PWLMI#1: Support-Vector Networks

Cortes, Vapnik (1995)

Papers We Love[SM] $f(x)=x$

Luca Ciuffreda

Milano, 24/10/2019

About me

- Theoretical Particle Physics, M.Sc. @ Unimib
- Computational Neuroscience @ SISSA, Trieste
- HPC/Deep Learning @ SISSA/ICTP/CNR-IOM, Trieste
- Data Scientist @ Prometeia, Milano

prometeia

Support-Vector Networks

CORINNA CORTES corinna@neural.att.com
VLADIMIR VAPNIK vlad@neural.att.com
AT&T Bell Labs., Holmdel, NJ 07733, USA

Editor: Lorenza Saitta

Abstract. The *support-vector network* is a new learning machine for two-group classification problems. The machine conceptually implements the following idea: input vectors are non-linearly mapped to a very high-dimension feature space. In this feature space a linear decision surface is constructed. Special properties of the decision surface ensures high generalization ability of the learning machine. The idea behind the support-vector network was previously implemented for the restricted case where the training data can be separated without errors. We here extend this result to non-separable training data.

High generalization ability of support-vector networks utilizing polynomial input transformations is demonstrated. We also compare the performance of the support-vector network to various classical learning algorithms that all took part in a benchmark study of Optical Character Recognition.

Keywords: pattern recognition, efficient learning algorithms, neural networks, radial basis function classifiers, polynomial classifiers.

Corinna **Cortes** is a Danish computer scientist known for her contributions to machine learning. She is currently the **Head of Google Research**, New York. Cortes is a recipient of the **Paris Kanellakis Theory and Practice Award** for her work on theoretical foundations of support vector machines.

Vladimir Naumovich **Vapnik** is one of the main developers of the **Vapnik–Chervonenkis** theory of **statistical learning**, and the co-inventor of the support vector machine method, and support vector clustering algorithm.

prometeia

- **Machine Learning and Classification Problems**
- **Paper Exploration**
  - [1] Introduction
  - [2] Optimal Hyperplanes
  - [3] The Soft Margin Hyperplane
  - [4] The Method of Convolution of the Dot-Product in Feature Space
  - [6] General Features of Support-Vector Networks
  - [7] Experimental Analysis
- **A Bayesian interpretation**
- **Demo**

prometeia

- **Machine Learning and Classification Problems**
- **Paper Exploration**
  - [1] Introduction
  - [2] Optimal Hyperplanes
  - [3] The Soft Margin Hyperplane
  - [4] The Method of Convolution of the Dot-Product in Feature Space
  - [6] General Features of Support-Vector Networks
  - [7] Experimental Analysis
- **A Bayesian interpretation**
- **Demo**

prometeia

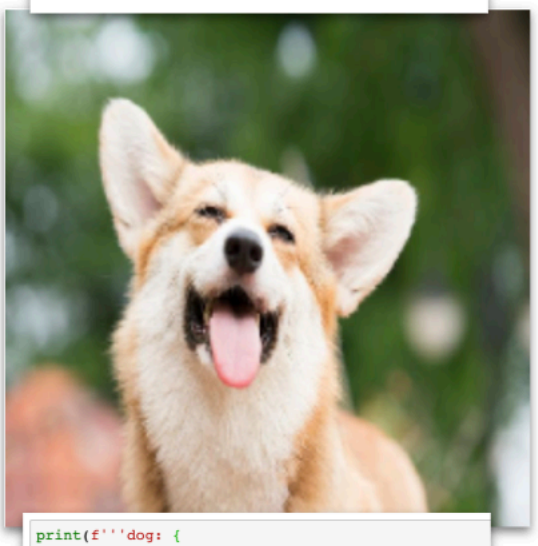
Machine Learning and Classification problems

100% cat

```
print(f'''cat: {
  np.round(model.predict(cat),2)
}''')
```

```
cat: [[1. 0. 0.]]
```

97% dog



```
print(f'''dog: {
  np.round(model.predict(dog),2)
}''')
```

```
dog: [[0.02 0.97 0.01]]
```

14% dog
85% Elon Musk

```
print(f'''elon:
  {np.round(model.predict(elon_with_disguise),2)
}''')
```

```
elon:
  [[0.   0.14 0.85]]
```

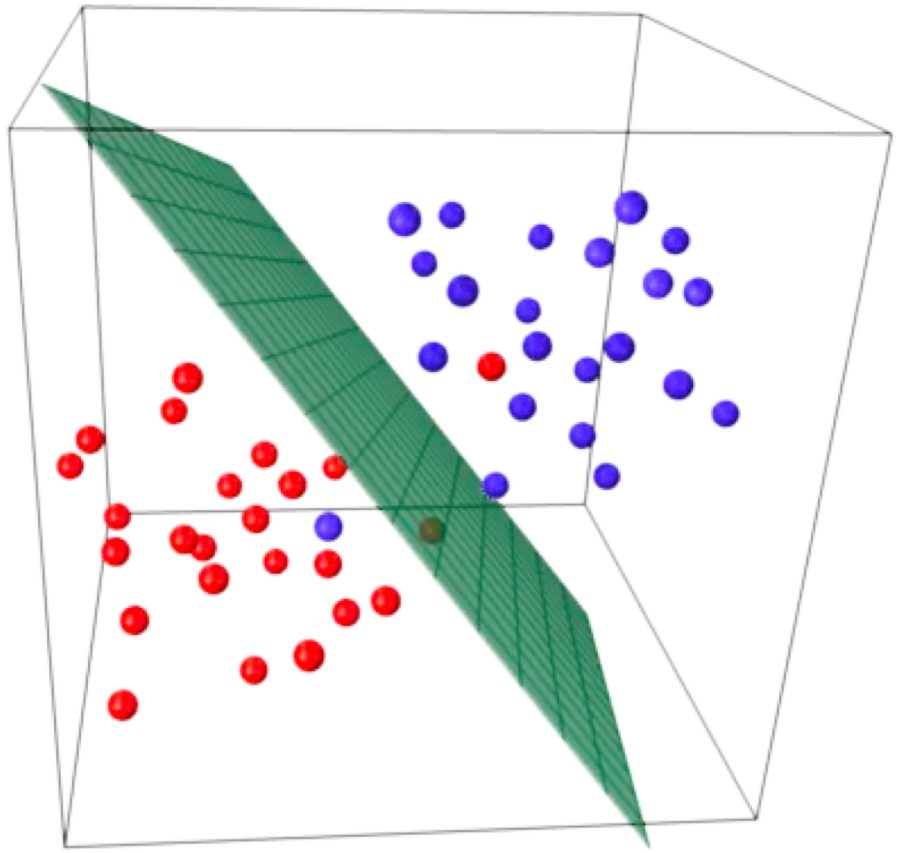
100% Elon Musk

```
print(f'''elon: {
  np.round(model.predict(elon_without_disguise),2)
}''')
```

```
elon: [[0. 0. 1.]]
```


prometeia

Machine Learning and Classification problems



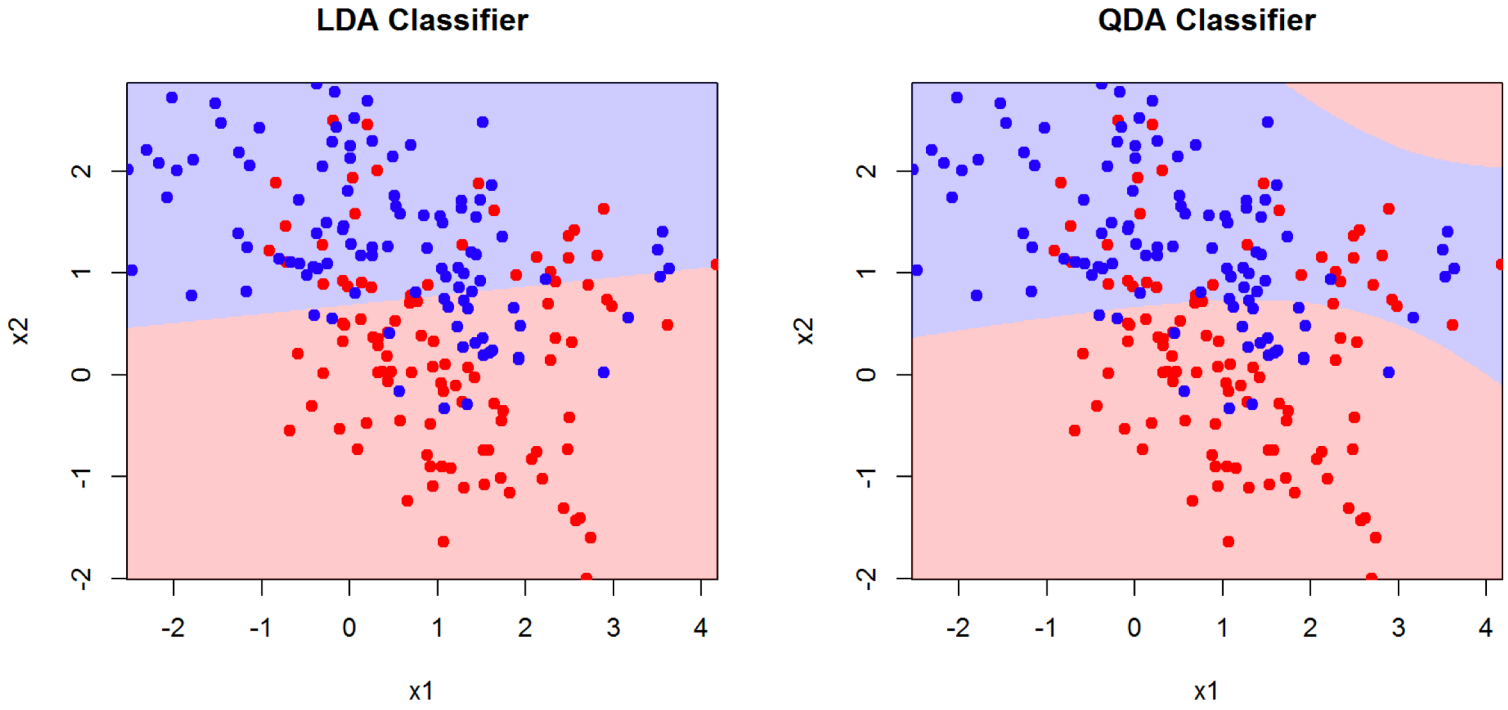
prometeia

- Machine Learning and Classification Problems
- **Paper Exploration**
  - [1] Introduction
  - [2] Optimal Hyperplanes
  - [3] The Soft Margin Hyperplane
  - [4] The Method of Convolution of the Dot-Product in Feature Space
  - [6] General Features of Support-Vector Networks
  - [7] Experimental Analysis
- A Bayesian interpretation
- Demo

prometeia

Introduction

Decision functions for normally distributed data



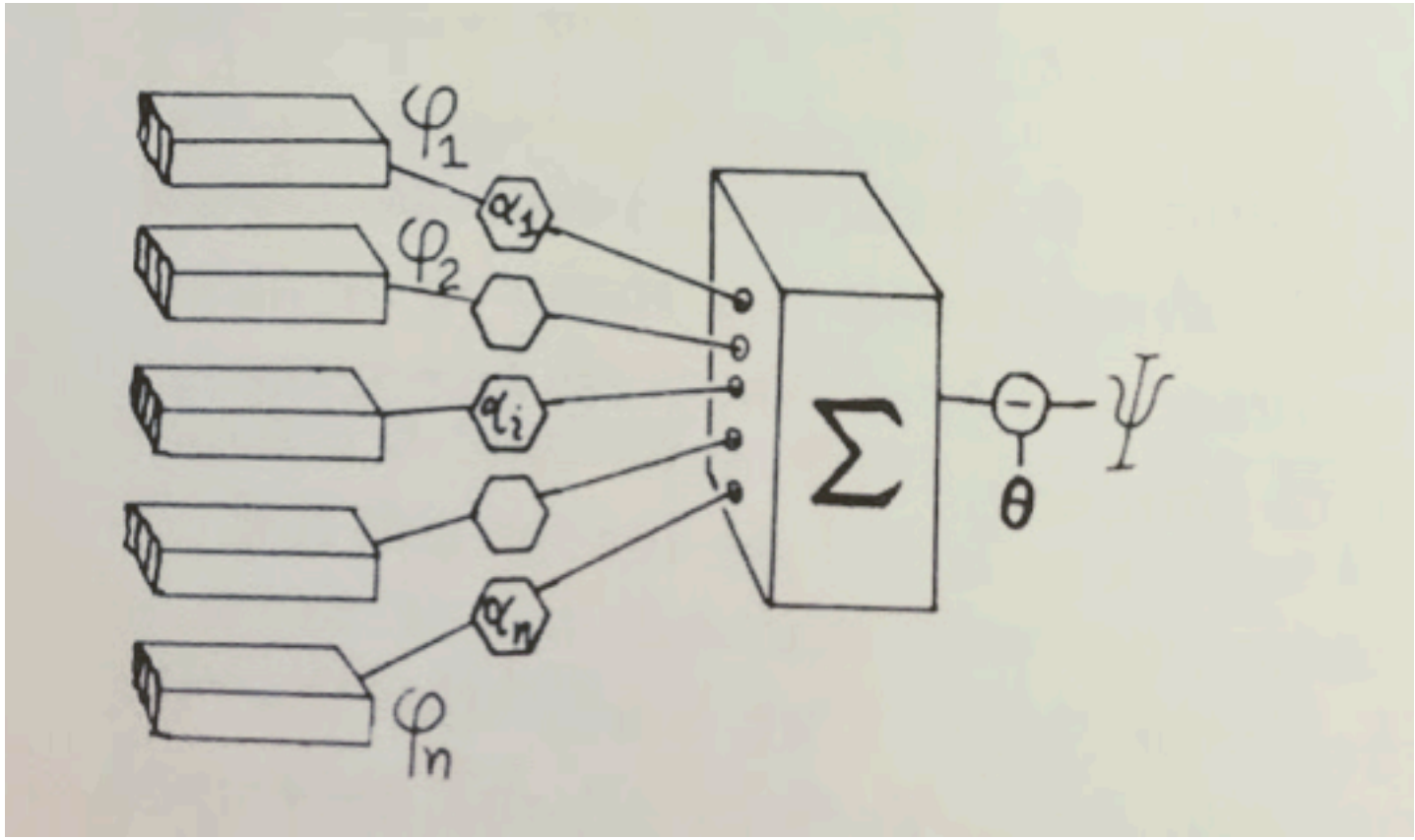
$$p(y = c|x, \theta) \sim p(c) \frac{1}{\sqrt{2\pi|\Sigma|}} \exp\left[-\frac{1}{2}(x - \mu_c)^T \Sigma_c (x - \mu_c)\right]$$

Fisher, 1936

prometeia

Introduction

Perceptron

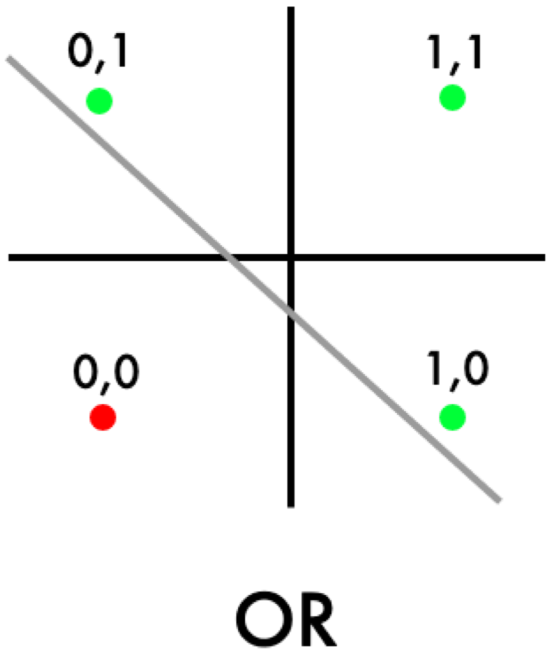


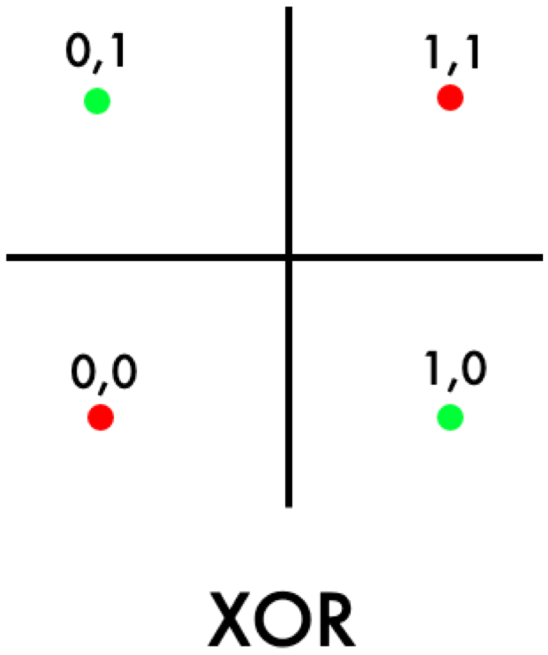
Rosenblatt, 1962

prometeia

Introduction

Perceptron – The XOR problem





prometeia

Introduction

Support-Vector Networks

In this article we construct a new type of learning machine, the so-called support-vector network. The support-vector network implements the following idea: it maps the input vectors into some high dimensional feature space Z through some non-linear mapping chosen a priori. In this space a linear decision surface is constructed with special properties that ensure high generalization ability of the network.

Introduction

Support-Vector Networks

Two problems arise in the above approach: one conceptual and one technical. The conceptual problem is how to find a separating hyperplane that will generalize well: the dimensionality of the feature space will be huge, and not all hyperplanes that separate the training data will necessarily generalize well[2]. The technical problem is how computationally to treat such high-dimensional spaces: to construct polynomial of degree 4 or 5 in a 200 dimensional space it may be necessary to construct hyperplanes in a billion dimensional feature space.

Introduction

Support-Vector Networks: How to solve these problems?

a) *How to find hyperplanes that generalize well?*
   Optimal hyperplanes and **support vectors**

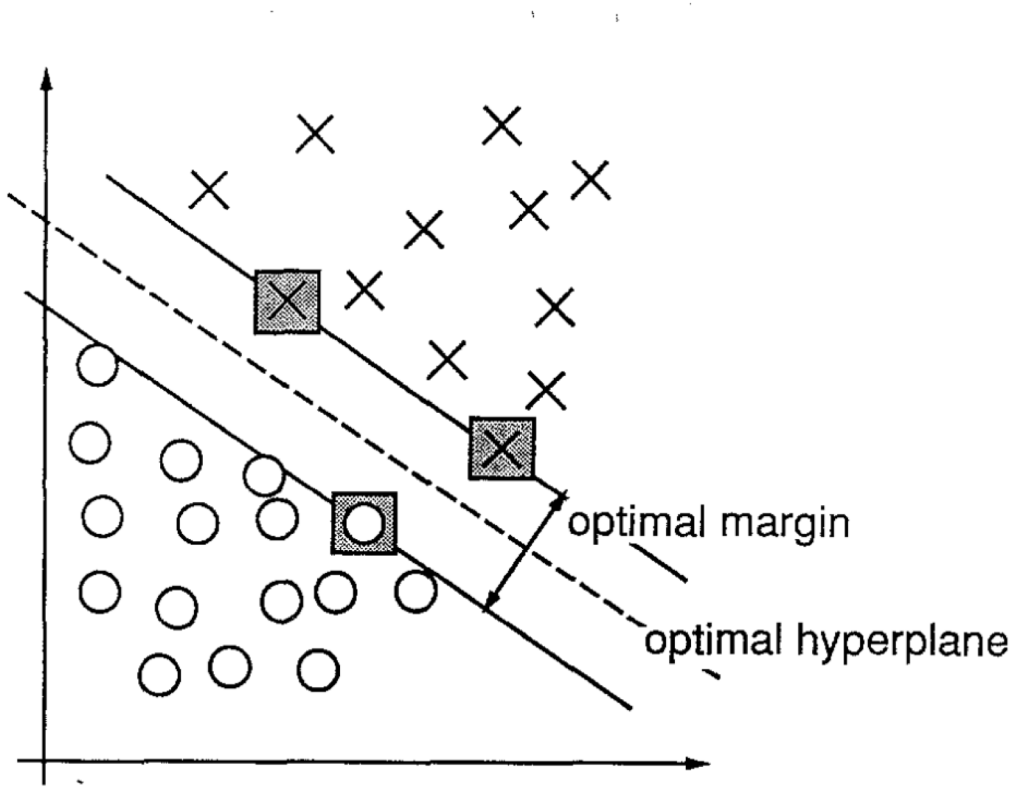
$$\mathrm{E}\left[p\left(error\,\middle|_{test}\right)\right] \leq \frac{E[\#support\ vectors]}{N}$$

b) *How do we compute in high dimensional feature spaces?*
   Kernel *trick*

- Machine Learning and Classification Problems
- **Paper Exploration**
  - [1] Introduction
  - [2] Optimal Hyperplanes
  - [3] The Soft Margin Hyperplane
  - [4] The Method of Convolution of the Dot-Product in Feature Space
  - [6] General Features of Support-Vector Networks
  - [7] Experimental Analysis
- A Bayesian interpretation
- Demo

prometeia

Optimal Hyperplanes



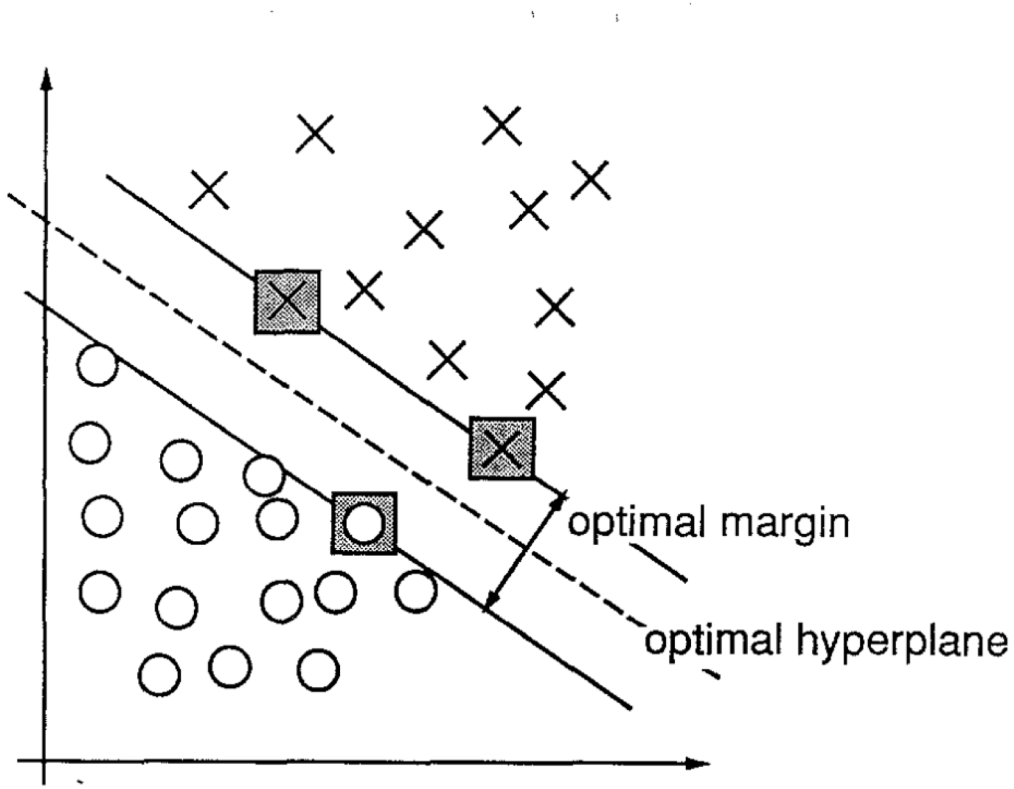
$$(x_i, y_i), \quad y_i \in \{-1,1\}$$

The dataset is ***linearly separable*** if there exist w and b such that:

$$\begin{cases} w \cdot x_i + b \geq 1 & y_i = 1 \\ w \cdot x_i + b \leq -1 & y_i = -1 \end{cases}$$

prometeia

Optimal Hyperplanes



A plane $w_0 \cdot x + b_0 = 0$ is optimal if it ***maximizes*** the ***margin***:

$$\rho(w, b) \equiv \min_{\{x:y=1\}} \frac{w \cdot x_i}{||w||} - \max_{\{x:y=-1\}} \frac{w \cdot x_i}{||w||}$$

prometeia

Optimal Hyperplanes

$$max_{\{w,b_0\}}\, \rho(w,b) \quad \text{s.t.} \quad y_i(x_i \cdot w + b) \geq 1$$

$$L(w,b;\{\alpha_i\}) = \frac{1}{2}\left|\left|w\right|\right|^2 - \sum_{i=1}^{n} \alpha_i[y_i(x_i \cdot w + b) - 1], \qquad \alpha_i \geq 0\ \forall i$$

Minimize w.r.t. (w,b)

$$L(\{\alpha_i\}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \qquad \alpha_i \geq 0\ \forall i$$

prometeia

Optimal Hyperplanes

With additional constraint (from **Karush-Kuhn-Tucker** conditions)

$$\alpha_i [y_i (w_0 \cdot x_i + b_0) - 1] = 0 \; \forall i.$$

Support vectors

$\alpha_i = 0$

$y_i (w_0 \cdot x_i + b_0) = 1$

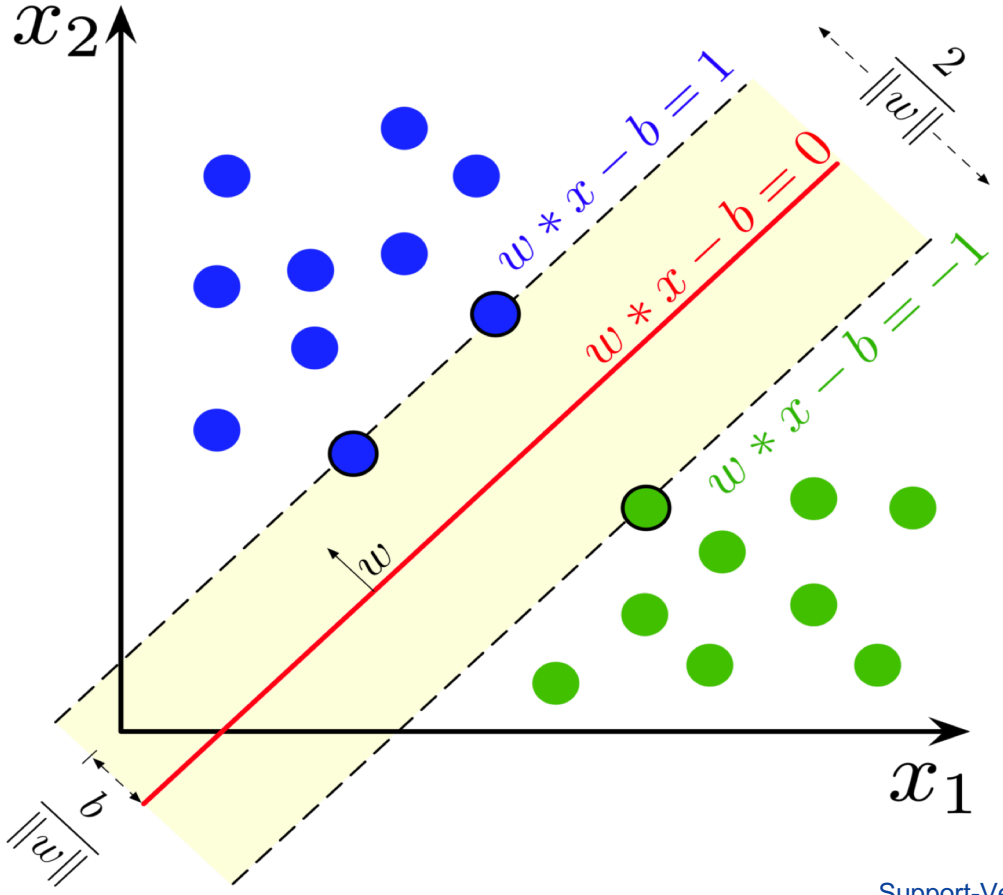
prometeia

Optimal Hyperplanes

Putting all pieces together we get the solution as the plane defined by the direction

$$w_0 = \sum_{i=1}^{n} y_i \alpha_i^0 x_i$$

The hyperplane is defined only in terms of **support vectors**.



prometeia

→ Machine Learning and Classification Problems

→ **Paper Exploration**

[1] Introduction

[2] Optimal Hyperplanes

[3] The Soft Margin Hyperplane

[4] The Method of Convolution of the Dot-Product in Feature Space

[6] General Features of Support-Vector Networks

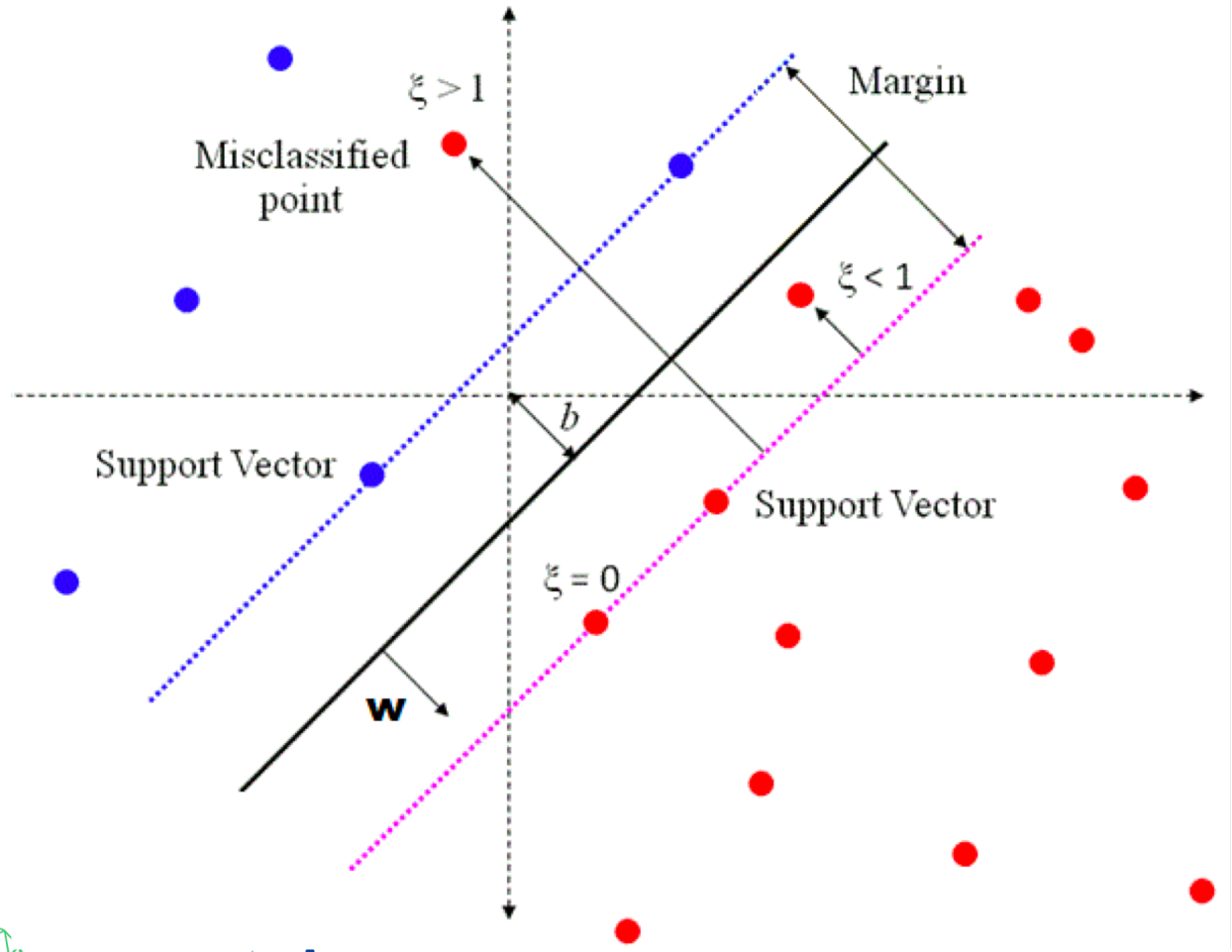
[7] Experimental Analysis

→ A Bayesian interpretation

→ Demo

prometeia

The Soft Margin Hyperplane



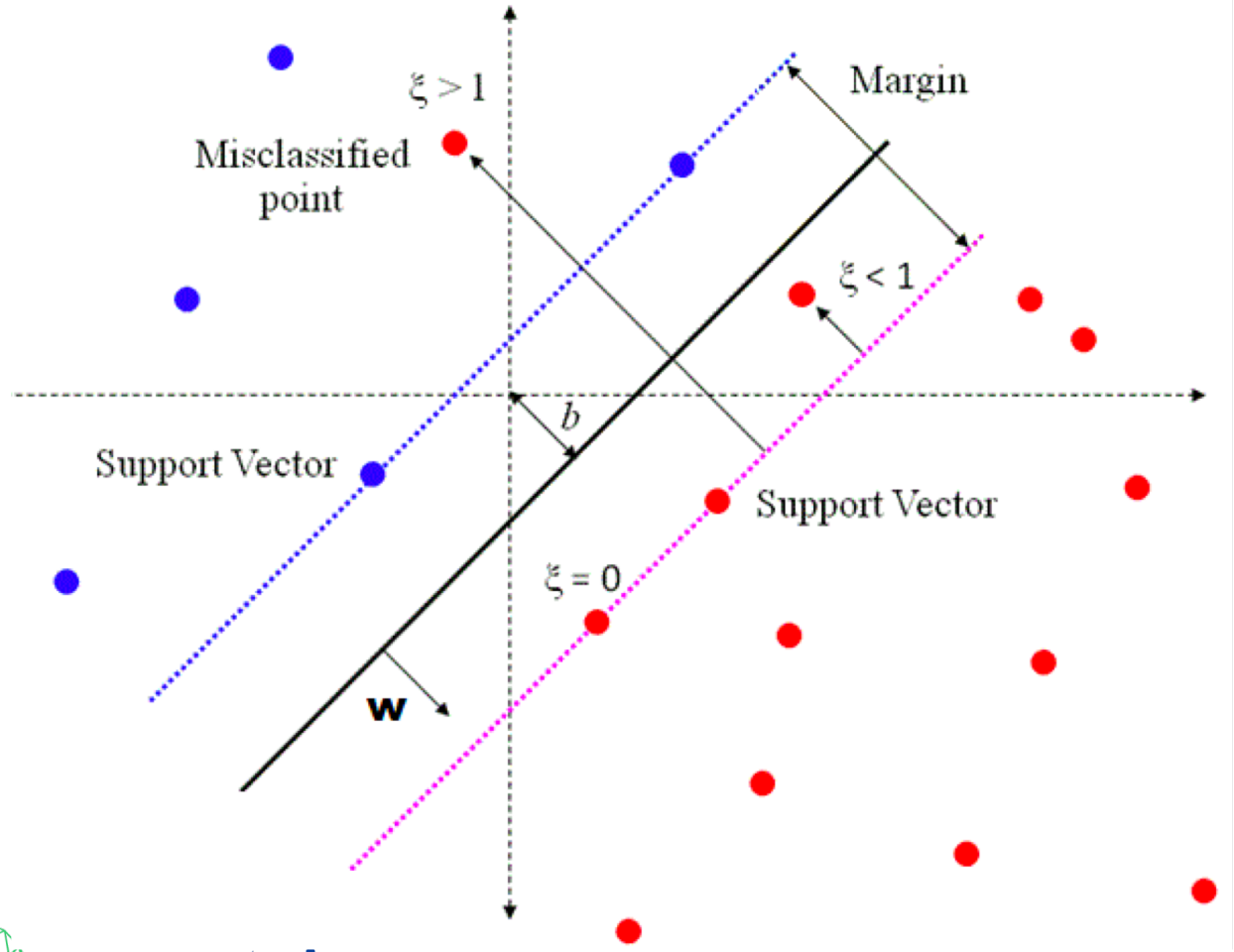
Suppose data cannot be separated perfectly by a line. We introduce a set of ***slack variables***

$$\xi_i \geq 0 \; \forall i = 1, \ldots, l$$

With $\xi_i > 0$ **only** on training errors.

prometeia

The Soft Margin Hyperplane



The functional

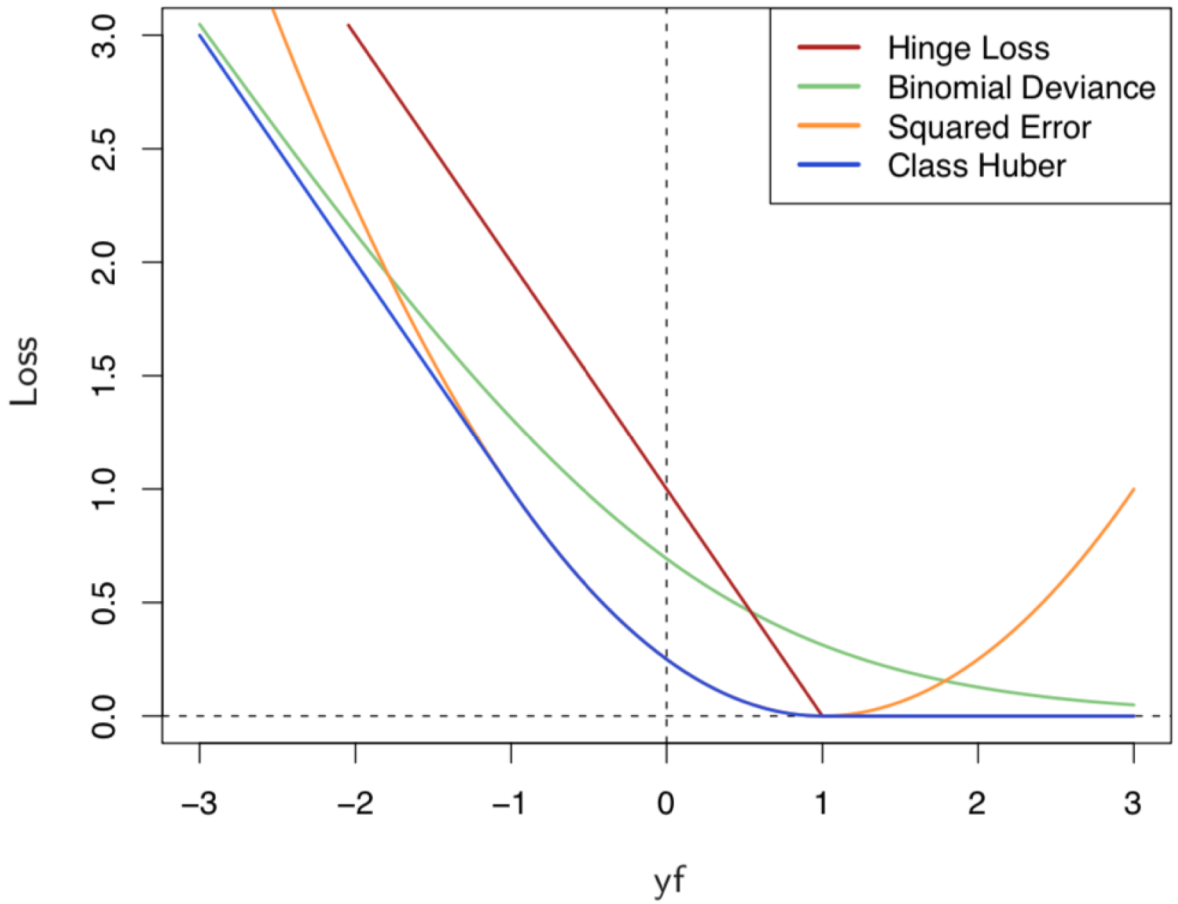
$$\Phi(\{\xi_i\}) \equiv \sum_{i=1}^{l} \xi_i^{\sigma}, \qquad \sigma > 0$$

is a ***measure of training errors***. Minimizing training errors and maximizing the margin at the same time we get the soft margin solution.

$$\min \frac{1}{2}||w||^2 + Cf\left(\sum_{i=1}^{l} \xi_i^{\sigma}\right)$$

prometeia

The Soft Margin Hyperplane



The soft margin optimization problem can be casted as the minimization of a regularized **error function**, known as the ***hinge loss***

$$\sum_{i=1}^{n} [1 - y_i f_i]_+ + \lambda \left|\left|w\right|\right|^2$$

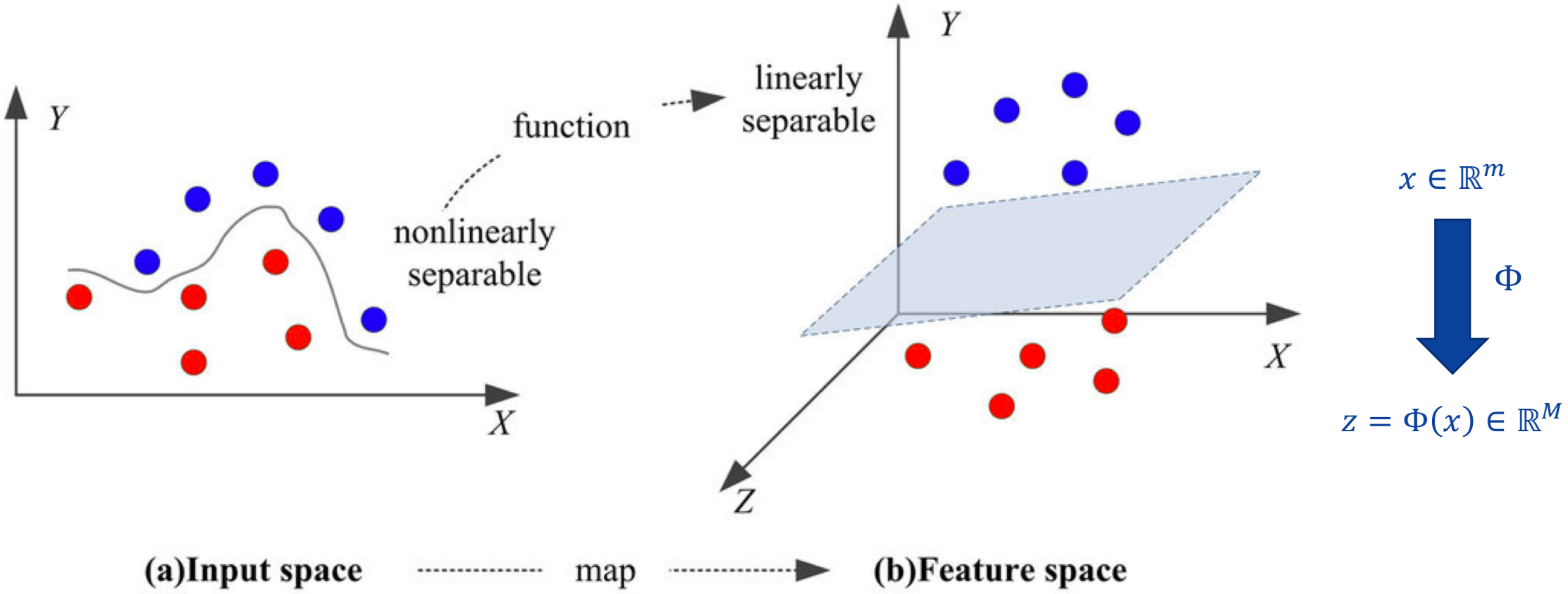
Loss function

Regularization

prometeia

- Machine Learning and Classification Problems
- **Paper Exploration**
  - [1] Introduction
  - [2] Optimal Hyperplanes
  - [3] The Soft Margin Hyperplane
  - [4] The Method of Convolution of the Dot-Product in Feature Space
  - [6] General Features of Support-Vector Networks
  - [7] Experimental Analysis
- A Bayesian interpretation
- Demo

The Method of Convolution of the Dot-Product in Feature Space



$x \in \mathbb{R}^m$

Φ

$z = \Phi(x) \in \mathbb{R}^M$

prometeia

The Method of Convolution of the Dot-Product in Feature Space

In feature space we can look for **optimal separating hyperplanes**,

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

Dot-Product $\Phi(x) \cdot \Phi(y) \equiv K(x, y)$ Kernel function

Mercer's Theorem

prometeia

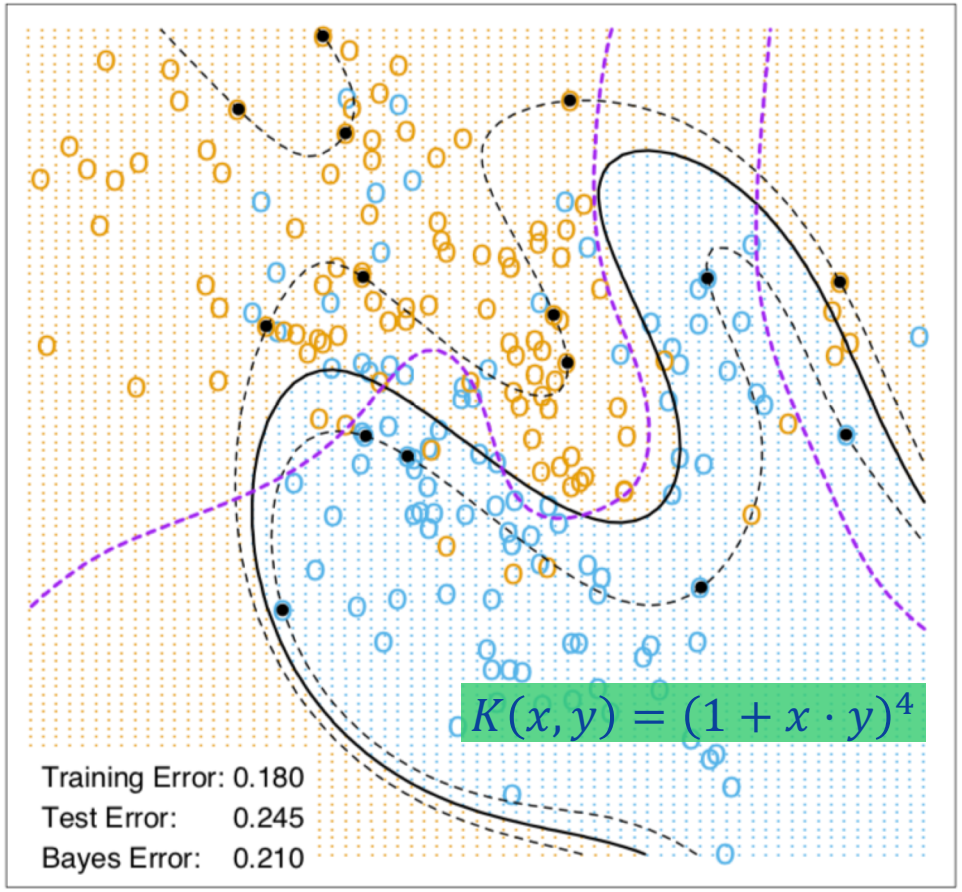
The Method of Convolution of the Dot-Product in Feature Space

Some examples of kernel functions are

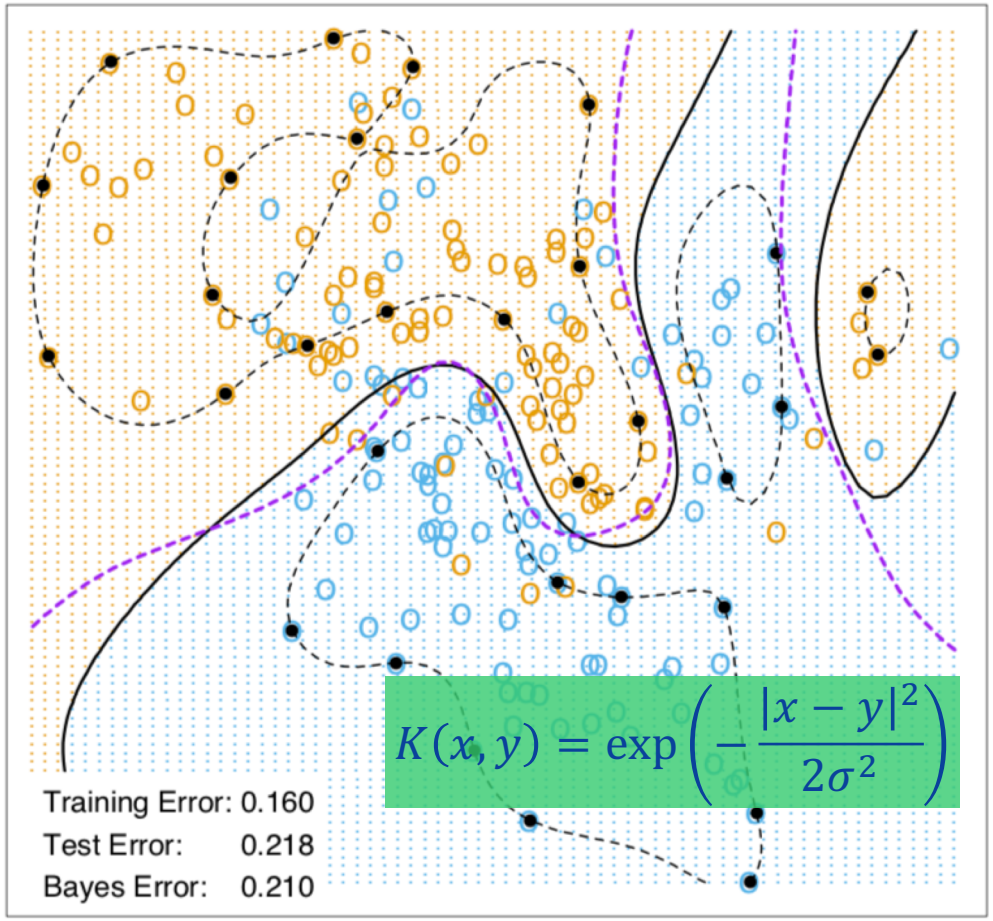
$K(x,y) = \exp\left(-\frac{\lvert x-y\rvert}{\sigma}\right)$	Potential function
$K(x,y) = \exp\left(-\frac{\lvert x-y\rvert^2}{2\sigma^2}\right)$	**Radial basis**
$K(x,y) = (1 + x \cdot y)^d.$	**Polynomial** of degree *d*
$K(x,y) = \tanh(\kappa_1 x \cdot y + \kappa_2)$	Neural network-like

The Method of Convolution of the Dot-Product in Feature Space

SVM - Degree-4 Polynomial in Feature Space



SVM - Radial Kernel in Feature Space



prometeia

- → Machine Learning and Classification Problems
- → **Paper Exploration**
  - [1] Introduction
  - [2] Optimal Hyperplanes
  - [3] The Soft Margin Hyperplane
  - [4] The Method of Convolution of the Dot-Product in Feature Space
  - [6] General Features of Support-Vector Networks
  - [7] Experimental Analysis
- → A Bayesian interpretation
- → Demo

General Features of Support-Vector Networks

Constructing the Decision Rules by Support-Vector Networks is Efficient

The decision boundary is found by optimizing a **quadratic programming problem** with simple constraints,

$$L(\{\alpha_i\}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \qquad \alpha_i \geq 0 \ \forall i$$

This can be solved with **efficient algorithms** such as ***chunking*** (Vapnik, 1982), ***decomposition methods*** (Osuna, 1996) or ***sequential minimal optimization*** (Platt, 1999).

General Features of Support-Vector Networks

The Support-Vector Network is a Universal Machine

Choice of the kernel function $K(x, y)$ is **arbitrary** and allows for **general formulations** of the learning algorithm. In this sense SVMs are *a rather* ***general class of learning machines***.

General Features of Support-Vector Networks

Support-Vector Networks and Control of Generalization Ability

$$p\left(error\Big|_{test}\right) \leq p\left(error\Big|_{train}\right) + D[l, h, \eta]$$

with probability $1 - \eta$.

Where:

- l is the dimension of training set
- h is the VC dimension (**complexity** of the model)
- D is also known as **VC confidence**

(Vapnik, 1982)

General Features of Support-Vector Networks

Support-Vector Networks and Control of Generalization Ability

$$p\left(error\Big|_{test}\right) \leq p\left(error\Big|_{train}\right) + D[l, h, \eta]$$

This result is basically a **trade-off** between the **complexity** of the model and the **error rate**.

SVMs can control both terms adapting the parameter C, and thus ***reducing the impact of overfitting***.

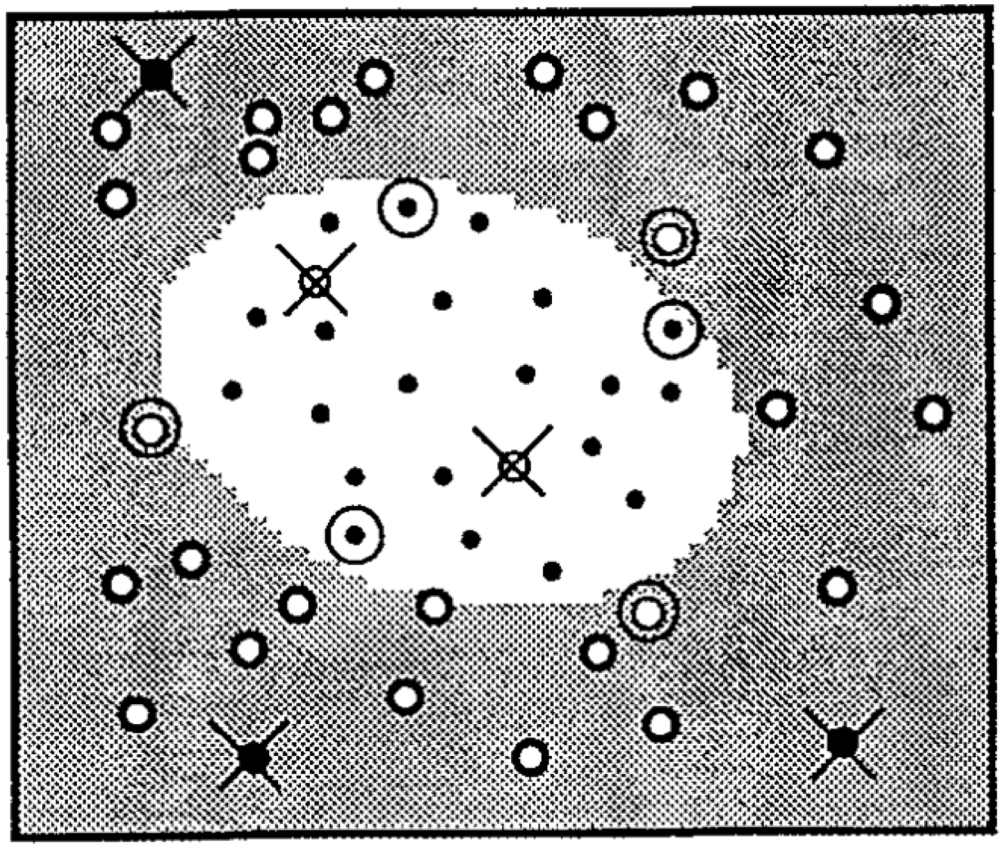
- Machine Learning and Classification Problems
- **Paper Exploration**
  - [1] Introduction
  - [2] Optimal Hyperplanes
  - [3] The Soft Margin Hyperplane
  - [4] The Method of Convolution of the Dot-Product in Feature Space
  - [6] General Features of Support-Vector Networks
  - [7] Experimental Analysis
- A Bayesian interpretation
- Demo

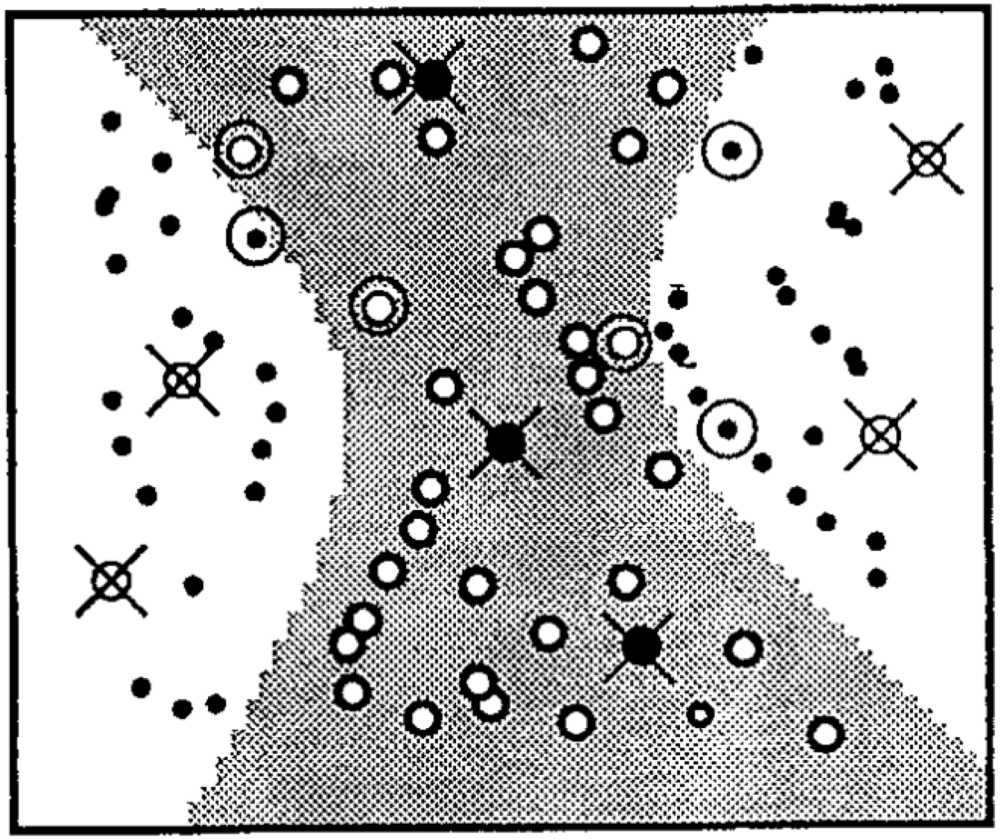
prometeia

Experimental Analysis

Experiments in the Plane

$$K(x, y) = (1 + x \cdot y)^2$$

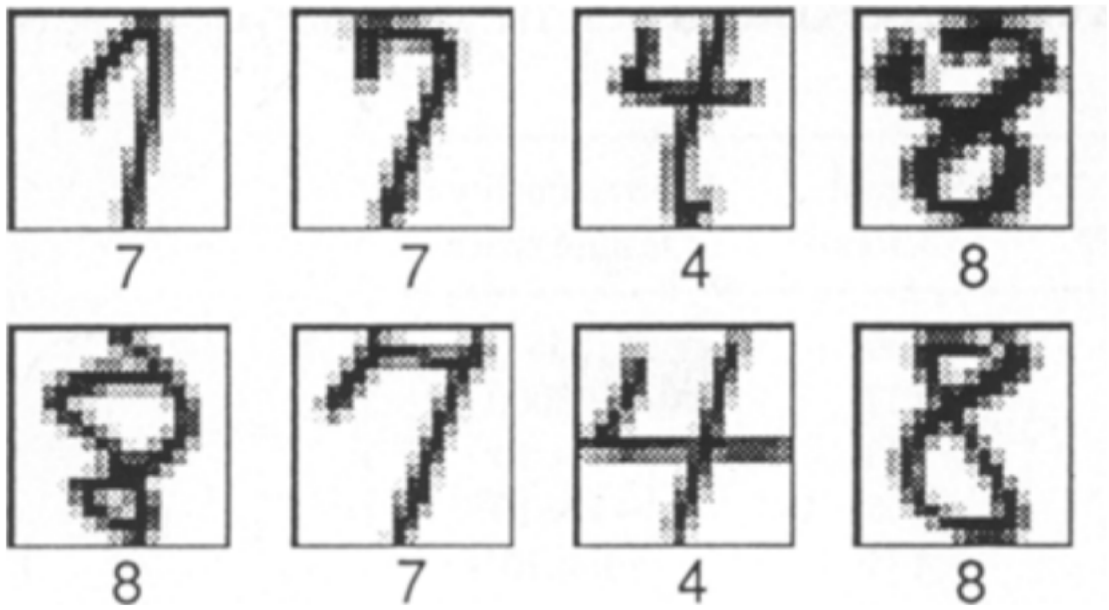




prometeia

Experimental Analysis

Experiments with Digit Recognition



Dataset 1: **US Postal Service database**

16 x 16 pixels
7300 training examples, 2000 test

prometeia

Experimental Analysis

Experiments with Digit Recognition

Classifier	Raw error, %
Human performance	2.5
Decision tree, CART	17
Decision tree, C4.5	16
Best 2 layer neural network	6.6
Special architecture 5 layer network	5.1

Degree of polynomial	Raw error, %	Support vectors	Dimensionality of feature space
1	12.0	200	256
2	4.7	127	~33000
3	4.4	148	$\sim 1 \times 10^6$
4	4.3	165	$\sim 1 \times 10^9$
5	4.3	175	$\sim 1 \times 10^{12}$
6	4.2	185	$\sim 1 \times 10^{14}$
7	4.3	190	$\sim 1 \times 10^{16}$

Experimental Analysis

Experiments with Digit Recognition

Classifier	Raw error, %
Human performance	2.5
Decision tree, CART	17
Decision tree, C4.5	16
Best 2 layer neural network	6.6
Special architecture 5 layer network	5.1

LeNet1 (LeCun, 1990)

Experimental Analysis

Experiments with Digit Recognition

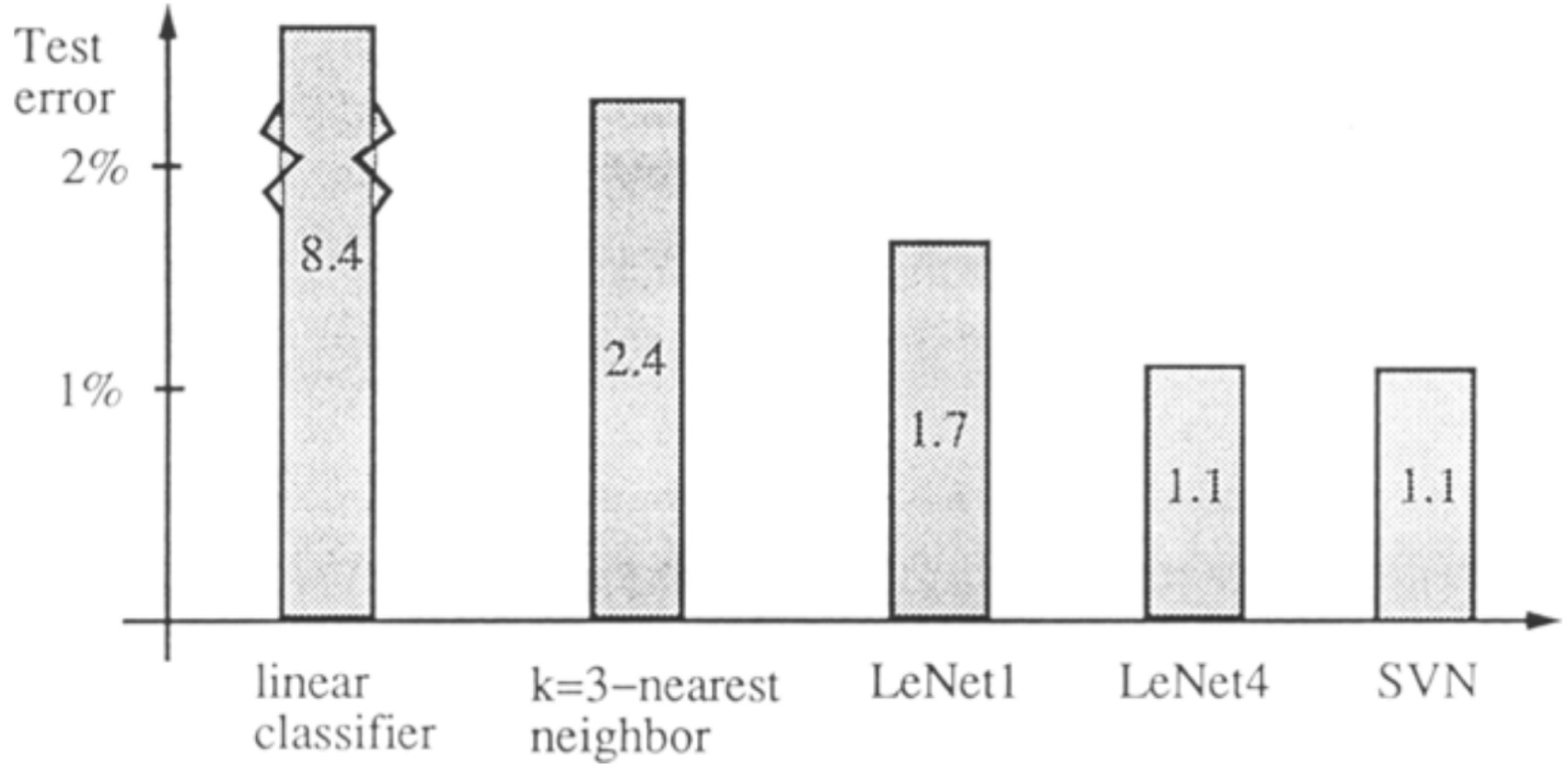
Dataset 2: **NIST database**

28 x 28 pixels
60000 training examples, 10000 test

prometeia

Experimental Analysis

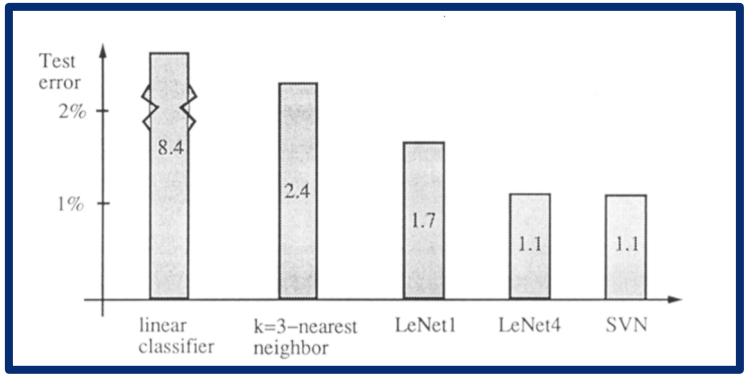
Experiments with Digit Recognition



prometeia

Experimental Analysis

Experiments with Digit Recognition



Rank	Method	Percentage error	Accuracy	Paper Title	Year	Paper	Code
1	RMDL	0.18		RMDL: Random Multimodel Deep Learning for Classification	2018		
2	MCDNN	0.2		Multi-column Deep Neural Networks for Image Classification	2012		
3	DropConnect	0.2					
4	APAC	0.2		APAC: Augmented PAttern Classification with Neural Networks	2015		
5	BNM NiN	0.2		Batch-normalized Maxout Network in Network	2015		

prometeia

- → Machine Learning and Classification Problems
- → Paper Exploration
  - [1] Introduction
  - [2] Optimal Hyperplanes
  - [3] The Soft Margin Hyperplane
  - [4] The Method of Convolution of the Dot-Product in Feature Space
  - [6] General Features of Support-Vector Networks
  - [7] Experimental Analysis
- → **A Bayesian interpretation**
- → Demo

A Bayesian Interpretation

Henao, 2014

$$\arg\min_f \gamma R[f] + \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ \qquad (1)$$

Regularization Hinge Loss

If f is linear (1) is equivalent to

$$p(w, b|D) \sim \prod_{i=1}^{N} L(y_i|x_i, w, b)p(w, b)$$

$$L(y_i|x_i, w, b) = \int_0^\infty \frac{d\lambda_i}{\sqrt{2\pi\lambda_i}} \exp\left(-\frac{1}{2}\frac{(1+\lambda_i-y_i x_i w)^2}{\lambda_i}\right)$$

$$\log p(w, b) \sim -2\gamma R(w, b)$$

- → Machine Learning and Classification Problems
- → Paper Exploration
  - [1] Introduction
  - [2] Optimal Hyperplanes
  - [3] The Soft Margin Hyperplane
  - [4] The Method of Convolution of the Dot-Product in Feature Space
  - [6] General Features of Support-Vector Networks
  - [7] Experimental Analysis
- → A Bayesian interpretation
- → **Demo**

References

Fisher, R. A. (1936): *The Use of Multiple Measurements in Taxonomic Problems*. *Annals of Eugenics*. **7** (2): 179–188

Rosenblatt, F. (1962): *Principles of neurodynamics - perceptrons and the theory of brain mechanisms*.

LeCun, Y. et Al. (1990): *Handwritten digit recognition with a back-propagation network.* Advances in Neural Information Processing Systems (NIPS 1989)

Vapnik, V. (1982): *Estimation of Dependences Based on Empirical Data*.

Henao, R. et Al. (2014): *Bayesian Nonlinear Support Vector Machines and Discriminative Factor Modeling.* Advances in Neural Information Processing Systems (NIPS 2014)

References

LIBSVM: https://www.csie.ntu.edu.tw/~cjlin/libsvm/#GUI

Rémi Flamary - **SVM and regularization:** https://remi.flamary.com/demos/svmreg.html

Cristian Dima - **Basics of support vector machines:** http://www.cristiandima.com/basics-of-support-vector-machines/