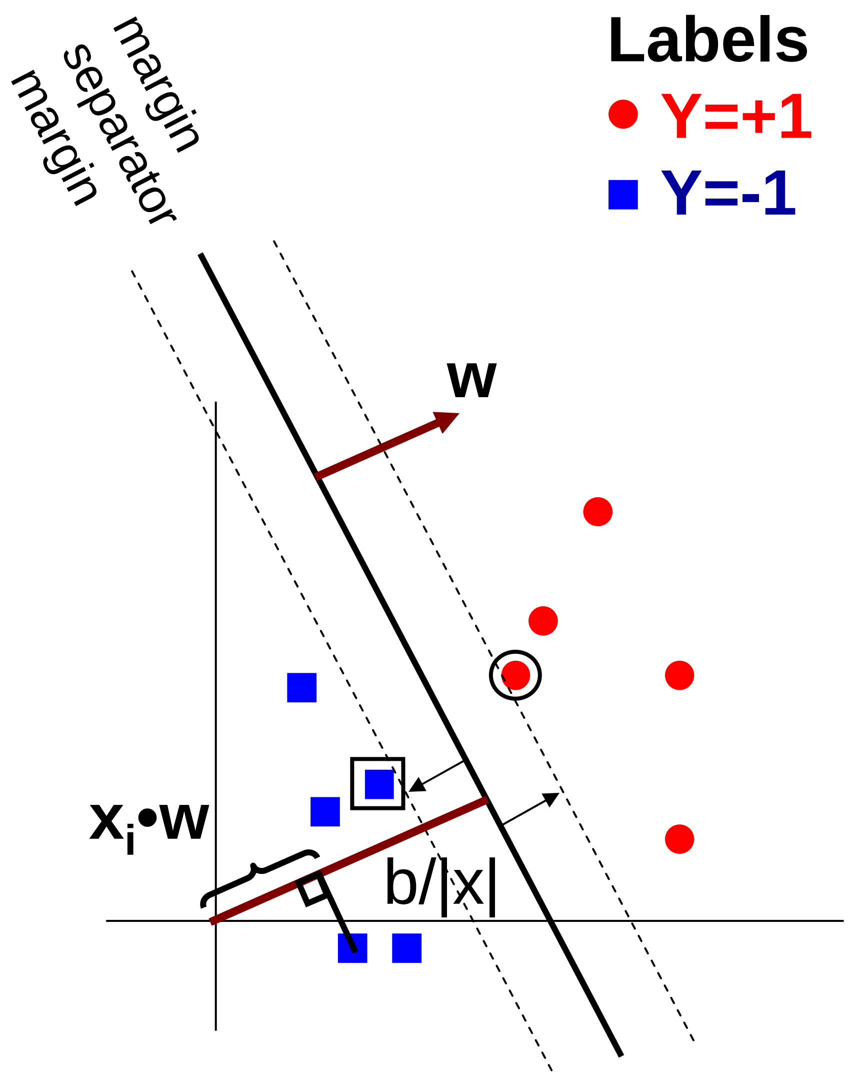
We define a vector **w** normal to the separating line

Assume all data satisfy the following:

$$\mathbf{x_i} \bullet \mathbf{w} - b \geq +1 \quad \text{for } y_i = +1$$

$$\mathbf{x_i} \bullet \mathbf{w} - b \leq -1 \quad \text{for } y_i = -1$$

$$y_i \left(\mathbf{x_i} \bullet \mathbf{w} - b \geq 1 \right)$$



Labels

● Y=+1

■ Y=-1

An Optimization Problem

Only need dot product of input data!

For full derivation, see Burges (1998)

$$\text{Minimize } L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \boxed{\mathbf{x_i} \bullet \mathbf{x_j}}$$

Quadratic Programming

$$\text{subject to } \sum_i \alpha_i y_i = 0 \text{ and } \alpha_j > 0$$

Solve for α

$$\alpha_i \left(y_i \left(\mathbf{x_i} \bullet \mathbf{w} - b \right) - 1 \right) = 0$$

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x_i}$$

Only some α_i are non-zero

x$_i$ with a$_i$ >0 are the *support vectors*

w is *determined by these data points!*

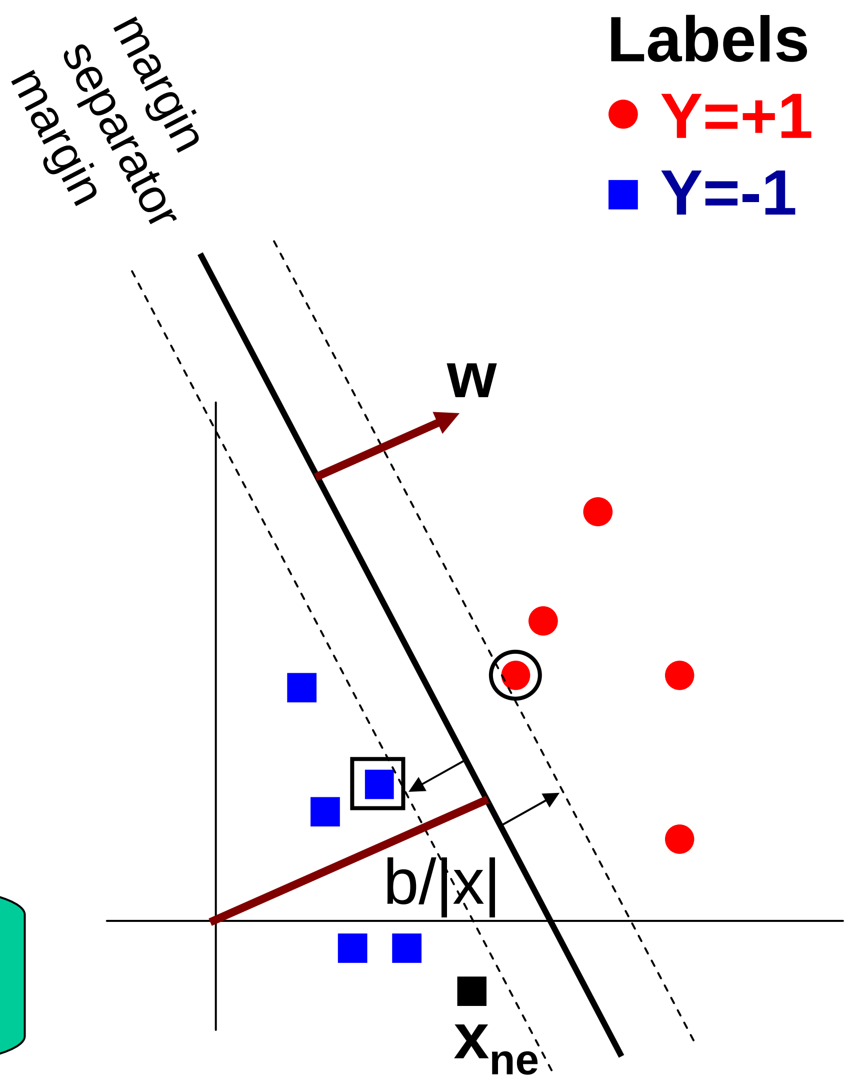
Using an SVM

Given a new data point we simply assign it the label:

$$y_i = \text{sign}\left(\mathbf{w} \bullet \mathbf{x}_{\text{new}}\right)$$

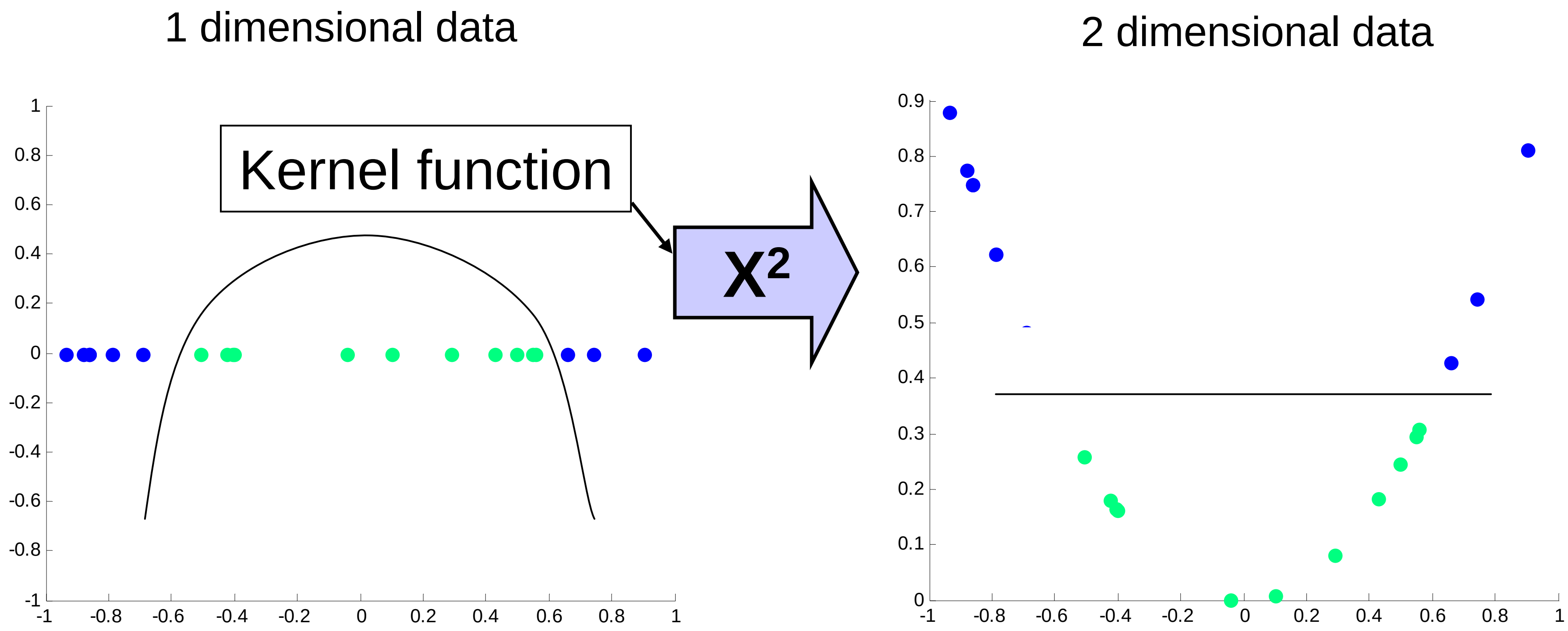
$$= \text{sign}\left(\sum_{\text{i}} \alpha_i y_i \boxed{\mathbf{x_i} \bullet \mathbf{x}_{\text{new}}}\right)$$

Again, only dot product of input data!



Non-linear Classifier

- **Some data not linearly separable in low dimensions**
- **What if we transform it to a higher dimension?**



Kernel Mapping

Want a **mapping** from input space
to other euclidean space

$\Phi(x): R^d \rightarrow H$

But $\Phi(X)$ can be a mapping to an infinite dimensional space
i.e. d points become an infinite number of points

$$\mathbf{X=(x_1,x_2)} \Rightarrow \mathbf{\Phi(X)=(\phi_1,\phi_2,\phi_3,\ldots.\phi_\infty)}$$

Rather difficult to work with!

Kernel Mapping

Want a **mapping** from input space to other euclidean space

From previous slide, SVMs *only depend* on **dot product**

$\Phi(x): R^d \rightarrow H$

$\mathbf{X_i \bullet X_j} \xrightarrow{\textbf{becomes}} \Phi(\mathbf{X_i}) \bullet \Phi(\mathbf{X_j})$

Here is trick: if we have a kernel function such that

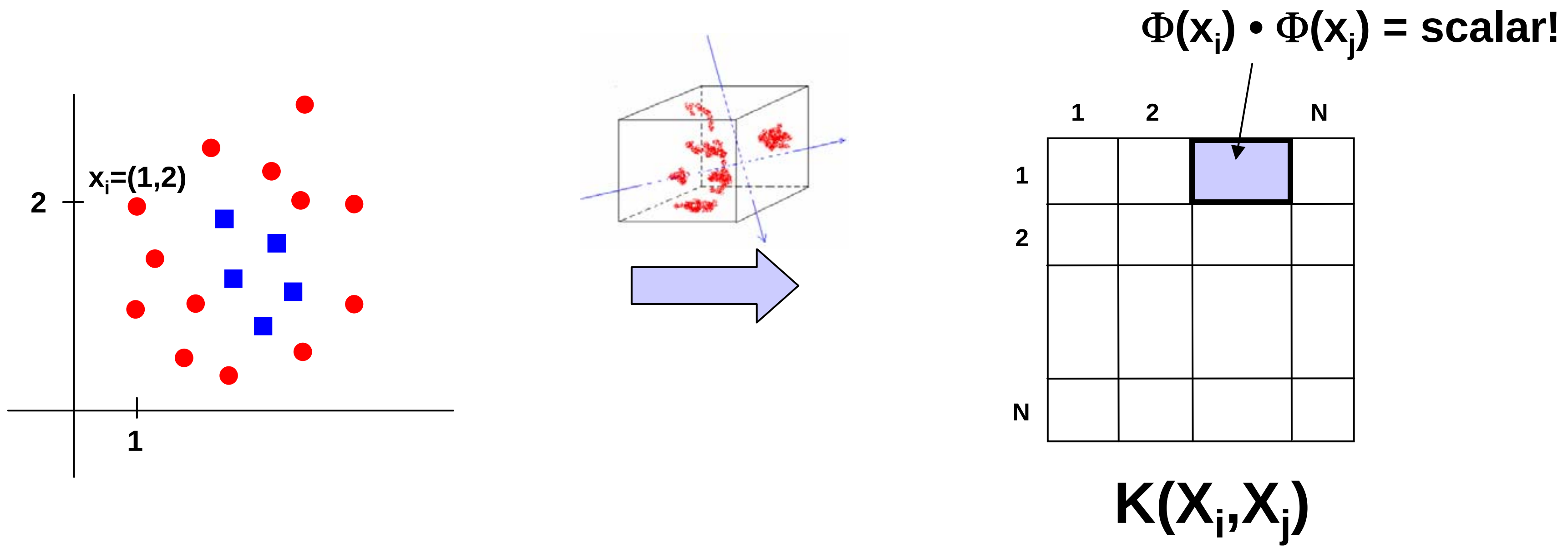
$$\mathbf{K(X_i,X_j)} = \Phi(\mathbf{X_i}) \bullet \Phi(\mathbf{X_j})$$

We can just use K and never know $\Phi(x)$ explicitly!

$\Phi(X)$ is high dimensional
K is a scalar

Kernels

So the key step is to take your input data and transform it into a kernel matrix



We have then done two very useful things:

1. Transformed X into a high (possibly infinite) dimensional space (where we hope are data are separable)
2. Taken dot products in this space to create scalars

Example Kernels

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$$ **Linear**

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i^T \mathbf{x}_j + r)^d$$ **Polynomial**

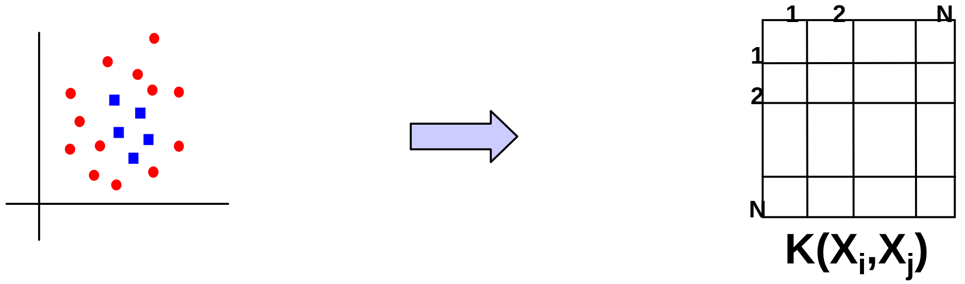
$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \left\| \mathbf{x}_i - \mathbf{x}_j \right\|^2\right)$$ **Radial Basis Function**

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \mathbf{x}_i^T \mathbf{x}_j + r)$$ **Sigmoid**

What K(X_i,X_j) are valid kernels?
Answer given by Mercer's Condition (see Burgess 1998)

Using (Non-Linear) SVMs

Step 1 – Transform data to Kernel Matrix K



Step 2 – Train SVM on transformed data – get support vectors

$$\text{Minimize } L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x_i} \bullet \mathbf{x_j} = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j K\left(\mathbf{x_i}, \mathbf{x_j}\right)$$

Step 2 – Test/Classify on new samples

$$y_{new} = \text{sign}\left(\mathbf{w} \bullet \mathbf{x}_{\text{new}}\right) = \text{sign}\left(\sum_{\text{i}} \alpha_i y_i \mathbf{x_i} \bullet \mathbf{x}_{\text{new}}\right) = \text{sign}\left(\sum_{\text{i}} \alpha_i y_i K\left(\mathbf{x_i}, \mathbf{x}_{\text{new}}\right)\right)$$

Classifying Tumors with Array Data

- Primary samples:
  - 38 bone marrow samples
  - 27 ALL, 11 AML
  - obtained from acute leukemia patients at the time of diagnosis;
- Independent samples:
  - 34 leukemia samples
  - 24 bone marrow
  - 10 peripheral blood samples
- Assay ~6800 Genes

Weighted Voting Classfication

General approach of Golub et al (1999) paper:

- Choosing a set of informative genes based on their correlation with the class distinction
- Each informative gene casts a weighted vote for one of the classes
- Summing up the votes to determine the winning class and the prediction strength

Results

Initial Samples

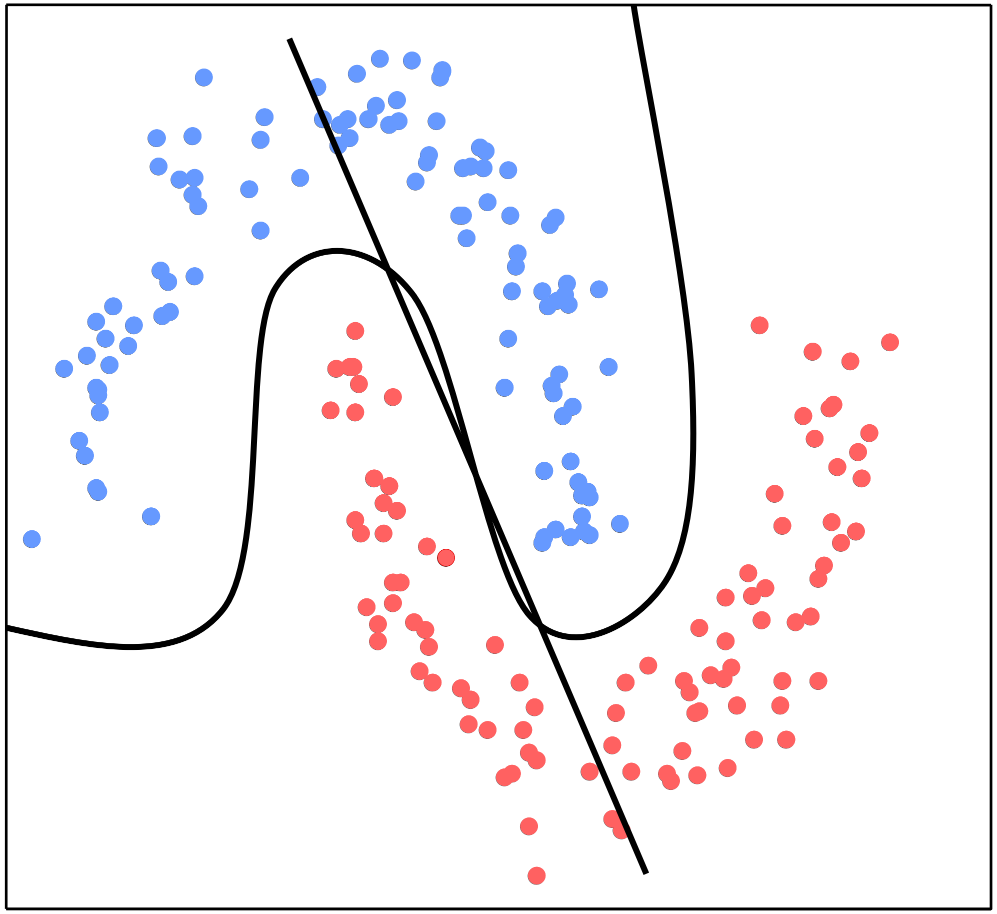
- 36 of the 38 samples as either AML or ALL. All 36 samples agree with clinical diagnosis
- 2 not predicted

Independent Samples

- 29 of 34 samples are strongly predicted with 100% accuracy.
- 5 not predicted

Bringing Clustering and Classification Together

Semi-Supervised Learning



Common Scenario

- Few labeled
- Many unlabeled
- Structured data

What if we cluster first?

Then clusters can help us classify