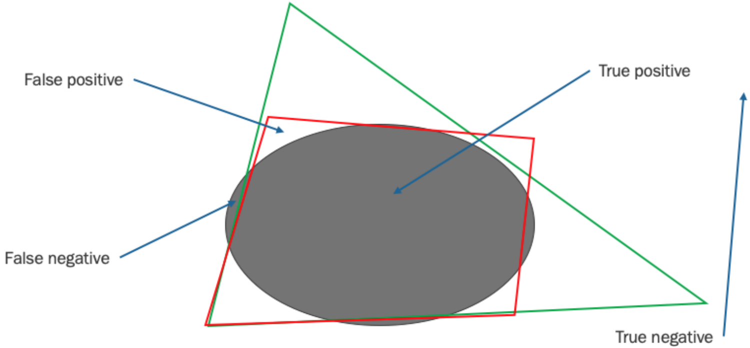
Advanced Machine Learning, Lecture 8

András Kornai

BME 2020 Oct 29

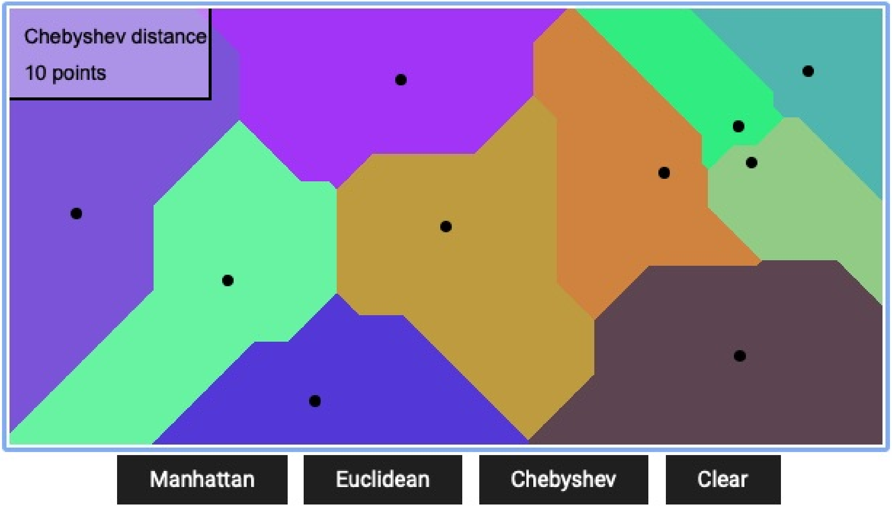
NEIGHBORHOODS

- Binary classification revisited



-
- PAC learning
- kNN classification
- Tangent distance
- Back to word vectors

Voronoi diagram



PAC LEARNING

CONCEPT

A concept c defined as a subset of $2^n(\mathbb{R}^n)$ endowed with a probability distribution π_c over c

- We only have *positive* evidence: we can always request new examples of c, which will be drawn according to π_c
- We demand that the learned function f have no false positives, and only has ϵ error
- Further, we demand that the learning process lead to such an f with probability $1-\delta$, and that it be polynomial in $1/\epsilon, 1/\delta$
- Finally, we are not interested in a single concept c, but a class of concepts C
- We say that such a class is C Probably Approximately Correctly (PAC) learnable, if there is a learning process that leads to an f_c for each $c \in C$ in polynomial time
- We owe this idea to Les Valiant, check out `valiant_1984.pdf`

PAC learning (2)

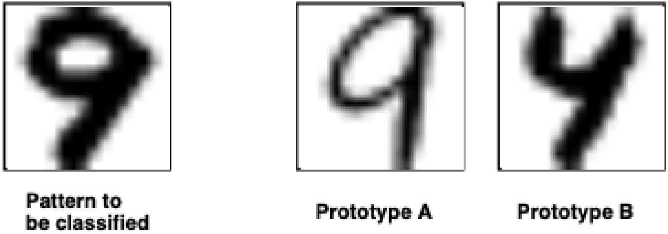
- It is the polynomial restriction that makes it hard. Otherwise, we just request new samples and define f as the disjunction of the vectors (discrete case) received so far, and sooner or later we hit coverage $1 - \epsilon$
- Let p_i be Boolean variables ($1 \leq i \leq t$), and assume the concept to be learned is p_1 in the space of monomials such as $p_2 \bar{p}_4 p_5$
- We can provide 2^{t-1} examples which all have $p_1, \ldots, p_{t/2}$ set to one, and we still haven't learned the diference between 2^{t-1} candidate monomials
- Theorem: C is PAC-learneable iff it has finite VC-dimension
- Vapnik-Chervonenkis dimension: size of maximum set C can shatter
- C shatters m if $\{m \cap c | c \in C\} = 2^m$
- HW: let C be the corners of the n-cube. What is the VC-dimension? How many samples are needed?

k NEAREST NEIGHBOR

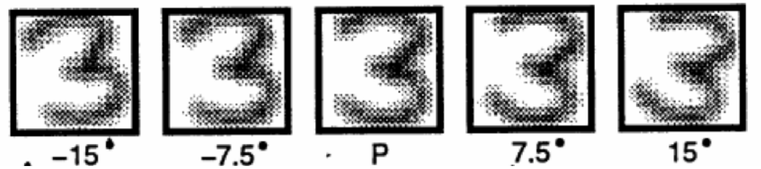
- 1NN domains are voronoi polytopes
- Next step is 3NN – don't know how to break ties
- Has regression version (predicted value is average of k nearest values), here k even is also sensible
- Dimension reduction helps, especially if the original space was large
- Lots of flavors, e.g. weighing points by distance
- HW: Project PB data down to 2 dim by PCA, look at , and generate pictures like Figs 1-3 there

Tangent distance

- Often, we don't have enough reasonable training data



-
- Euclidean distance goes wrong! One possible approach is data enrichment
- For example we may add small rotations:



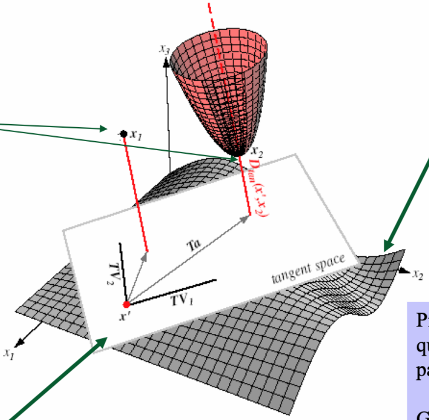
Tangent distance (2)

- Or, we may as well perform the operation at test time
- Find the degree α that produces the minimum Euclidean distance to target
- Can be done with several transformations besides rotation
- Idea comes from Simard et al `simard_1998.pdf`

Minimizing value *a* for Tangent Distance

In Euclidean space x_1 is closer to x' than x_2

In Tangent space x_2 is closer to x' than x_1



Stored prototype x' falls on this manifold when subjected to transformations

Pink paraboloid is a quadratic function of the parameter vector ***a***

Gradient descent is used to calculate Tangent distance D_{tan}(x', x2)

Tangent Space at x' is an r-dimensional Euclidean space spanned by Tangent vectors TV_1 and TV_2