Machine Learning: Some Techniques in Statistical Learning and Modeling

Part I: Support Vector Machines and Kernels

Leopoldo Bertossi

Carleton University

School of Computer Science

Ottawa, Canada

Vectors, Inner Products, Distances and Kernels

ML is related, in one way or another, with a notion of distance

E.g. the distance from a regression line to the set of sample points

Distances are related to metrics and norms in vector spaces

We can take advantage of vector space notions, in particular, whenever possible, e.g. in $\mathbb{R}^d$ as a vector space, to interior (dot) products

They are sometimes related to- or defined in terms of "kernels"

Many contemporary ML approaches employ kernels or distance metrics

Lately there has been a significant interest in developing kernels and distance metrics for structured and relational data[1]

[1] We closely follow the treatment in Luc De Daedt. *Logical and Relational Learning*. Springer, 2010. See also Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2002.

A *kernel function* can be directly interpreted as measuring some kind of similarity

The distance between two objects is inversely related to their similarity

To fix ideas, assume that our vector space $\mathcal{V}$ is $\mathbb{R}^d$ over the reals

For vectors $\mathbf{a} = \langle a_1, \ldots, a_d \rangle$ and $\mathbf{b} = \langle b_1, \ldots, b_d \rangle$ in $\mathbb{R}^d$, the inner product is defined by:[2]

$$\mathbf{a} \bullet \mathbf{b} := \sum_{i=1}^{d} a_i \cdot b_i \tag{1}$$

The *norm* of a vector $\mathbf{a} \in \mathbb{R}^d$ is: $||\mathbf{a}|| := \sqrt{\mathbf{a} \bullet \mathbf{a}}$

The Euclidean distance between vectors $\mathbf{a}$ and $\mathbf{b}$ is defined by:

$$d_E(\mathbf{a}, \mathbf{b}) := ||\mathbf{a} - \mathbf{b}||,$$

which goes back to (1)

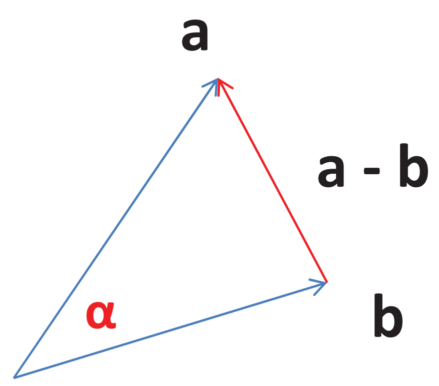
[2]Sometimes denoted $\langle \mathbf{a}, \mathbf{b} \rangle$

It holds: $\mathbf{a} \bullet \mathbf{b} = ||\mathbf{a}|| \cdot ||\mathbf{b}|| \cdot cos(\alpha)$

Here, $cos(\alpha)$ is the angle formed by $\mathbf{a}$ and $\mathbf{b}$

Then:

- When $\alpha = 0$, $\mathbf{a} \bullet \mathbf{b} = 1$ (if $||\mathbf{a}|| = ||\mathbf{b}|| = 1$)
- When $\mathbf{a}$ and $\mathbf{b}$ are orthogonal, the inner product is 0



We can see that the inner product measures a kind of similarity

We consider symmetric, positive-definite kernels on $\mathbb{R}^d$, i.e. functions $K\colon\ \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ (more on this later ...)

For them in particular: $K(\mathbf{a},\mathbf{a}) \geq 0$, and $K(\mathbf{a},\mathbf{a}) > 0$ for $\mathbf{a} \neq \mathbf{0}$

Any positive-definite kernel K induces a distance metric d_K:

$$d_K(\mathbf{a},\mathbf{b}) := \sqrt{K(\mathbf{a},\mathbf{a}) - 2K(\mathbf{a},\mathbf{b}) + K(\mathbf{b},\mathbf{b})}$$

Exercise: Verify that $d_E(\mathbf{a},\mathbf{b}) = d_K(\mathbf{a},\mathbf{b})$ with $K(\mathbf{a},\mathbf{b}) := \mathbf{a} \bullet \mathbf{b}$

Exercise: Consider only vectors in $\{-1,+1\}^d$ (a common discrete case in ML), and the same inner product $\mathbf{a} \bullet \mathbf{b}$

For example, for $d = 3$, it holds: $\langle 1,-1,-1\rangle \bullet \langle -1,-1,1\rangle = -1+1-1 = -1$

What is the maximum value of the inner product? When is it taken? The minimum value? When is it taken? Analyze all this geometrically, and also the distance function associated to this inner product

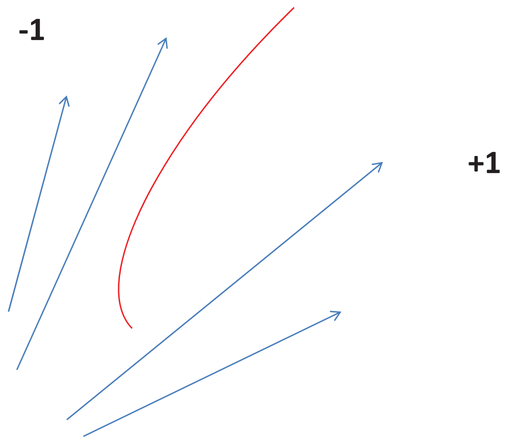
Introduction to Kernel-Based Methods in ML

The Max-Margin Approach:

This is for a binary classification task of vectors, say in $\mathbb{R}^d$

The classes under which they are classified correspond to $+1$ and -1

Intuitively, the frontier in red has to be learned



We can look for a separation plane, more precisely, a hyperplane, a generalization of the notion of plane that we know in $\mathbb{R}^3$ (think also of linear regression in $\mathbb{R}^2$)

E.g. a hyperplane in $\mathbb{R}^2$ is a straight line, with general equation:
$a \cdot x + b \cdot y + c = 0$, with $a, b, c \in \mathbb{R}$, $|a| + |b| > 0$

A hyperplane in $\mathbb{R}^3$ is usual plane, with general equation:
$a \cdot x + b \cdot y + c \cdot z + d = 0$, with $a, b, c, d \in \mathbb{R}$, $|a| + |b| + |c| > 0$

A hyperplane in $\mathbb{R}^d$ can be defined in general by an equation of the form: $\mathbf{w} \bullet \mathbf{x} + b = 0$ (denote the LHS with $\mathcal{H}(\mathbf{x})$)

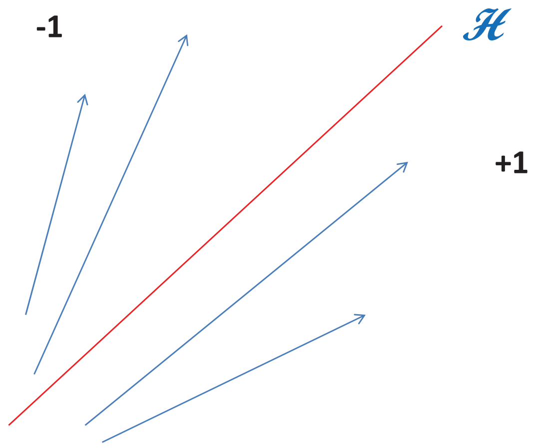
Here, $\mathbf{x}$ is a vector variable in $\mathbb{R}^d$ (say $\langle x_1, \ldots, x_d \rangle$), $\mathbf{w}$ is a "weight" vector in $\mathbb{R}^d$, and $b, 0$ are real constants

E.g. in $\mathbb{R}^4$, this could be the equation of a hyperplane:

$$\mathcal{H}(x_1, x_2, x_3, x_4) := \langle 2.3, 4.5, -2.6, 5.8 \rangle \bullet \langle x_1, x_2, x_3, x_4 \rangle + 7.9 = 0$$

Hyperplane of all vectors $\mathbf{x}$ that are solutions to the equation, making the real-valued function $\mathcal{H}(\cdot)$ on $\mathbb{R}^d$ take the value 0

A hyperplane $\mathcal{H}$ can then be used to classify any vector (an example) $\mathbf{e}$ as positive or negative (in one or the other class), depending on which side of the $\mathcal{H}$ it lies



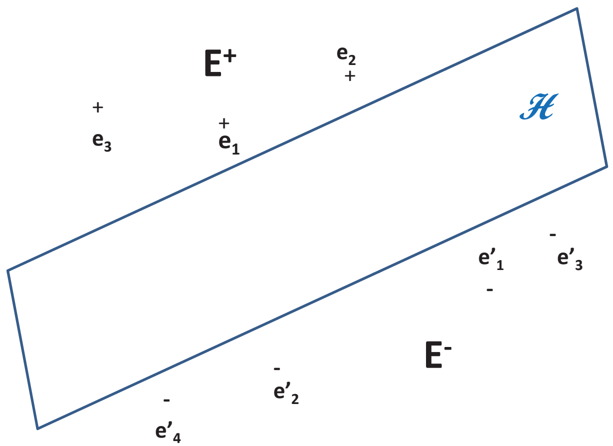
This is determined by computing: $h(\mathbf{e}) := sgn(\mathbf{w} \bullet \mathbf{e} + b)$

Here, $sgn(y) = 1$ when $y > 0$; and $sgn(y) = -1$ when $y \leq 0$

Given a set $E = E^+ \cup E^-$ of examples, say $\mathbf{e}_1, \ldots, \mathbf{e}_N$, we want to learn a hyperplane $\mathcal{H}$ (i.e. $\mathbf{w}$ and b) that separates the two classes

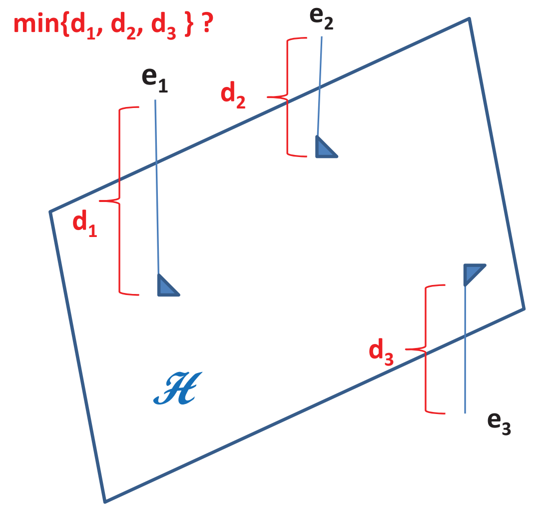
Notice that here we have a *feature* associated to examples and points in general, the measure function $f\colon \mathcal{V} \to \{-1, +1\}$

Here, $f(e) = +1$, for every $e \in E^+$



We want a hyperplane, $\mathcal{H}^o$, that is "optimal", and will be used to classify new, incoming vectors $\mathbf{e}$

We first consider "the margin" for E, which is the minimum distance between an example and the hyperplane



$$margin(\mathcal{H}, E) := min\{||\mathbf{x} - \mathbf{e}_i|| \mid \mathbf{e}_i \in E, \text{ and } \mathbf{w} \bullet \mathbf{x} + b = 0\}$$

We are minimizing by making both the points in the plane (the $\mathbf{x}$) and the examples (the $\mathbf{e}_i$) vary, obtaining what is, by definition, the distance of a set of points to a hyperplane

In this minimization, $\mathbf{w}$ and b are fixed ...

A quadratic function and a linear condition on $\mathbf{x}$

Given the set of examples, each hyperplane $\mathcal{H}$ has a margin, which is a minimum

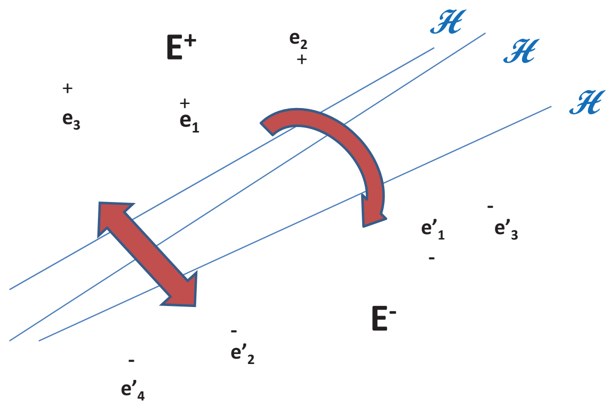
Which hyperplane should be choose?

We can move a hyperplane (to be) in the range between E^- and E^+

We can easily make the margin equal to 0, by making one of the points fall on the plane

A balance must be found between the two classes E^+, E^-

We want a hyperplane that generalizes well beyond the set of given points



This is a crucial property, at the very heart of SVM, we will revisit

The maximum margin approach computes the optimal hyperplane, that is, the hyperplane $\mathcal{H}^o$ that maximizes the margin: (a max-min)

$$\mathcal{H}^o := \text{arg max}_{\mathcal{H}}\, margin(\mathcal{H}, E) \tag{2}$$

The theory of statistical learning shows that maximizing the margin provides optimal generalization behavior

Generalization has to do with over-fitting, “regularization”

With the possibility of adequately classifying new incoming points, as opposed to only the examples provided for the learning process

How to compute $\mathcal{H}^o$?

Support Vector Machines (basic case)

Computing the maximum margin hyperplane is an optimization problem

A real-valued function is maximized in (2)

Most ML problems can be casted as optimization problems

We have to find (optimal) $\mathbf{w}$ and b

As usual, this problem has some constraints that have to be satisfied during the optimization process

A first requirement is that the training examples $(\mathbf{e}_i, f(\mathbf{e}_i))$ are correctly classified

Mathematically, this results in the constraints requiring that the feature and the location of each example wrt. the hyperplane have the same sign:

$$\text{For every } \mathbf{e} \in E, \quad f(\mathbf{e}) \cdot \mathcal{H}(\mathbf{e}) = f(\mathbf{e}) \cdot (\mathbf{w} \bullet \mathbf{e} + b) > 0 \qquad (3)$$

A second set of constraints comes from the requirement that the closest training examples $\mathbf{e}$ to the hyperplane depart (or deviate) by 1 from it: (no loss of generality, just re-scaling)

$$|\mathbf{w} \bullet \mathbf{e} + b| = 1 \qquad (4)$$

(3) and (4) can be combined into a single set of constraints:

$$\text{For every } \mathbf{e} \in E, \quad f(\mathbf{e}) \cdot (\mathbf{w} \bullet \mathbf{e} + b) \geq 1$$

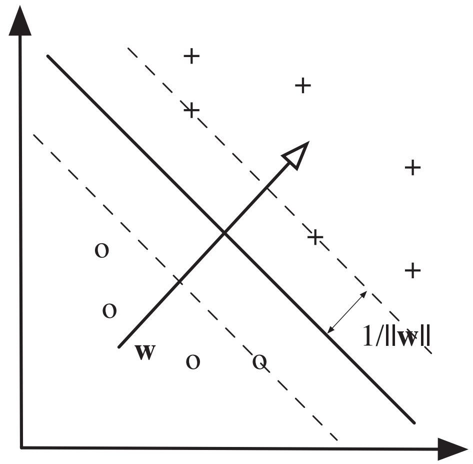
Now, it holds: $margin(\mathcal{H}, E) = \frac{1}{||\mathbf{w}||}$ [9], [3, sec. 6.1.1]

Then, maximizing the margin is equivalent to minimizing $||\mathbf{w}||$, or equivalently, minimizing $\mathbf{w} \bullet \mathbf{w}$

Finally, the conditioned optimization problem becomes:

$$\min_{\mathbf{w},b} \mathbf{w} \bullet \mathbf{w} \quad (5)$$

subject to: For every $\mathbf{e} \in E$: $\quad f(\mathbf{e}) \cdot (\mathbf{w} \bullet \mathbf{e} \; + b) \geq 1$



Here, the positive examples are marked with a "+" and the negative ones with an "o"

The maximum-margin hyperplane is the middle line

$\mathbf{w}$ is a vector normal to the margin strip

(5) is a minimization problem of a quadratic function with N linear conditions on $\mathbf{w}$

There are established methods for solving this minimization problem (OP), i.e. for computing $\mathbf{w}$ and b

A standard one consists in turning it into an OP of a different form

We now use Kuhn & Tucker' theory [8] [3, sec. 6.1.1]

We introduce N non-negative Lagrange multipliers, α_i, one for each condition, and produce a new max-min problem:

$$\max_{\alpha_i} \min_{\mathbf{w},b} \ \frac{1}{2}(\mathbf{w} \bullet \mathbf{w}) + \sum_{i=1}^{N} \alpha_i \cdot [y_i \cdot (\mathbf{w} \bullet \mathbf{e}_i + b) - 1] \tag{6}$$

Here, $y_i = f(\mathbf{e}_i), \ i = 1, \cdots, N$

Partially deriving wrt. $\mathbf{w}$ and b we obtain that the weight vector $\mathbf{w}$ is a linear combination of the data vectors:

$$\mathbf{w} = \sum_i \alpha_i y_i \cdot \mathbf{e}_i \tag{7}$$

This turns (6) into the *dual optimization problem*:

$$\max_{\alpha_i} \ \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \ \mathbf{e}_i \bullet \mathbf{e}_j$$

subject to:[3] For every i: $\alpha_i \geq 0$ and $\sum_i y_i \alpha_i = 0$

[3] 2nd condition here follows from partial derivation above

This conditioned, quadratic OP becomes that of computing the optimal α_i, and is typically easier to solve

There are various implementations, which use Gram's matrix that contains the values $\mathbf{e}_i \bullet \mathbf{e}_j$ over E

Typically, many α_i have the value 0

Those vectors $\mathbf{e}_i$ with non-zero α_i are the support vectors (SVs)

In the figure on slide 14, the SVs lie on the dotted lines (margin borders); for them:

$$f(\mathbf{e}_i) \cdot (\mathbf{w} \bullet \mathbf{e}_i + b) = 1 \tag{8}$$

SVs are the only ones that influence the position of $\mathcal{H}^o$

Constant b does not appear in the dual problem, and can be obtained by substituting in (8) one SV for each class [3, rem. 6.4]

Having determined $\mathcal{H}^o$, what about classification of new values?

From (7) and slide 8, we obtain that a new vector (data point) $\mathbf{e}$ is classified by computing its value $h(\mathbf{e})$:

$$h(\mathbf{e}) = sgn(\sum_i \alpha_i \cdot f(\mathbf{e}_i) \cdot \mathbf{e}_i \bullet \mathbf{e} + b)$$

Only the SVs need to be taken into account in this sum

Fully expressed in terms of inner products ...

Remarks:

- We assumed that the classes E^+, E^- are linearly separable, in the sense that there is a hyperplane that separates them

  See top of slide 14 ...

- If not, the technique can be applied introducing “slack variables”

  Or mapping the space to another one where separation is achieved

  Coming ... [6, sec. 9.1.5]

- It can be applied with noisy data, using “soft” borders [3, sec. 6.1.2]

- We addressed the classification problem of determining to which of two complementary classes points belong

  Or whether points belong to a certain spatial region (or not)

  This is a single-task problem

We could address multi-task classification problems as well, i.e. classifying a point according to several criteria (several spatial regions)

- It is possible to apply SVM to learn a regression model

  For predicting the values of a real-valued spatial function

  Minimized loss functions are introduced, as with regression lines on the plane with minimum-squares minimization

- We used only the classical inner product, induced by a very particular kernel

  We could use any kernel with reasonable properties ... Coming ...

LogicQL and an Application to ER

Integrating traditional data management with ML is a new area of research and technology

- A newer and increasingly hot topic
- The two processes give feedback to each other

  Data, explicit and generated, is used for ML

  Results from ML are mathematical models that generate new data
- Tendency is to create single systems that integrate: data management, optimization, e.g. linear integer programming (ILP), and ML

  Not surprising: Many (most?) machine ML problems and techniques can be cast as optimization problems

A personal experience ...

LogicQL: new name for the LogicBlox system

Developed and used by the company LogicBlox

`http://www.logicblox.com/`

Goal: Extend, implement and leverage Datalog technology

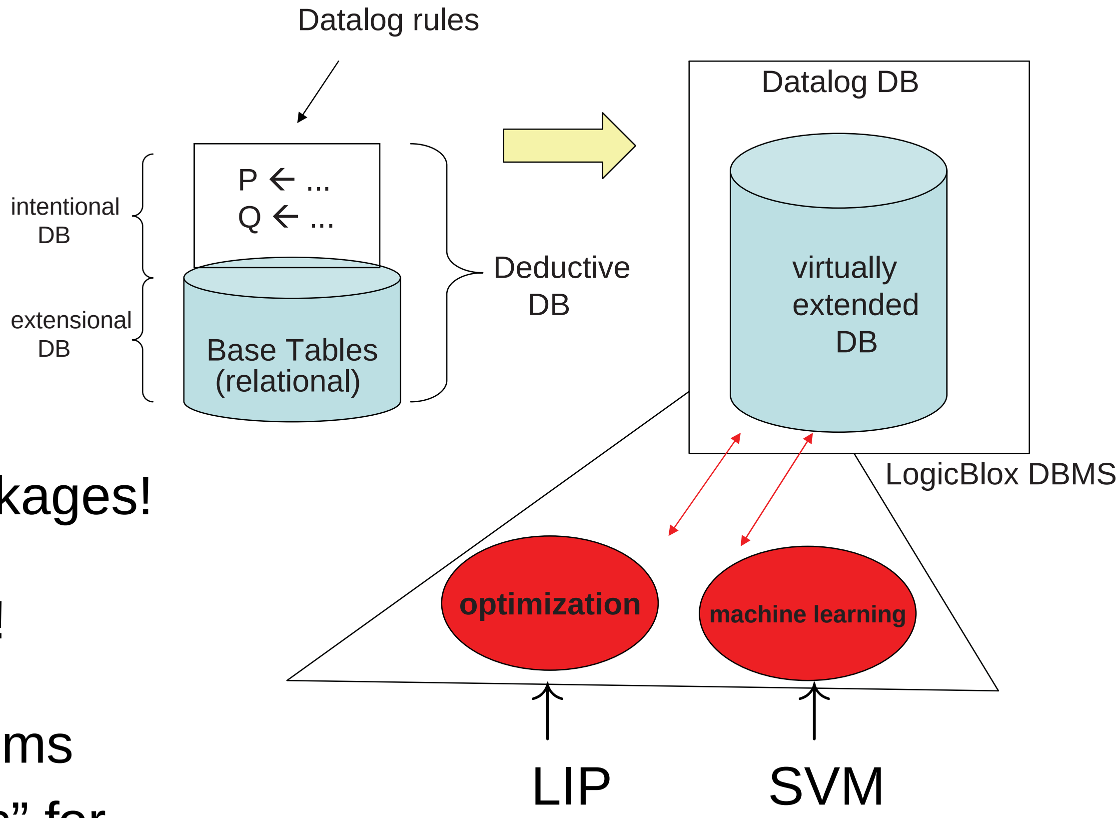
Datalog has been around since the early 80s

Used mostly in DB research

It has experienced a revival during the last few years, and many new applications have been found!

Declarative and executable specifications of data-related domains

An extension of relational algebra/calculus/databases



LogicQL is tightly coupled with ML and optimization packages!

In one single system!

Data for these problems stored as “extensions” for DB/Datalog predicates

Optimizer reads necessary data from tables or Datalog computations

Results of optimizations may become contents for newly defined predicates

Smooth interaction between Datalog/relational engine and optimization packages ...

New optimization and ML methods are being added ...

Optimization methods/packages developed by other groups and companies

ML methods mostly developed in house ...

`http://www.ismion.net/documentation/index.html`

In particular, several applications to logistics

We applied the system with its SVM implementation to entity resolution (ER) or record de-duplication in data cleaning (more coming)

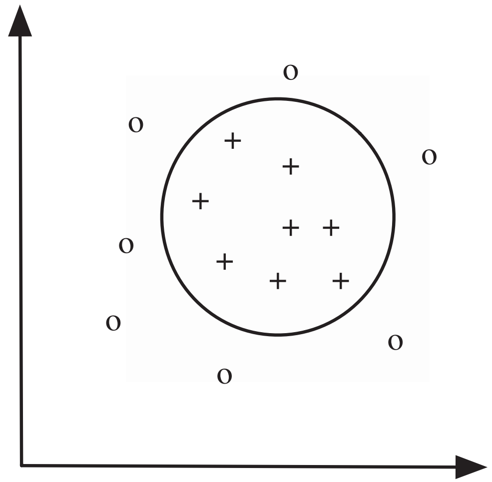
Generalizing SVM and Statistical Learning: The Kernel Trick

Again, the classification problem

We may not have separability

$+$: positive examples

$\circ$: negative examples



The max margin approach does not work in this case

Idea: Map the vectors to a different vector (feature) space where the classes become separable

And then apply the inner-product ...

Example: $\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ given by: $(x_1, y_1) \mapsto (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$

The classes in the previous slide become separable (verify this!)

So, vectors $\mathbf{x} \in \mathbb{R}^2$ are mapped to vectors $\Phi(\mathbf{x}) \in \mathbb{R}^3$ via the feature function Φ

In $\mathbb{R}^3$ there is the usual inner product

So, we have defined a “new inner-product” in $\mathbb{R}^2$:

$$\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^2 \quad \mapsto \quad K(\mathbf{x}_1, \mathbf{x}_2) := \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2) \rangle \quad (= \Phi(\mathbf{x}_1) \bullet \Phi(\mathbf{x}_2))$$

A “kernel” defined in $\mathbb{R}^2$! (different from the inner-product kernel there)[4]

It holds in $\mathbb{R}^2$: $K(\mathbf{x}_1, \mathbf{x}_2) = \langle \mathbf{x}_1, \mathbf{x}_2 \rangle^2$ The kernel trick!

The kernel can be computed reusing the “old” inner product in $\mathbb{R}^2$!

[4]We still have to elaborate on the desirable properties of a kernel

Some remarks:

- Kernels have been around in math for much longer than ML methods

  They can be defined in abstract terms, without appealing directly to inner products

  However, they are sometimes defined as inner-product based mappings or inner-products in feature spaces [3, sec. 3.2]

- A kernel-based inner-product (simply, kernel) in $\mathbb{R}^2$ becomes an (ordinary) inner-product in some feature space (a subspace or "manifold" of $\mathbb{R}^3$ induced by a function as Φ above)

- The (mapped) classes are linearly separable in the latter space

- The kernel-related inner-product in $\mathbb{R}^3$ can be reduced to the computation of the usual inner-product in $\mathbb{R}^2$

  Nice: the usual inner-product can be applied to the original examples without transforming them into examples in the feature space

  Efficient: the dimension of the original space is smaller than that of the feature space

Some General Remarks about Kernels

1. Start with a (real-valued) kernel K in the space $\mathcal{V}$ of the examples
2. The classes in $\mathcal{V}$ may not be (linearly) separable
3. Space $\mathcal{V}$, with its kernel K, is mapped to a (manifold) in a larger-dimension space $\mathcal{V}'$ with a kernel K'
4. The classes become separable in $\mathcal{V}'$
5. The SVM method can be applied with K' in $\mathcal{V}'$
6. For this to work the involved kernels need good properties[5]

   For symmetric $K(\cdot,\cdot)$ in $\mathcal{V}$, mainly, *positive definiteness*:[6]

   For all $m \in \mathbb{N}$, $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathcal{V}$, $a_1, \ \ldots, a_m \in \mathbb{R}$:

   $\sum_{i,j=1}^{m} a_i a_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0$ (0 only when $a_1 = \cdots = a_m = 0$)

---

[5] Usually called Mercer kernels

[6] Taking $\mathbf{x}_1 = \mathbf{x}_2 = \mathbf{x}$, and $a_1 = a_2 = 1$: $K(\mathbf{x}_1, \mathbf{x}_2) + K(\mathbf{x}_2, \mathbf{x}_1) = 2K(\mathbf{x}, \mathbf{x}) \geq 0$

7. Semi-definiteness is needed for the right convexity/concavity properties of the real functions under optimization, as required by Kuhn & Tucker's theory [8]

   They ensure that there are global maxima/minima (as opposed to relative)

8. The trick is that the values for K' on the (mapped) examples can be computed on the basis of their original values under K!

9. In spite of all above, the general approach is to start right away with a kernel in the space at hand, e.g. a polynomial, a Gaussian, ... kernel (see below)

   General results about (and conditions on) kernels in Hilbert spaces[7] will guarantee (or not) that the optimization problem has a solution (and in our case, that separability takes place)

[7] Specially, Riesz' theorem in functional analysis

In particular, solution vectors will be linear combinations whose coefficients depend on the kernel function

See

`http://people.scs.carleton.ca/~bertossi/trabajo/learningWconstraintsPUC14.pdf`

for more on this general use of kernels and an application to multi-task classification

10. Well-known kernels are constructed using kernel transformations/ compositions, starting from the fact that the inner product in the Euclidean space is a kernel

    Some of them:

If $H(\mathbf{x},\mathbf{y})$ and $G(\mathbf{x},\mathbf{y})$ are kernels on $\mathcal{V}$, then also the following are:

$$\begin{aligned}
K_s(\mathbf{x},\mathbf{y}) &= H(\mathbf{x},\mathbf{y}) + G(\mathbf{x},\mathbf{y})\\
K_p(\mathbf{x},\mathbf{y}) &= H(\mathbf{x},\mathbf{y}) \cdot G(\mathbf{x},\mathbf{y})\\
K_d(\mathbf{x},\mathbf{y}) &= (H(\mathbf{x},\mathbf{y}) + l)^d \quad (polynomial)\\
K_g(\mathbf{x},\mathbf{y}) &= exp(-\gamma(H(\mathbf{x},\mathbf{x}) - H(\mathbf{x},\mathbf{y}) + H(\mathbf{y},\mathbf{y}))) \qquad (Gaussian)\\
K_n(\mathbf{x},\mathbf{y}) &= \frac{H(\mathbf{x},\mathbf{y})}{\sqrt{H(\mathbf{x},\mathbf{x}) \cdot H(\mathbf{y},\mathbf{y})}} \qquad (normalized)
\end{aligned}$$

11. The SVM method can be seen as a particular form of kernel-based method

    Being SVM one of the first, and introduced in

    B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory, pages 144152. ACM Press, 1992.

Previous basis can be found in, e.g.

V. Vapnik and A. Lerner. Pattern recognition using generalized portrait method. Automation and Remote Control, 24, 1963.

V. Vapnik and A. Chervonenkis. A note on one class of perceptrons. Automation and Remote Control, 25, 1964.

12. For a historical account see [3, sec. 6.5]

13. They have been investigated in the more general context of *statistical learning*

    A whole area of ML that started with the seminal work by Vapnick et al. [10]

    With origin in statistical tasks confronted from an optimization point of view

    V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. Theory of Probability and its Applications, 16(2):264-280, 1971.

V. Vapnik. *Estimation of Dependences Based on Empirical Data*. (in Russian), Nauka, 1979. (English translation Springer Verlag, 1982).

It borrows techniques/approaches from Statistics and Functional Analysis [7], in particular, from kernel-endowed Hilbert Spaces [1]

14. The theory developed concentrates mostly in the *generalization power* of the methods, i.e. the behavior when extrapolating from (possibly few) examples

    They have good *regularization* properties

    That is confronting the *over-fitting* problem

    See [3, chap. 4] for an accessible account of the *generalization theory*; and [3, sec. 4.8] for a historical account

15. For a broad and mathematically sophisticated picture see [4]

    For a view from classical statistics (and more), see [2]

16. For applications of SVM and kernel methods to structured data, e.g. relational data, graphs, see [5, sec. 9.4]

17. For a kernel-based approach to ML with logical formulas (a form of structured data), see my previous tutorial:

    `http://people.scs.carleton.ca/~bertossi/trabajo/learningWconstraintsPUC14.pdf`

References

[1] N.I. Akhiezer and I.M. Glazman. *Theory of Linear Operators in Hilbert Space*. Dover, 1993.

[2] Leo Breiman. Statistical Modeling: The Two Cultures. *Statistical Science*, 2001, 16(3):199-231.

[3] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge U. Press, 2000.

[4] Felipe Cucker And Steve Smale. On the Mathematical Foundations of Learning. *Bulletin of the American Mathematical Society*, 2002, 39(1):1-49.

[5] Luc De Raedt. *Logical and Relational Learning*. Springer, 2010.

[6] G. James et al. *An Introduction to Satistical Learning*. Springer, 2013.

[7] E. Kreyzig. *Introductory Functional Analysis with Applications*. John Wiley & Sons, 1978.

[8] H.W. Kuhn and A.W. Tucker. Nonlinear Programming. In *Proc. 2nd Berkeley Symposium on Mathematical Statistics and Probabilistics*. U. California Press, 1951, pp. 481-492.

[9] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2002.

[10] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, 1995.