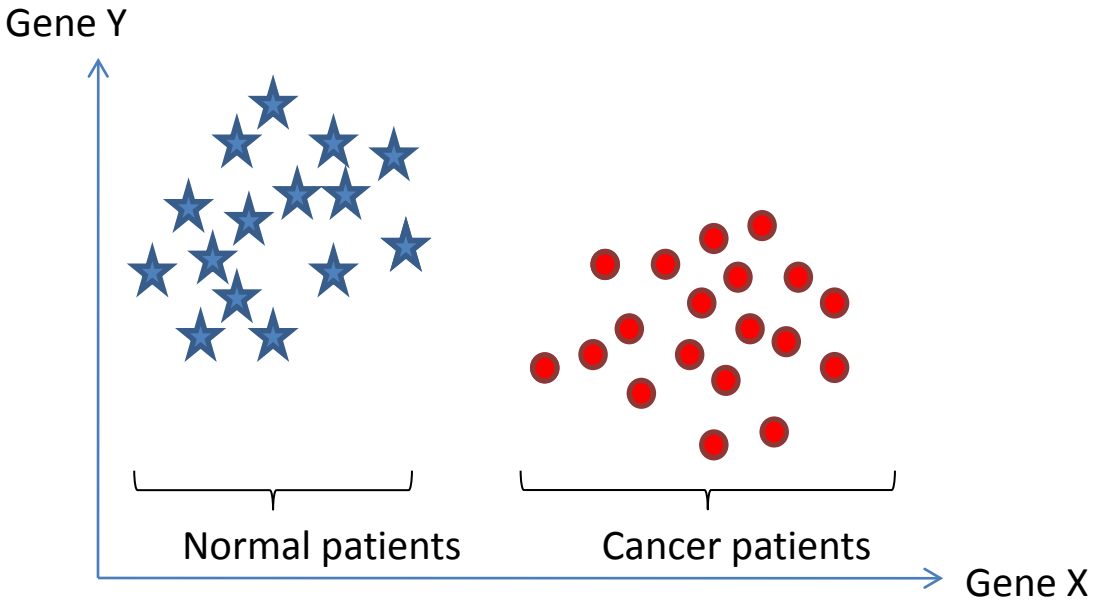
The Support Vector Machine (SVM) approach

- Support vector machines (SVMs) is a binary classification algorithm that offers a solution to problem #1.
- Extensions of the basic SVM algorithm can be applied to solve problems #1-#5.
- SVMs are important because of (a) theoretical reasons:
  - Robust to very large number of variables and small samples
  - Can learn both simple and highly complex classification models
  - Employ sophisticated mathematical principles to avoid overfitting

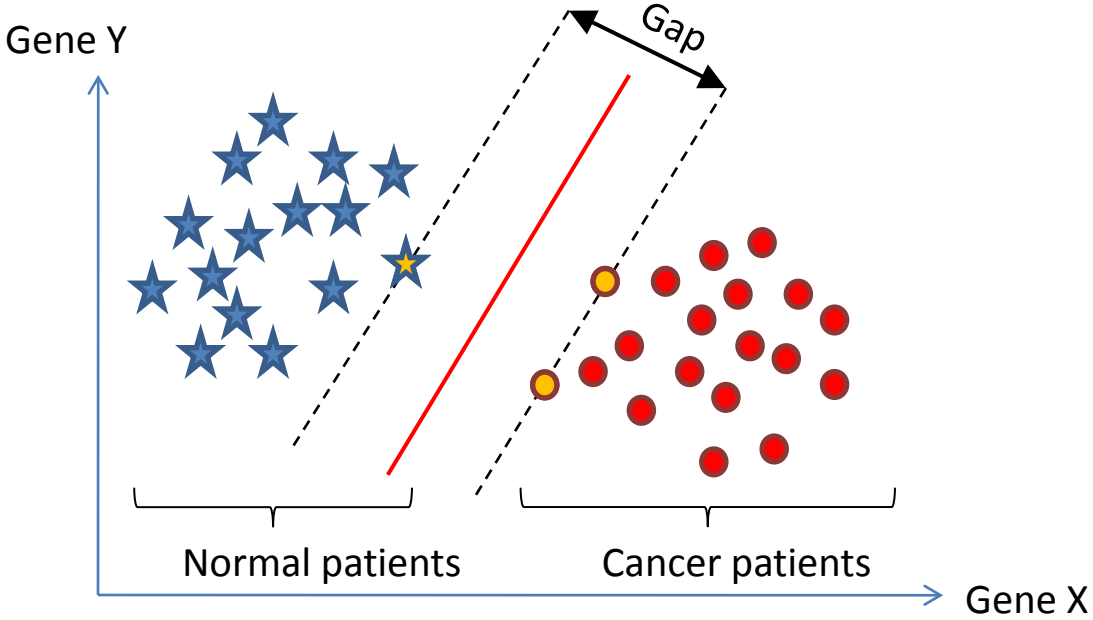
  and (b) superior empirical results.

Main ideas of SVMs



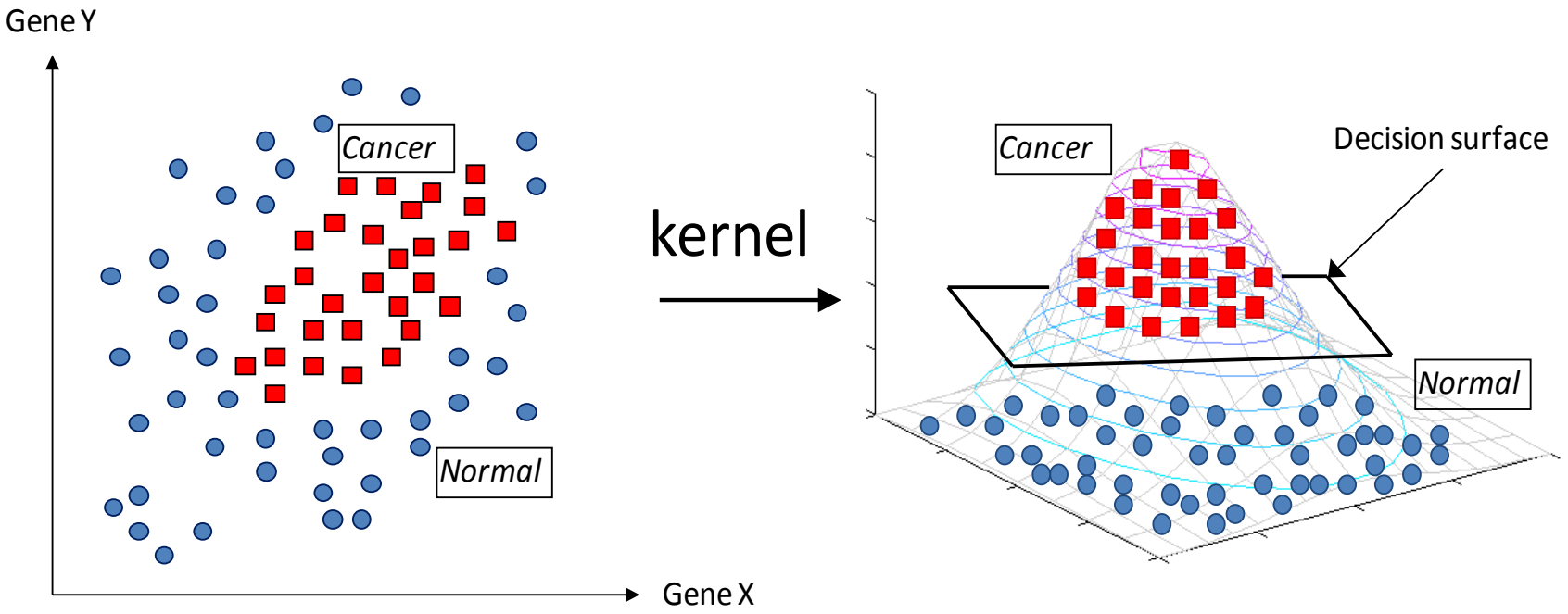
- Consider example dataset described by 2 genes, gene X and gene Y
- Represent patients geometrically (by “vectors”)

Main ideas of SVMs



- Find a linear decision surface ("hyperplane") that can separate patient classes and has the largest distance (i.e., largest "gap" or "margin") between border-line patients (i.e., "support vectors");

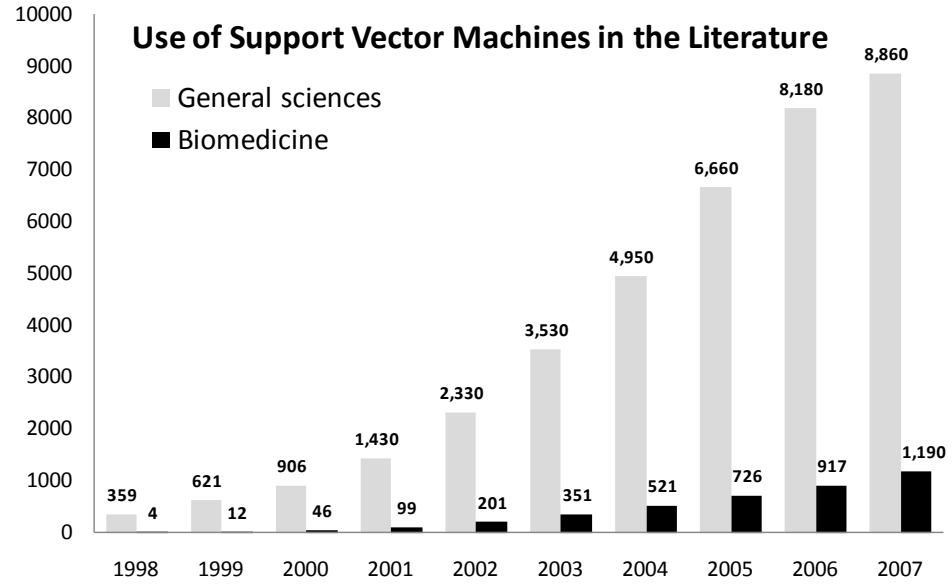
Main ideas of SVMs

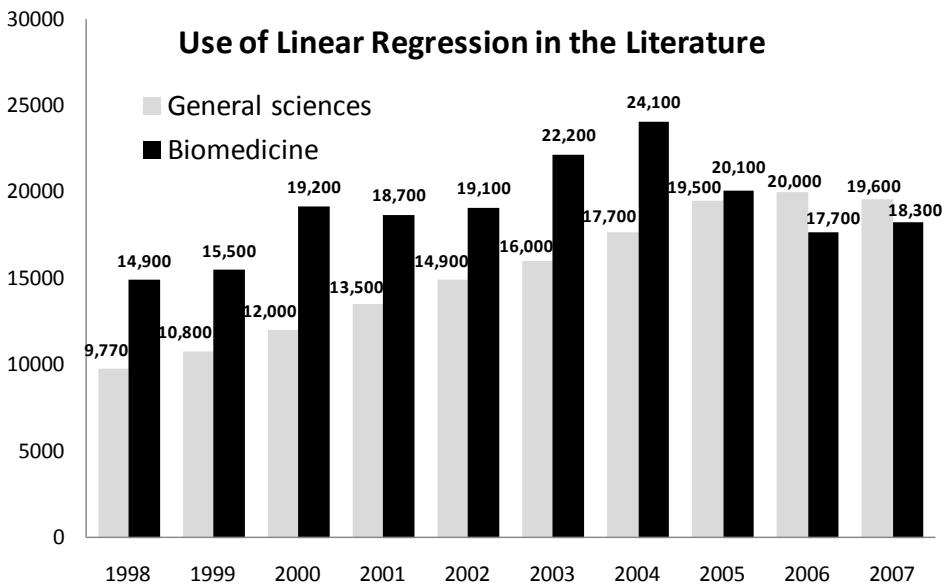


- If such linear decision surface does not exist, the data is mapped into a much higher dimensional space ("feature space") where the separating decision surface is found;
- The feature space is constructed via very clever mathematical projection ("kernel trick").

History of SVMs and usage in the literature

- Support vector machine classifiers have a long history of development starting from the 1960's.
- The most important milestone for development of modern SVMs is the 1992 paper by Boser, Guyon, and Vapnik ("*A training algorithm for optimal margin classifiers*")



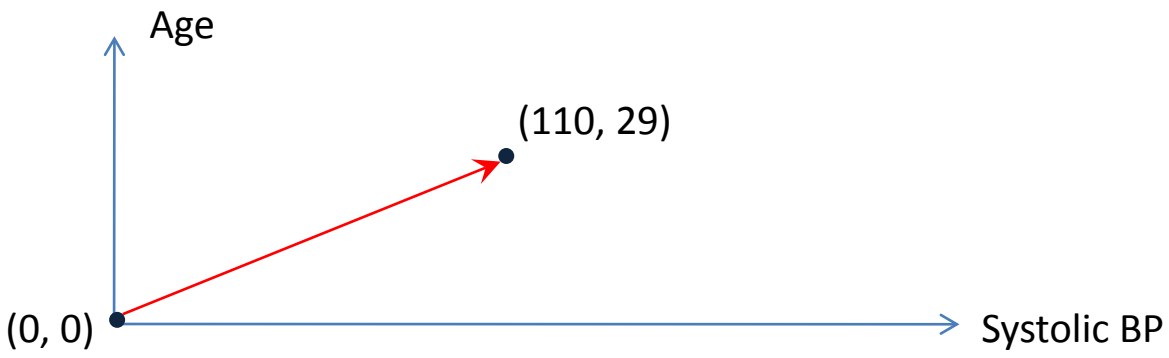


How to represent samples geometrically? Vectors in n-dimensional space ($\mathbb{R}^n$)

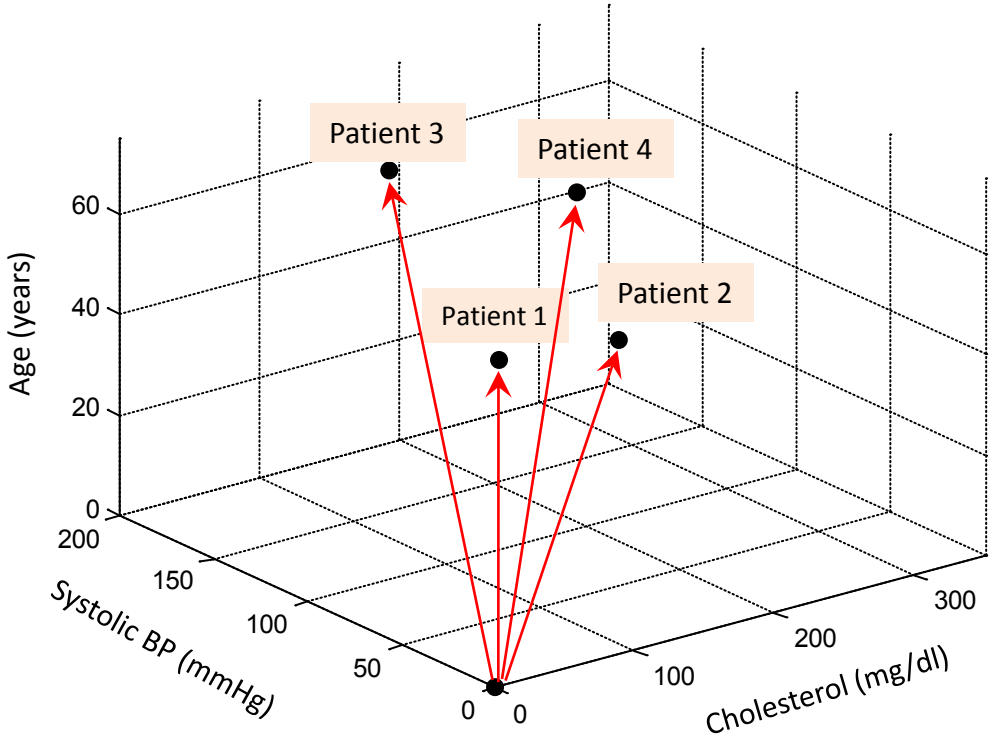
- Assume that a sample/patient is described by *n* characteristics ("features" or "variables")
- **Representation:** Every sample/patient is a vector in $\mathbb{R}^n$ with tail at point with 0 coordinates and arrow-head at point with the feature values.
- **Example:** Consider a patient described by 2 features:

  Systolic BP = 110 and *Age* = 29.

  This patient can be represented as a vector in $\mathbb{R}^2$:

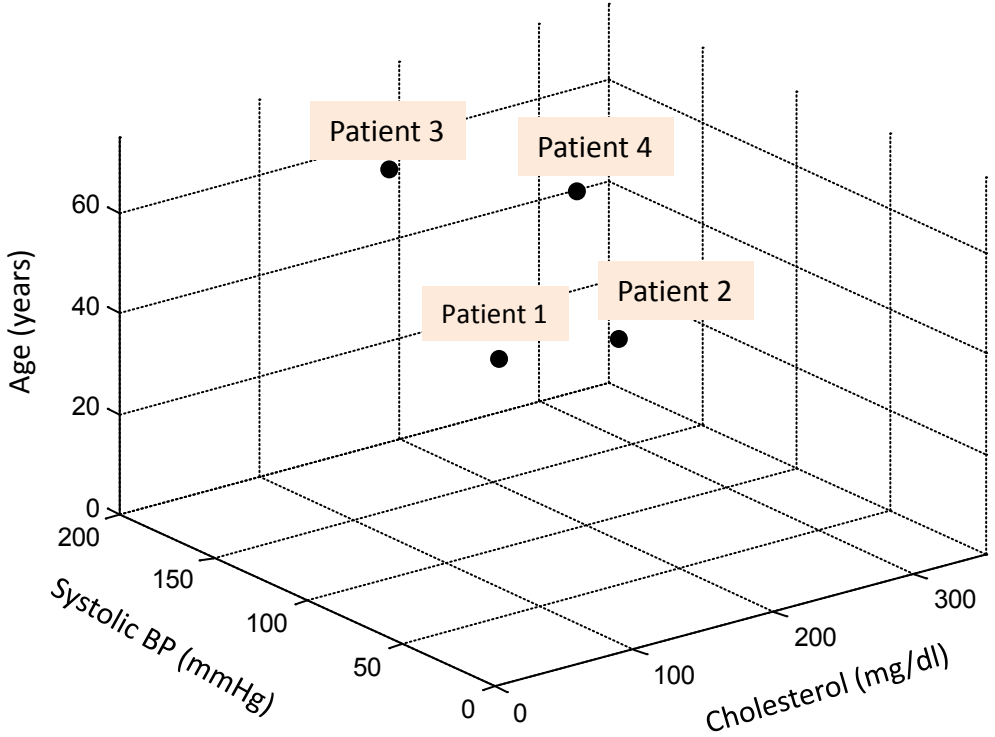


How to represent samples geometrically? Vectors in n-dimensional space ($\mathbb{R}^n$)



Patient id	Cholesterol (mg/dl)	Systolic BP (mmHg)	Age (years)	Tail of the vector	Arrow-head of the vector
1	150	110	35	(0,0,0)	(150, 110, 35)
2	250	120	30	(0,0,0)	(250, 120, 30)
3	140	160	65	(0,0,0)	(140, 160, 65)
4	300	180	45	(0,0,0)	(300, 180, 45)

How to represent samples geometrically? Vectors in n-dimensional space ($\mathbb{R}^n$)

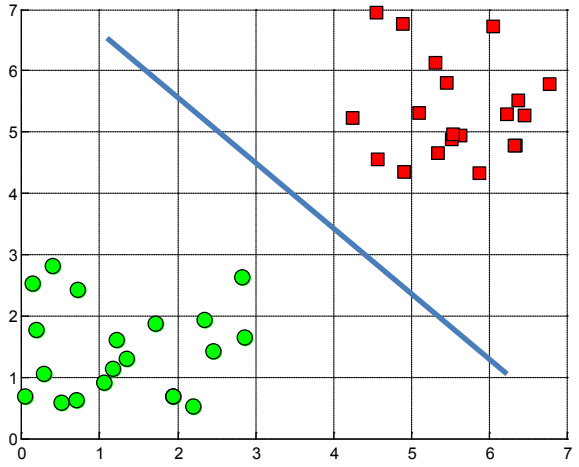


Since we assume that the tail of each vector is at point with 0 coordinates, we will also depict vectors as points (where the arrow-head is pointing).

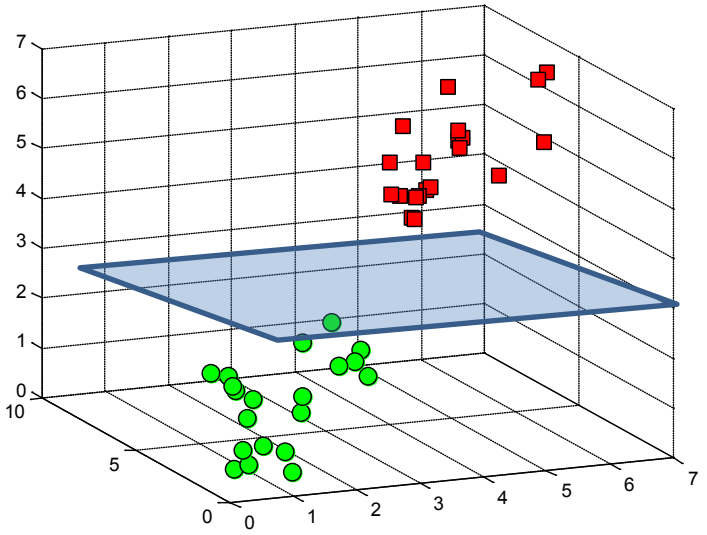
Hyperplanes as decision surfaces

- A hyperplane is a linear decision surface that splits the space into two parts;
- It is obvious that a hyperplane is a binary classifier.

A hyperplane in $\mathbb{R}^2$ is a line



A hyperplane in $\mathbb{R}^3$ is a plane



A hyperplane in $\mathbb{R}^n$ is an *n*-1 dimensional subspace

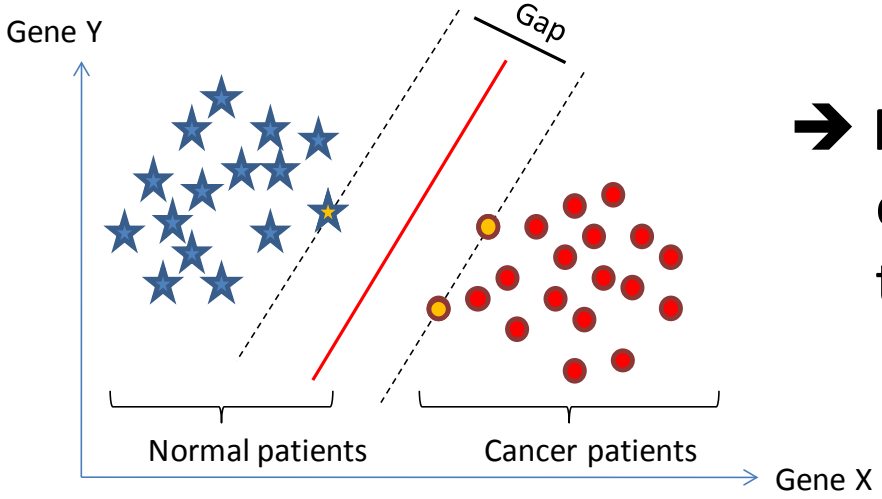
Recap

We know…

- How to represent patients (as “vectors”)
- How to define a linear decision surface (“hyperplane”)

We need to know…

- How to efficiently compute the hyperplane that separates two classes with the largest “gap”?



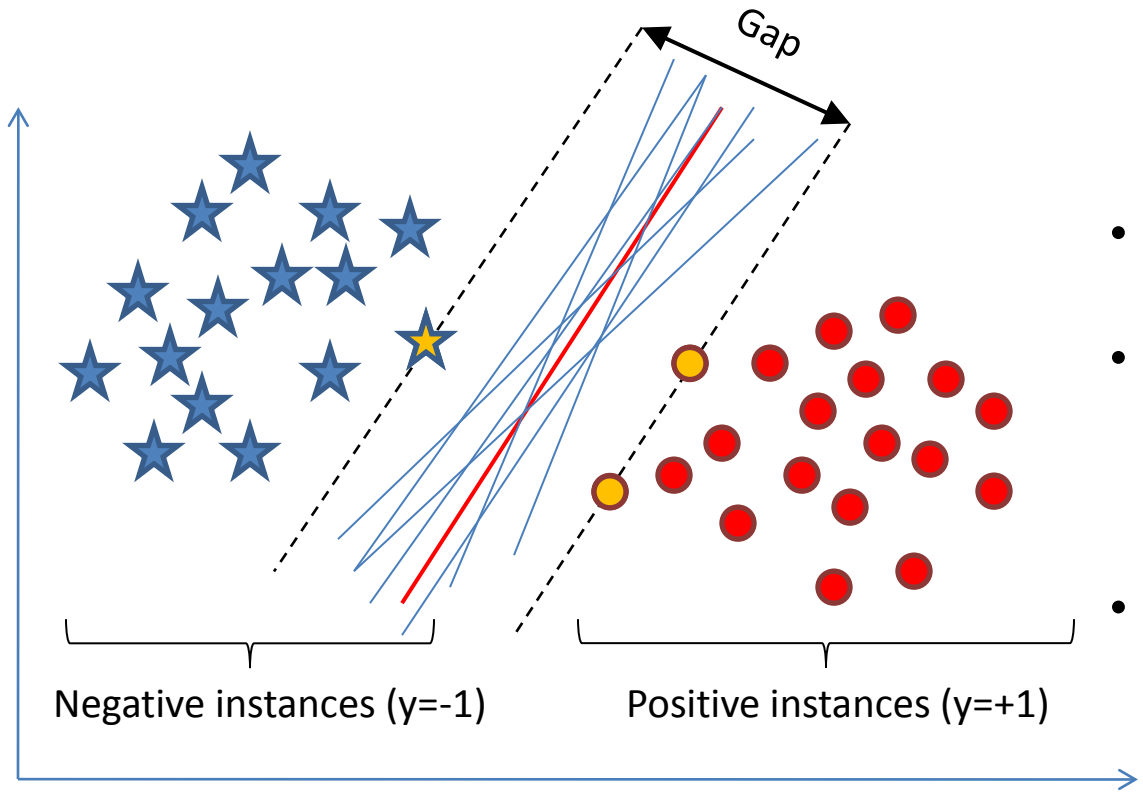
➔ Need to introduce basics of relevant optimization theory

Support vector machines for binary classification: classical formulation

Case 1: Linearly separable data; "Hard-margin" linear SVM

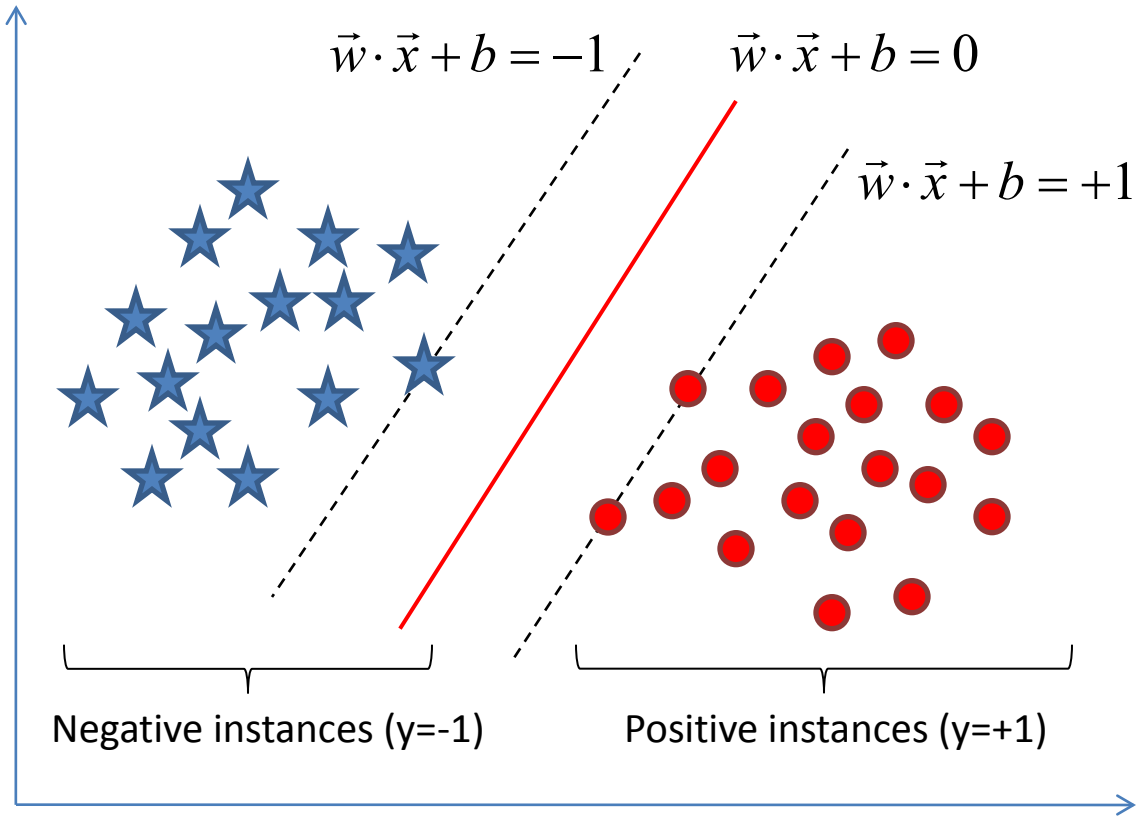
Given training data: $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_N \in R^n$

$y_1, y_2, \ldots, y_N \in \{-1, +1\}$



- Want to find a classifier (hyperplane) to separate negative instances from the positive ones.
- An infinite number of such hyperplanes exist.
- SVMs finds the hyperplane that maximizes the gap between data points on the boundaries (so-called "support vectors").
- If the points on the boundaries are not informative (e.g., due to noise), SVMs will not do well.

Statement of linear SVM classifier



The gap is distance between parallel hyperplanes:

$\vec{w}\cdot\vec{x}+b=-1$ and
$\vec{w}\cdot\vec{x}+b=+1$

Or equivalently:

$\vec{w}\cdot\vec{x}+(b+1)=0$
$\vec{w}\cdot\vec{x}+(b-1)=0$

We know that

$$D=\left|b_1-b_2\right|/\left\|\vec{w}\right\|$$

Therefore:

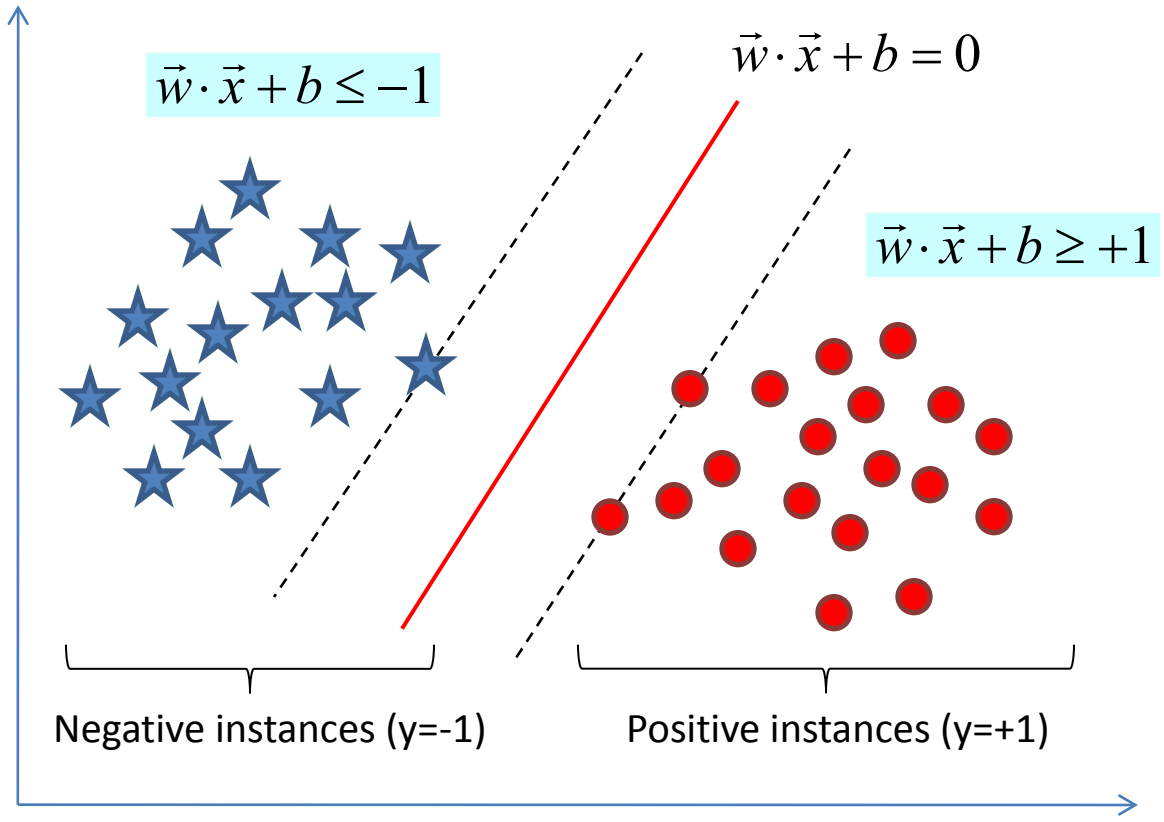
$$D=2/\left\|\vec{w}\right\|$$

Since we want to maximize the gap,

we need to minimize $\left\|\vec{w}\right\|$

or equivalently minimize $\frac{1}{2}\left\|\vec{w}\right\|^2$ ($\frac{1}{2}$ is convenient for taking derivative later on)

Statement of linear SVM classifier



In addition we need to impose constraints that all instances are correctly classified. In our case:

$\vec{w}\cdot\vec{x}_i+b\leq -1$ if $y_i=-1$

$\vec{w}\cdot\vec{x}_i+b\geq +1$ if $y_i=+1$

Equivalently:

$$y_i(\vec{w}\cdot\vec{x}_i+b)\geq 1$$

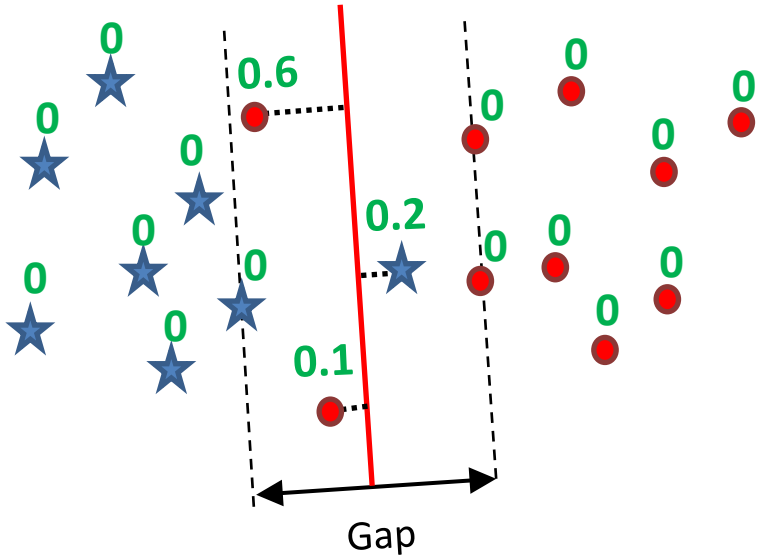
In summary:

Want to minimize $\frac{1}{2}\left\|\vec{w}\right\|^2$ subject to $y_i(\vec{w}\cdot\vec{x}_i+b)\geq 1$ for $i=1,\ldots,N$

Then given a new instance x, the classifier is $f(\vec{x})=sign(\vec{w}\cdot\vec{x}+b)$

Case 2: Not linearly separable data; "Soft-margin" linear SVM

What if the data is not linearly separable? E.g., there are outliers or noisy measurements, or the data is slightly non-linear.



Want to handle this case without changing the family of decision functions.

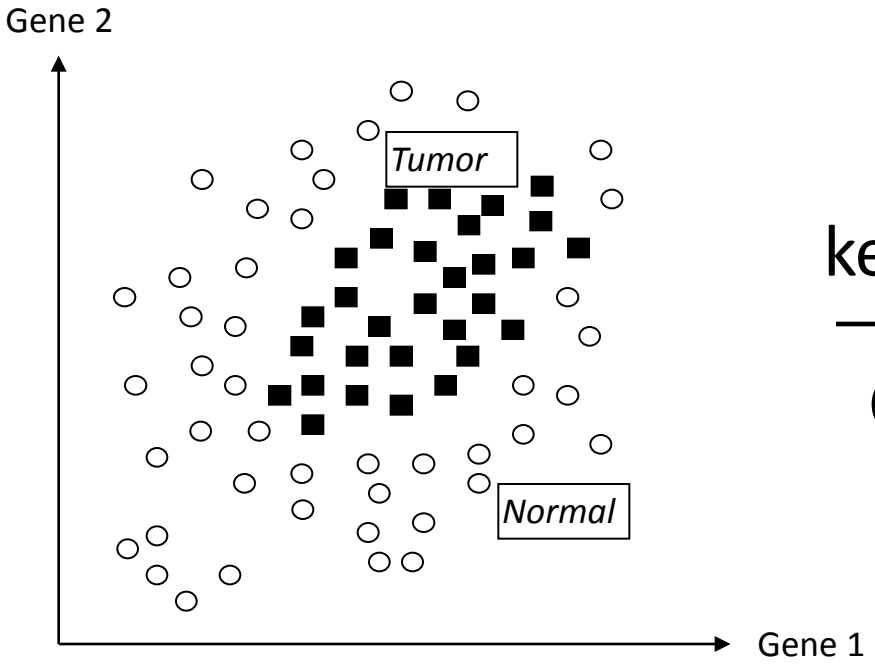
Approach:

Assign a "slack variable" to each instance $\xi_i \geq 0$, which can be thought of distance from the separating hyperplane if an instance is misclassified and 0 otherwise.

Want to minimize $\frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{N}\xi_i$ subject to $y_i(\vec{w}\cdot\vec{x}_i + b) \geq 1 - \xi_i$ for $i = 1,\ldots,N$

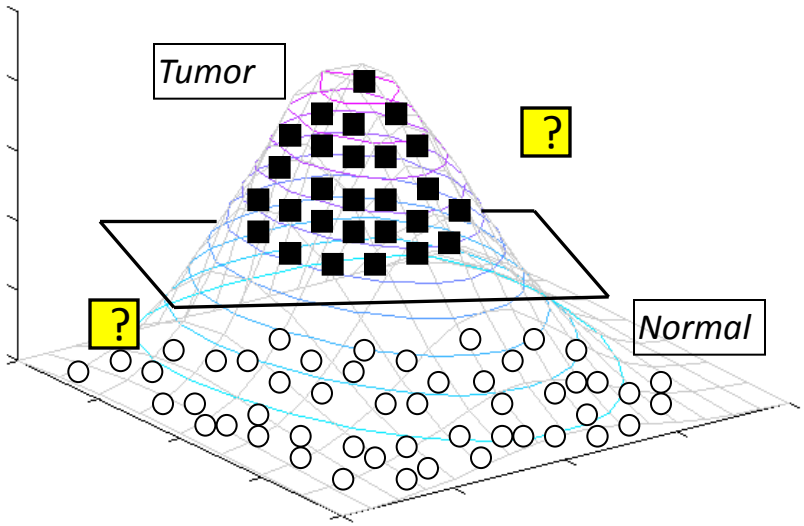
Then given a new instance x, the classifier is $f(x) = sign(\vec{w}\cdot\vec{x} + b)$

Case 3: Not linearly separable data; Kernel trick



kernel

Φ



Data is not linearly separable in the input space

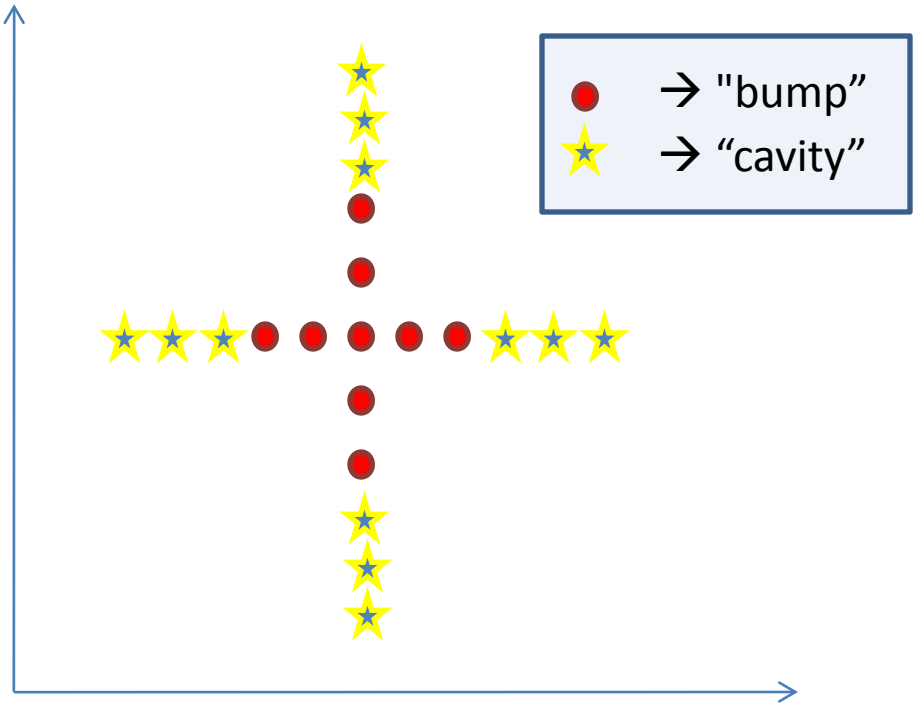
Data is linearly separable in the feature space obtained by a kernel

$$\Phi : \mathbf{R}^N \rightarrow \mathbf{H}$$

Understanding the Gaussian kernel

Consider Gaussian kernel: $K(\vec{x}, \vec{x}_j) = \exp(-\gamma \left\| \vec{x} - \vec{x}_j \right\|^2)$

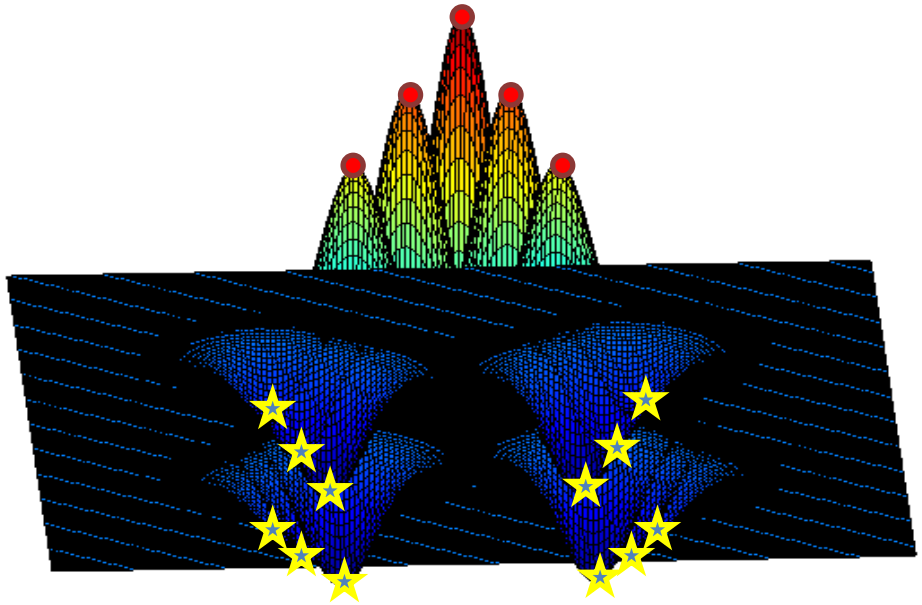
Geometrically, this is a "bump" or "cavity" centered at the training data point $\vec{x}_j$:

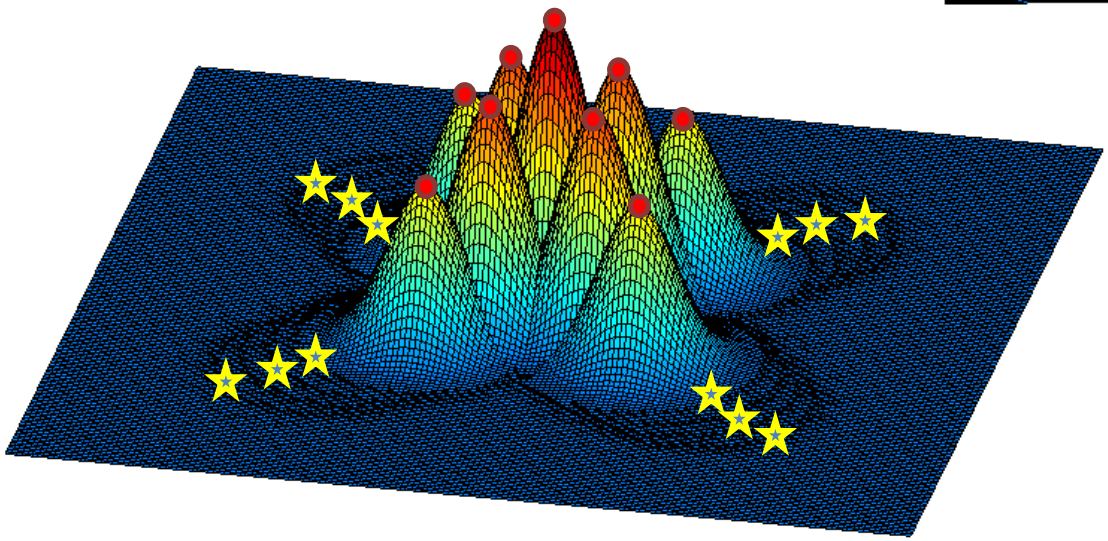


The resulting mapping function is a **combination** of bumps and cavities.

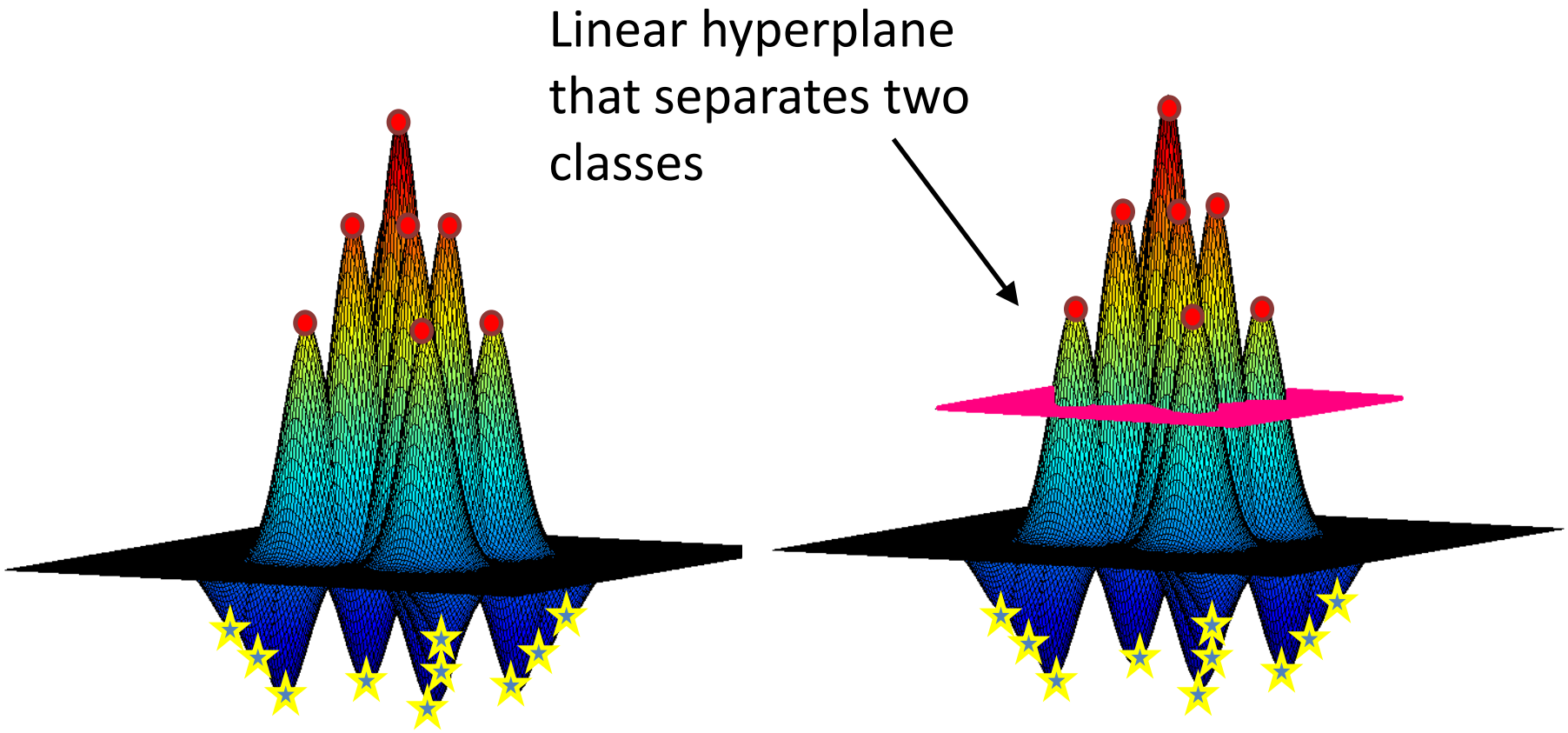
Understanding the Gaussian kernel

Several more views of the data is mapped to the feature space by Gaussian kernel





Understanding the Gaussian kernel

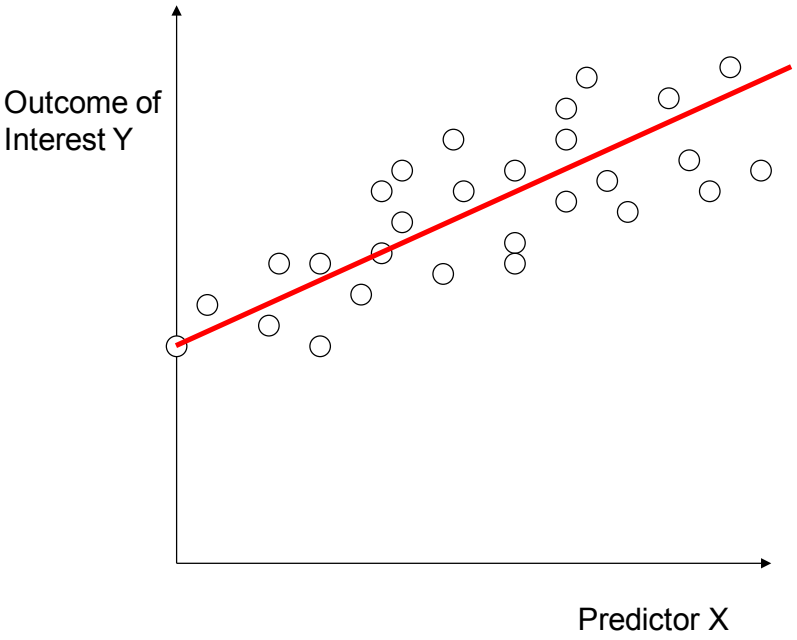


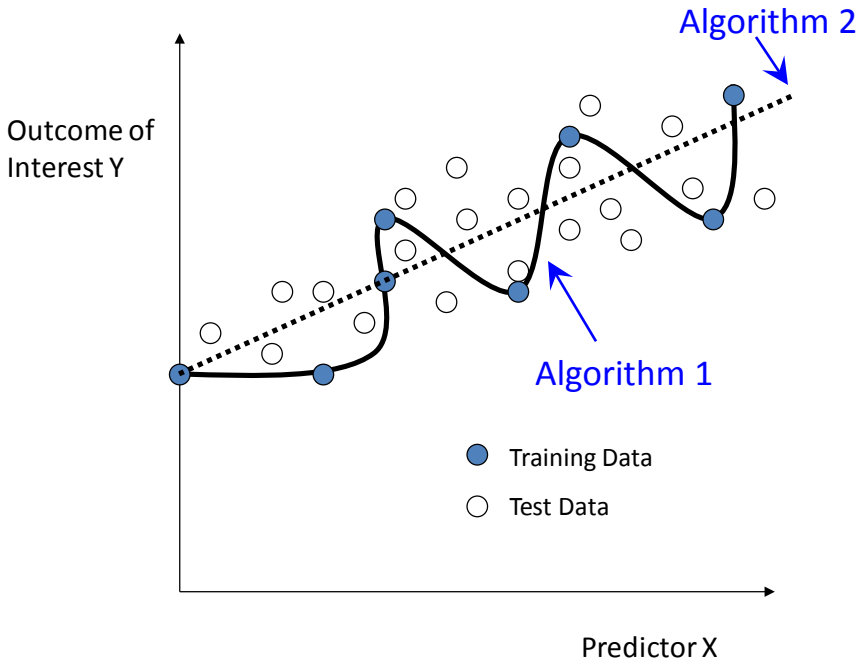
Generalization and overfitting

- **Generalization:** A classifier or a regression algorithm learns to correctly predict output from given inputs not only in previously seen samples but also in previously unseen samples.
- **Overfitting:** A classifier or a regression algorithm learns to correctly predict output from given inputs in previously seen samples but fails to do so in previously unseen samples.
- **Overfitting → Poor generalization**.

Example of overfitting and generalization

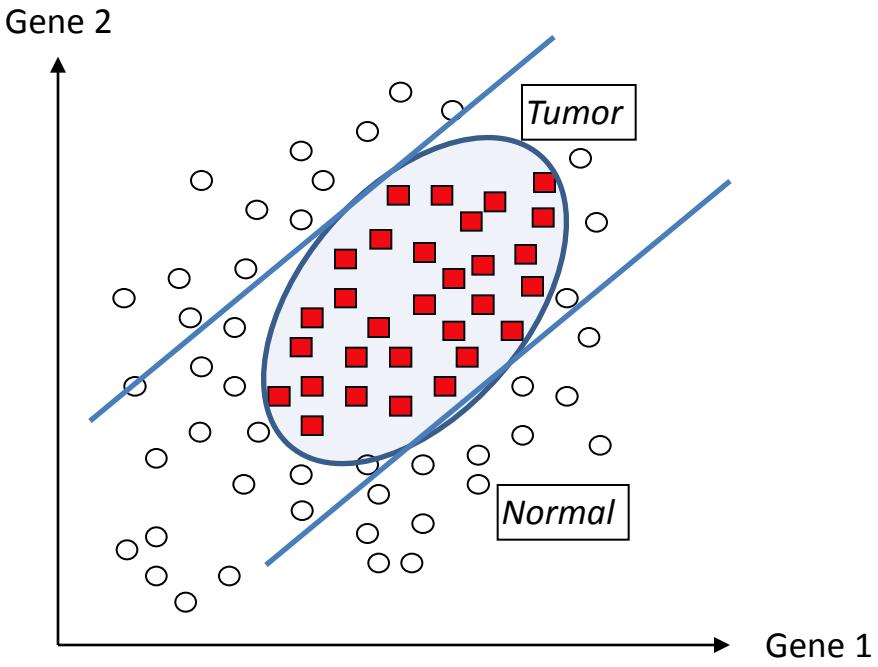
There is a linear relationship between predictor and outcome (plus some Gaussian noise).



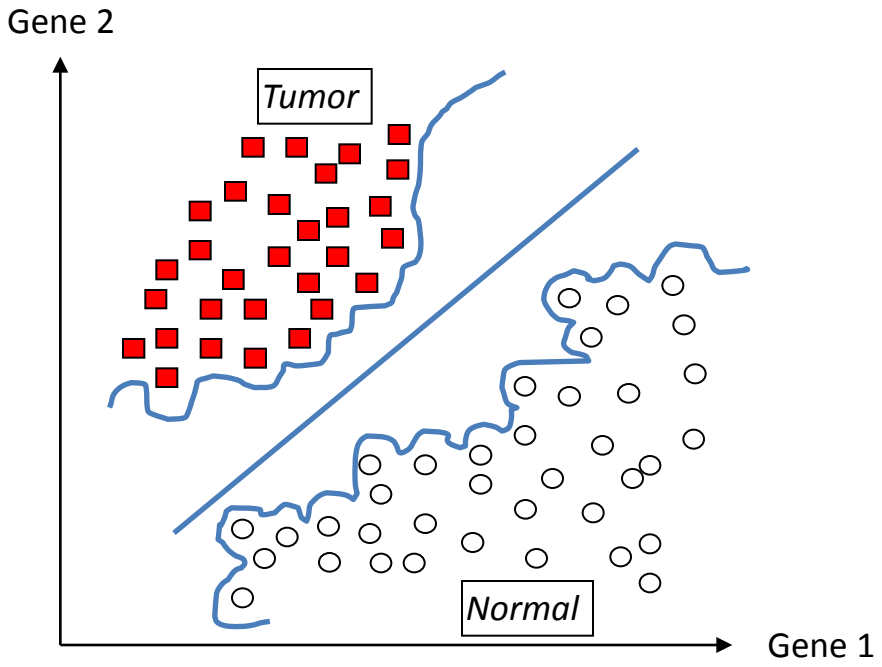


- Algorithm 1 learned non-reproducible peculiarities of the specific sample available for learning but did not learn the general characteristics of the function that generated the data. Thus, it is overfitted and has poor generalization.
- Algorithm 2 learned general characteristics of the function that produced the data. Thus, it generalizes.

Need for model selection for SVMs



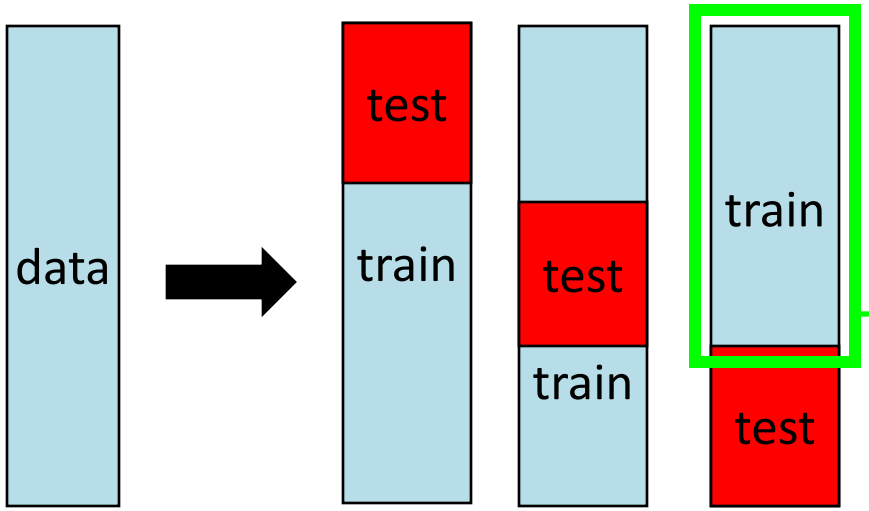
- It is impossible to find a linear SVM classifier that separates tumors from normals!
- Need a non-linear SVM classifier, e.g. SVM with polynomial kernel of degree 2 solves this problem without errors.



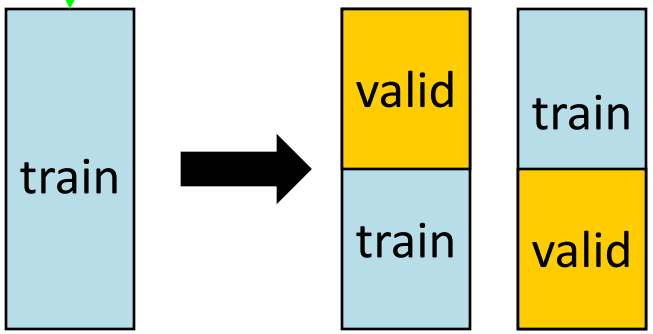
- We should not apply a non-linear SVM classifier while we can perfectly solve this problem using a linear SVM classifier!

Nested cross-validation

Recall the main idea of cross-validation:

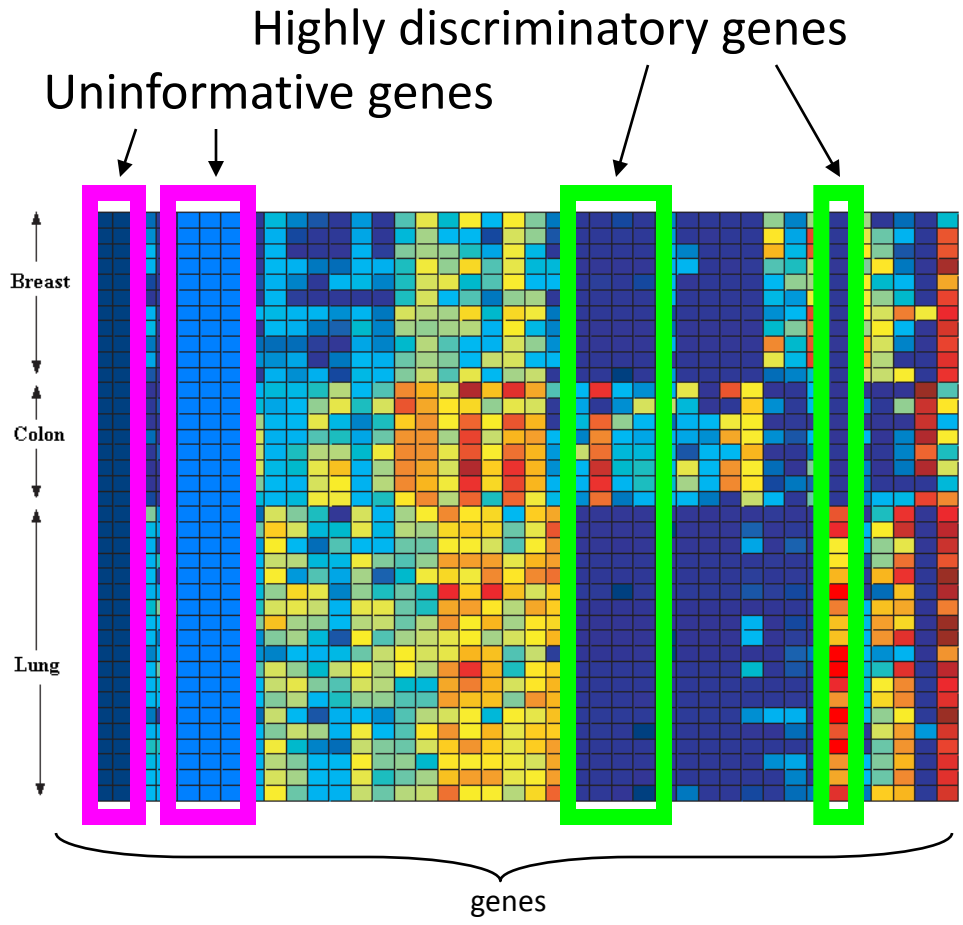


What combination of SVM parameters to apply on training data?



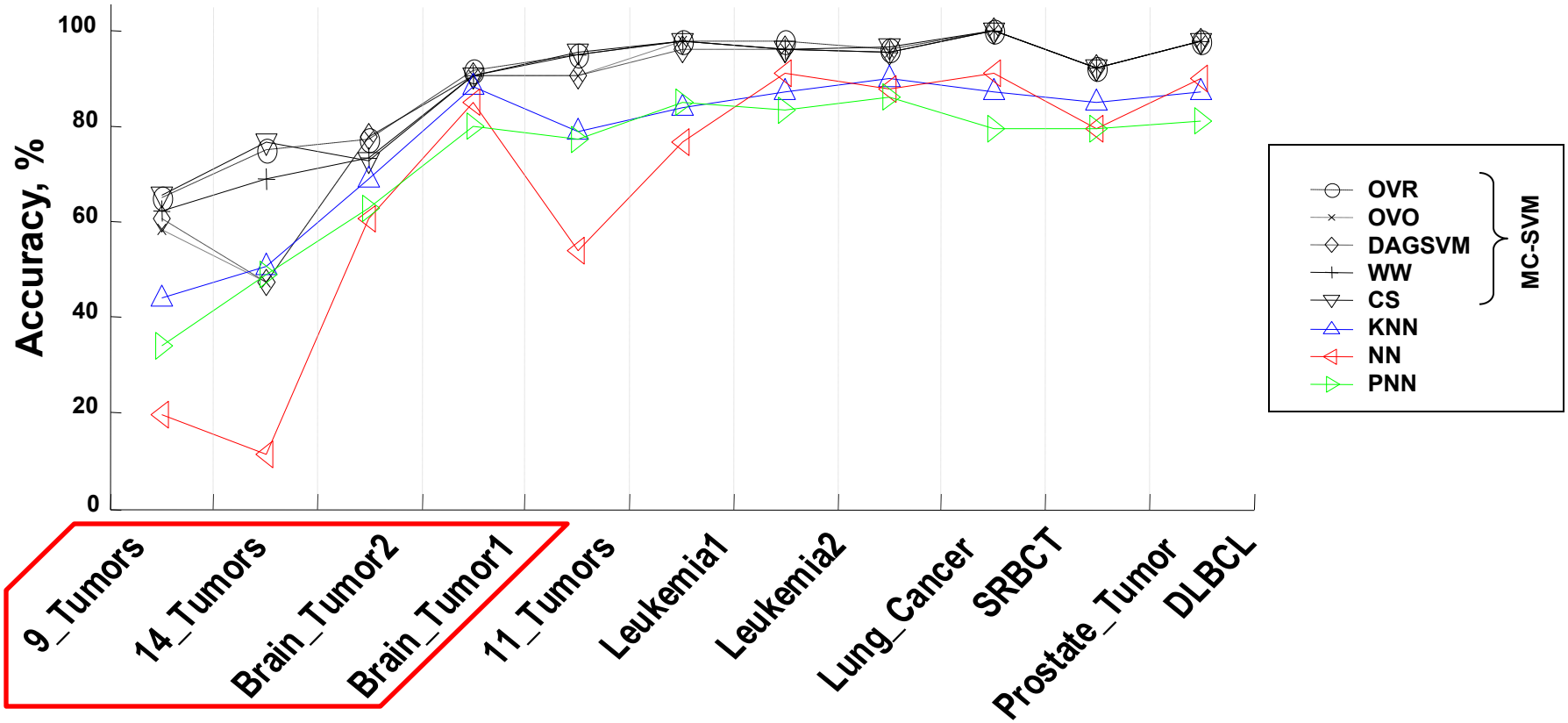
Perform “grid search” using another nested loop of cross-validation.

Gene selection methods



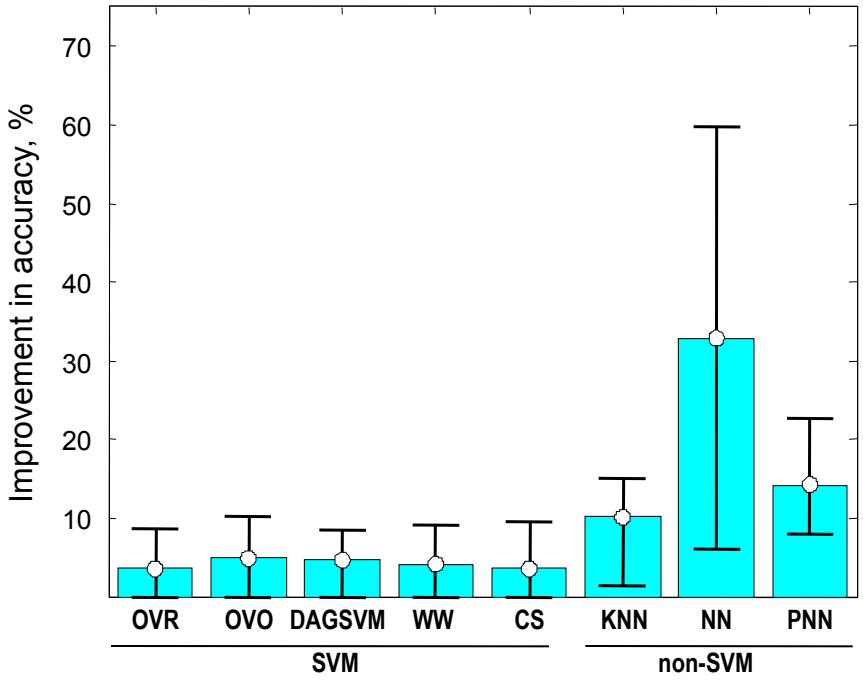
1. Signal-to-noise (S2N) ratio in one-versus-rest (OVR) fashion;
2. Signal-to-noise (S2N) ratio in one-versus-one (OVO) fashion;
3. Kruskal-Wallis nonparametric one-way ANOVA (KW);
4. Ratio of genes between-categories to within-category sum of squares (BW).

Results without gene selection

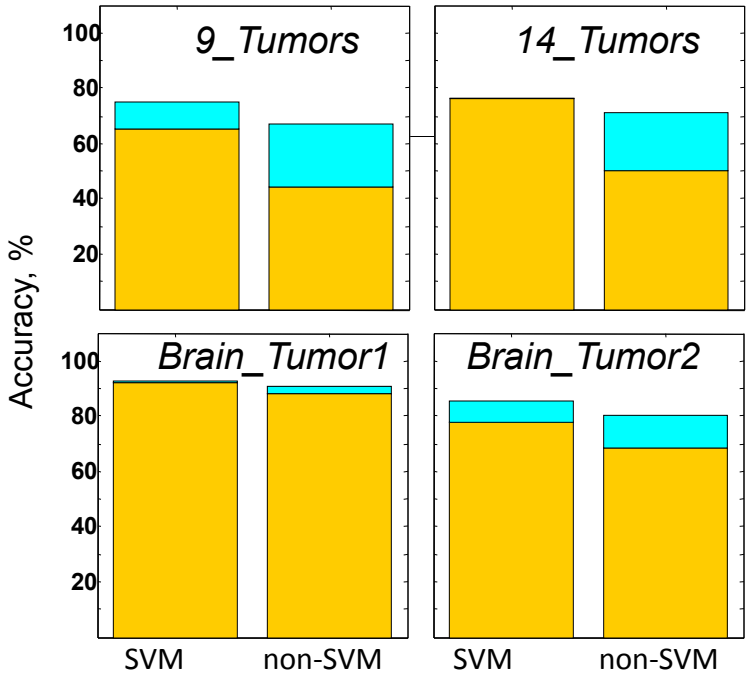


Results with gene selection

Improvement of diagnostic performance by gene selection (*averages for the four datasets*)



Diagnostic performance *before* and *after* gene selection



Average reduction of genes is 10-30 times