Introduction to Machine Learning

4. Perceptron and Kernels

Alex Smola
Carnegie Mellon University

http://alex.smola.org/teaching/cmu2013-10-701
10-701
Outline

- Perceptron
  - Hebbian learning & biology
  - Algorithm
  - Convergence analysis
- Features and preprocessing
  - Nonlinear separation
  - Perceptron in feature space
- Kernels
  - Kernel trick
  - Properties
  - Examples
Perceptron

Frank Rosenblatt
early theories of the brain
Biology and Learning

• Basic Idea
  • Good behavior should be rewarded, bad behavior punished (or not rewarded). This improves system fitness.
  • Killing a sabertooth tiger should be rewarded ...
  • Correlated events should be combined.
  • Pavlov’s salivating dog.

• Training mechanisms
  • Behavioral modification of individuals (learning)
    Successful behavior is rewarded (e.g. food).
  • Hard-coded behavior in the genes (instinct)
    The wrongly coded animal does not reproduce.
Neurons

- **Soma (CPU)**
  Cell body - combines signals

- **Dendrite (input bus)**
  Combines the inputs from several other nerve cells

- **Synapse (interface)**
  Interface and parameter store between neurons

- **Axon (cable)**
  May be up to 1m long and will transport the activation signal to neurons at different locations
Neurons

\[ f(x) = \sum_i w_i x_i = \langle w, x \rangle \]
Perceptron

- Weighted linear combination
- Nonlinear decision function
- Linear offset (bias)
- Linear separating hyperplanes (spam/ham, novel/typical, click/no click)
- Learning
  Estimating the parameters \( w \) and \( b \)

\[
f(x) = \sigma (\langle w, x \rangle + b)
\]
Perceptron

Ham

Spam
The Perceptron

initialize \( w = 0 \) and \( b = 0 \)

repeat
    if \( y_i \left[ \langle w, x_i \rangle + b \right] \leq 0 \) then
        \( w \leftarrow w + y_i x_i \) and \( b \leftarrow b + y_i \)
    end if
until all classified correctly

• Nothing happens if classified correctly
• Weight vector is linear combination \( w = \sum_{i \in I} y_i x_i \)
• Classifier is linear combination of inner products \( f(x) = \sum_{i \in I} y_i \langle x_i, x \rangle + b \)
Convergence Theorem

- If there exists some \((w^*, b^*)\) with unit length and
  \[ y_i \left(\langle x_i, w^* \rangle + b^* \right) \geq \rho \]  
  for all \(i\)

then the perceptron converges to a linear separator after a number of steps bounded by

\[
\left( b^{*2} + 1 \right) \left( r^2 + 1 \right) \rho^{-2} \text{ where } \|x_i\| \leq r
\]

- Dimensionality independent
- Order independent (i.e. also worst case)
- Scales with ‘difficulty’ of problem
Starting Point
We start from $w_1 = 0$ and $b_1 = 0$.

Step 1: Bound on the increase of alignment
Denote by $w_i$ the value of $w$ at step $i$ (analogously $b_i$).

Alignment: $\langle (w_i, b_i), (w^*, b^*) \rangle$

For error in observation $(x_i, y_i)$ we get

\[
\langle (w_{j+1}, b_{j+1}) \cdot (w^*, b^*) \rangle \\
= \langle [(w_j, b_j) + y_i(x_i, 1)] , (w^*, b^*) \rangle \\
= \langle (w_j, b_j) , (w^*, b^*) \rangle + y_i \langle (x_i, 1) \cdot (w^*, b^*) \rangle \\
\geq \langle (w_j, b_j) , (w^*, b^*) \rangle + \rho \\
\geq j_\rho.
\]

Alignment increases with number of errors.
Proof

Step 2: Cauchy-Schwartz for the Dot Product

$$\langle (w_{j+1}, b_{j+1}) \cdot (w^*, b^*) \rangle \leq \|(w_{j+1}, b_{j+1})\| \|(w^*, b^*)\|$$

$$= \sqrt{1 + (b^*)^2}\|(w_{j+1}, b_{j+1})\|$$

Step 3: Upper Bound on $\|(w_j, b_j)\|$,

If we make a mistake we have

$$\|(w_{j+1}, b_{j+1})\|^2 = \|(w_j, b_j) + y_i(x_i, 1)\|^2$$

$$= \|(w_j, b_j)\|^2 + 2y_i\langle (x_i, 1), (w_j, b_j) \rangle + \|(x_i, 1)\|^2$$

$$\leq \|(w_j, b_j)\|^2 + \|(x_i, 1)\|^2$$

$$\leq j(R^2 + 1).$$

Step 4: Combination of first three steps

$$j\rho \leq \sqrt{1 + (b^*)^2}\|(w_{j+1}, b_{j+1})\| \leq \sqrt{j(R^2 + 1)((b^*)^2 + 1)}$$

Solving for $j$ proves the theorem.
Consequences

• Only need to store errors. This gives a compression bound for perceptron.

• Stochastic gradient descent on hinge loss

\[ l(x_i, y_i, w, b) = \max(0, 1 - y_i (\langle w, x_i \rangle + b)) \]

• Fails with noisy data

do NOT train your avatar with perceptrons
Hardness
margin vs. size

hard

easy
Concepts & version space

- Realizable concepts
  - Some function exists that can separate data and is included in the concept space
  - For perceptron - data is linearly separable
- Unrealizable concept
  - Data not separable
  - We don’t have a suitable function class (often hard to distinguish)
Minimum error separation

- XOR - not linearly separable
- Nonlinear separation is trivial
- Caveat (Minsky & Papert)

Finding the minimum error linear separator is NP hard (this killed Neural Networks in the 70s).
Nonlinearity & Preprocessing
Nonlinear Features

- Regression
  We got nonlinear functions by preprocessing
- Perceptron
- Map data into feature space $x \rightarrow \phi(x)$
- Solve problem in this space
- Query replace $\langle x, x' \rangle$ by $\langle \phi(x), \phi(x') \rangle$ for code
- Feature Perceptron
- Solution in span of $\phi(x_i)$
Quadratic Features

- Separating surfaces are Circles, hyperbolae, parabolae
### Constructing Features
(very naive OCR system)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loops</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3 Joints</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4 Joints</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Angles</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Ink</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Feature Engineering for Spam Filtering

- bag of words
- pairs of words
- date & time
- recipient path
- IP number
- sender
- encoding
- links
- ... secret sauce ...

Delivered-To: alex.smola@gmail.com
Received: by 10.216.47.73 with SMTP id s51cs361171web;
  Tue, 3 Jan 2012 14:17:53 -0800 (PST)
Received: by 10.213.17.145 with SMTP id s17mr2519891eba.147.1325629071725;
  Tue, 3 Jan 2012 14:17:51 -0800 (PST)
Return-Path: alex+cuf=aalex.smola@gmail.com@smola.org
Received: from mail-ey0-f175.google.com (mail-ey0-f175.google.com [209.85.215.175])
  by mx.google.com with ESMTP id n4si292642323eeef.57.2012.01.03.14.17.51
  (version=TLSv1/SSLv3 cipher=OTHER);
  Tue, 3 Jan 2012 14:17:51 -0800 (PST)
Received-SPF: neutral (google.com: 209.85.215.175 is neither permitted nor denied by best
  guess record for domain of alex+cuf=aalex.smola@gmail.com@smola.org) client-ip=209.85.215.175;
Authentication-Results: mx.google.com; spf=neutral (google.com: 209.85.215.175 is neither
permitted nor denied by best guess record for domain of alex+cuf=aalex.smola@gmail.com@smola.org)
  smtp.mail=alex+cuf=aalex.smola@gmail.com@smola.org;
dkim=pass (test mode) header.i=@googlemail.com
Received: by eaal1 with SMTP id l1so15092746eaa.6
  for <alex.smola@gmail.com>; Tue, 03 Jan 2012 14:17:51 -0800 (PST)
Received: by 10.205.135.18 with SMTP id ie18mr5325064bkc.72.1325629071362;
  Tue, 03 Jan 2012 14:17:51 -0800 (PST)
X-Forewarded-To: alex.smola@gmail.com
X-Forewarded-For: alex@smola.org
Delivered-To: alex@smola.org
Received: by 10.204.65.198 with SMTP id k6cs206093bki;
  Tue, 3 Jan 2012 14:17:50 -0800 (PST)
Received: by 10.52.88.179 with SMTP id bh19mr10729402vdb.38.1325629068795;
  Tue, 03 Jan 2012 14:17:48 -0800 (PST)
Return-Path: <althoff.tim@googlemail.com>
Received: from mail-vx0-f179.google.com (mail-vx0-f179.google.com [209.85.220.179])
  by mx.google.com with ESMTPS id dt4si11767074vdb.93.2012.01.03.14.17.48
  (version=TLSv1/SSLv3 cipher=OTHER);
  Tue, 03 Jan 2012 14:17:48 -0800 (PST)
Received-SPF: pass (google.com: domain of althoff.tim@googlemail.com designates
209.85.220.179 as permitted sender) client-ip=209.85.220.179;
Received: by vcbf13 with SMTP id f13so11295098vcb.10
  for <alex@smola.org>; Tue, 03 Jan 2012 14:17:48 -0800 (PST)
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
  d=googlemail.com; s=gamma;
  h=mime-version:sender:date:x-google-sender-auth:message-id:subject
  :from:to:content-type;
  bh=WCbdZ5sXac25dpH02XcRyDOdts993KhwsAVxpGrFh0w=;
b=WK2B2+ExWnf/gyKtwu6uVkp4XeoKlNJq3USYtM0RARK8dSfjyq0qsIHeAP9yssxp60
  7ngGtoZyq4ZsyjFvqclAWp1CPJCjH8AMcnqWxxONMoeoFv1p2Hz0o2zxS0CxS5Ry7qX
  Ulbdna4U0DxJ6UEf65pLDCkptd8023gr7o=
MIME-Version: 1.0
Received: by 10.220.108.81 with SMTP id e17mr24104004vcp.67.1325629067787;
  Tue, 03 Jan 2012 14:17:47 -0800 (PST)
Sender: althoff.tim@googlemail.com
Received: by 10.220.17.129 with HTTP; Tue, 3 Jan 2012 14:17:47 -0800 (PST)
Date: Tue, 3 Jan 2012 14:17:47 -0800
X-Google-Sender-Author: 6bw160D17HYpZllx0Eo138NZzrYeHs
Message-ID: <CAFJH0GPRKq+5d2zYMDMAB1kKvynj9KteM0B1jY6joG0-WC7osm@mail.gmail.com>
Subject: CS 281B. Advanced Topics in Learning and Decision Making
From: Tim Althoff <althoff@eecs.berkeley.edu>
To: alex@smola.org
Content-Type: multipart/alternative; boundary=f46d043c7af4b078ed04b5a7113a
--f46d043c7af4b078ed04b5a7113a
Content-Type: text/plain; charset=ISO-8859-1
More feature engineering

- Two Interlocking Spirals
  Transform the data into a radial and angular part
  \[(x_1, x_2) = (r \sin \phi, r \cos \phi)\]

- Handwritten Japanese Character Recognition
  - Break down the images into strokes and recognize it
  - Lookup based on stroke order

- Medical Diagnosis
  - Physician’s comments
  - Blood status / ECG / height / weight / temperature ...
  - Medical knowledge

- Preprocessing
  - Zero mean, unit variance to fix scale issue (e.g. weight vs. income)
  - Probability integral transform (inverse CDF) as alternative
The Perceptron on features

initialize $w, b = 0$

repeat
   Pick $(x_i, y_i)$ from data
   if $y_i(w \cdot \Phi(x_i) + b) \leq 0$ then
      $w' = w + y_i \Phi(x_i)$
      $b' = b + y_i$
   until $y_i(w \cdot \Phi(x_i) + b) > 0$ for all $i$

• Nothing happens if classified correctly
• Weight vector is linear combination $w = \sum_{i \in I} y_i \phi(x_i)$
• Classifier is linear combination of inner products $f(x) = \sum_{i \in I} y_i \langle \phi(x_i), \phi(x) \rangle + b$
Problems

• Problems
  • Need domain expert (e.g. Chinese OCR)
  • Often expensive to compute
  • Difficult to transfer engineering knowledge
• Shotgun Solution
  • Compute many features
  • Hope that this contains good ones
  • Do this efficiently
Kernels

Grace Wahba
Solving XOR

- XOR not linearly separable
- Mapping into 3 dimensions makes it easily solvable
Quadratic Features

Quadratic Features in $\mathbb{R}^2$

$$\Phi(x) := \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right)$$

Dot Product

$$\langle \Phi(x), \Phi(x') \rangle = \langle \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right), \left(x_1'^2, \sqrt{2}x_1'x_2', x_2'^2\right) \rangle$$

$$= \langle x, x' \rangle^2.$$

Insight

Trick works for any polynomials of order $d$ via $\langle x, x' \rangle^d$. 
SVM with a polynomial Kernel visualization

Created by: Udi Aharoni
Problem
- Extracting features can sometimes be very costly.
- Example: second order features in 1000 dimensions. This leads to $5 \cdot 10^5$ numbers. For higher order polynomial features much worse.

Solution
Don’t compute the features, try to compute dot products implicitly. For some features this works . . .

Definition
A kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a symmetric function in its arguments for which the following property holds

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle$$ for some feature map $\Phi$.

If $k(x, x')$ is much cheaper to compute than $\Phi(x)$ . . .
The Kernel Perceptron

initialize $f = 0$

repeat

Pick $(x_i, y_i)$ from data

if $y_i f(x_i) \leq 0$ then

$f(\cdot) \leftarrow f(\cdot) + y_i k(x_i, \cdot) + y_i$

until $y_i f(x_i) > 0$ for all $i$

• Nothing happens if classified correctly
• Weight vector is linear combination $w = \sum_{i \in I} y_i \phi(x_i)$
• Classifier is linear combination of inner products

$$f(x) = \sum_{i \in I} y_i \langle \phi(x_i), \phi(x) \rangle + b = \sum_{i \in I} y_i k(x_i, x) + b$$
Idea

We want to extend $k(x, x') = \langle x, x' \rangle^2$ to

$$k(x, x') = (\langle x, x' \rangle + c)^d \quad \text{where} \quad c > 0 \quad \text{and} \quad d \in \mathbb{N}.$$ 

Prove that such a kernel corresponds to a dot product.

Proof strategy

Simple and straightforward: compute the explicit sum given by the kernel, i.e.

$$k(x, x') = (\langle x, x' \rangle + c)^d = \sum_{i=0}^{m} \binom{d}{i} \langle x, x' \rangle^i c^{d-i}$$

Individual terms $(\langle x, x' \rangle)^i$ are dot products for some $\Phi_i(x)$. 
Kernel Conditions

Computability
We have to be able to compute \( k(x, x') \) efficiently (much cheaper than dot products themselves).

“Nice and Useful” Functions
The features themselves have to be useful for the learning problem at hand. Quite often this means smooth functions.

Symmetry
Obviously \( k(x, x') = k(x', x) \) due to the symmetry of the dot product \( \langle \Phi(x), \Phi(x') \rangle = \langle \Phi(x'), \Phi(x) \rangle \).

Dot Product in Feature Space
Is there always a \( \Phi \) such that \( k \) really is a dot product?
Mercer’s Theorem

The Theorem
For any symmetric function \( k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R} \) which is square integrable in \( \mathcal{X} \times \mathcal{X} \) and which satisfies
\[
\int_{\mathcal{X} \times \mathcal{X}} k(x, x') f(x) f(x') dx dx' \geq 0 \text{ for all } f \in L_2(\mathcal{X})
\]
there exist \( \phi_i : \mathcal{X} \rightarrow \mathbb{R} \) and numbers \( \lambda_i \geq 0 \) where
\[
k(x, x') = \sum_i \lambda_i \phi_i(x) \phi_i(x') \text{ for all } x, x' \in \mathcal{X}.
\]

Interpretation
Double integral is the continuous version of a vector-matrix-vector multiplication. For positive semidefinite matrices we have
\[
\sum_j \sum_i k(x_i, x_j) \alpha_i \alpha_j \geq 0
\]
**Properties**

**Distance in Feature Space**
Distance between points in feature space via

\[ d(x, x')^2 := ||\Phi(x) - \Phi(x')||^2 \]
\[ = \langle \Phi(x), \Phi(x) \rangle - 2 \langle \Phi(x), \Phi(x') \rangle + \langle \Phi(x'), \Phi(x') \rangle \]
\[ = k(x, x) + k(x', x') - 2k(x, x) \]

**Kernel Matrix**
To compare observations we compute dot products, so we study the matrix \( K \) given by

\[ K_{ij} = \langle \Phi(x_i), \Phi(x_j) \rangle = k(x_i, x_j) \]

where \( x_i \) are the training patterns.

**Similarity Measure**
The entries \( K_{ij} \) tell us the overlap between \( \Phi(x_i) \) and \( \Phi(x_j) \), so \( k(x_i, x_j) \) is a similarity measure.
**Properties**

\(K\) is Positive Semidefinite

Claim: \(\alpha^\top K \alpha \geq 0\) for all \(\alpha \in \mathbb{R}^m\) and all kernel matrices \(K \in \mathbb{R}^{m \times m}\). Proof:

\[
\sum_{i,j} \alpha_i \alpha_j K_{ij} = \sum_{i,j} \alpha_i \alpha_j \langle \Phi(x_i), \Phi(x_j) \rangle
\]

\[
= \left\langle \sum_{i} \alpha_i \Phi(x_i), \sum_{j} \alpha_j \Phi(x_j) \right\rangle = \left\| \sum_{i=1}^{m} \alpha_i \Phi(x_i) \right\|^2
\]

**Kernel Expansion**

If \(w\) is given by a linear combination of \(\Phi(x_i)\) we get

\[
\langle w, \Phi(x) \rangle = \left\langle \sum_{i=1}^{m} \alpha_i \Phi(x_i), \Phi(x) \right\rangle = \sum_{i=1}^{m} \alpha_i k(x_i, x).
\]
A Candidate for a Kernel

\[ k(x, x') = \begin{cases} 
1 & \text{if } \|x - x'\| \leq 1 \\
0 & \text{otherwise} 
\end{cases} \]

This is symmetric and gives us some information about the proximity of points, yet it is not a proper kernel . . .

Kernel Matrix

We use three points, \( x_1 = 1, x_2 = 2, x_3 = 3 \) and compute the resulting “kernel matrix” \( K \). This yields

\[
K = \begin{bmatrix}
1 & 1 & 0 \\
1 & 1 & 1 \\
0 & 1 & 1 
\end{bmatrix}
\]

and eigenvalues \((\sqrt{2}-1)^{-1}, 1\) and \((1-\sqrt{2})\).

as eigensystem. Hence \( k \) is not a kernel.
Examples

Examples of kernels $k(x, x')$

- Linear: $\langle x, x' \rangle$
- Laplacian RBF: $\exp (-\lambda \| x - x' \|)$
- Gaussian RBF: $\exp (-\lambda \| x - x' \|^2)$
- Polynomial: $\left( \langle x, x' \rangle + c \right)^d$, $c \geq 0$, $d \in \mathbb{N}$
- B-Spline: $B_{2n+1}(x - x')$
- Cond. Expectation: $E_c[p(x|c)p(x'|c)]$

Simple trick for checking Mercer’s condition
Compute the Fourier transform of the kernel and check that it is nonnegative.
Laplacian Kernel

\[ k(x, y) \text{ for } y = 1 \]
Gaussian Kernel

$k(x,y)$ for $x=1$
$B_3$ Spline Kernel
Summary

- Perceptron
  - Hebbian learning & biology
  - Algorithm
  - Convergence analysis
- Features and preprocessing
  - Nonlinear separation
  - Perceptron in feature space
- Kernels
  - Kernel trick
  - Properties
  - Examples