Machine Learning
A Geometric Approach

CUNY Graduate Center, Spring 2013

Lectures 2-4  Online Learning:
Perceptron and its Extensions

(including voted/avg perc, (aggressive) MIRA, multiclass, and feature preprocessing)

Professor Liang Huang
huang@cs.qc.cuny.edu

Alex Smola (CMU) slides
Liang Huang (CUNY) slides

http://acl.cs.qc.edu/~lhuang/teaching/machine-learning
• Perceptron
  • Classification in Augmented Space
  • Perceptron Algorithm
  • Convergence Proof
• Extensions of Perceptron
  • Voted/Averaged, MIRA, passive-aggressive, p-aggressive MIRA
  • Multiclass Perceptron
• Features and preprocessing
  • Nonlinear separation
  • Perceptron in feature space
• Kernels
  • Kernel trick
  • Kernelized Perceptron in Dual (Kai)
  • Properties
Perceptron

Frank Rosenblatt
early theories of the brain
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 (output cable)**
  May be up to 1m long and will transport the activation signal to neurons at different locations
Neurons

\[ f(x) = \sum_i w_i x_i = \langle w, x \rangle \]
Frank Rosenblatt’s Perceptron
Multilayer Perceptron (Neural Net)
Perceptron w/ bias

- Weighted linear combination
- Nonlinear decision function
- Linear offset (bias)

- Linear separating hyperplanes (spam/ham, novel/typical, click/no click)
- Learning: $w$ and $b$

$$f(x) = \sigma \left( \langle w, x \rangle + b \right)$$
Perceptron w/o bias

- Weighted linear combination
- Nonlinear decision function
- No Linear offset (bias): hyperplane through the origin
- Linear separation (spam/ham, n)
- Learning: $w$

$$f(x) = \sigma \left( \langle w, x \rangle + b \right)$$
Augmented Space

can’t separate in 1D from the origin

can separate in 2D from the origin

can separate in 3D from the origin

can’t separate in 2D from the origin
The Perceptron w/o bias

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 \( f(x) = \sum_{i \in I} y_i \langle x_i, x \rangle + b \)
The Perceptron w/ bias

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

repeat
  if \( y_i [\langle w, x_i \rangle + b] \leq 0 \) 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** \( f(x) = \sum_{i \in I} y_i \langle x_i, x \rangle + b \)
Demo

\[ (\text{bias}=0) \]
Demo
Demo
Convergence Theorem

• If there exists some oracle unit vector $u : \|u\| = 1$
  
  \[ y_i(u \cdot x_i) \geq \delta \text{ for all } i \]

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

  \[ R^2 / \delta^2 \text{ where } R = \max_i \|x_i\| \]

• Dimensionality independent
• Order independent (i.e. also worst case)
• Scales with ‘difficulty’ of problem
Geometry of the Proof

• part 1: progress (alignment) on oracle projection

assume $w_i$ is the weight vector before the $i$th update (on $\langle x_i, y_i \rangle$) and assume initial $w_0 = 0$

$$w_{i+1} = w_i + y_i x_i$$

$$u \cdot w_{i+1} = u \cdot w_i + y_i (u \cdot x_i)$$

$$u \cdot w_{i+1} \geq u \cdot w_i + \delta$$

$$u \cdot w_{i+1} \geq i \delta$$

projection on $u$ increases!

(more agreement w/ oracle)

$$\|w_{i+1}\| = \|u\| \|w_{i+1}\| \geq u \cdot w_{i+1} \geq i \delta$$

y_i (u \cdot x_i) \geq \delta \text{ for all}$$
Geometry of the Proof

- **part 2**: bound the norm of the weight vector

\[
\begin{align*}
    w_{i+1} &= w_i + y_i x_i \\
    \|w_{i+1}\|^2 &= \|w_i + y_i x_i\|^2 \\
    &= \|w_i\|^2 + \|x_i\|^2 + 2y_i (w_i x_i) \\
    &\leq \|w_i\|^2 + R^2 \\
    &\leq i R^2 \quad \text{(radius)}
\end{align*}
\]

Combine with part 1

\[
\begin{align*}
    \|w_{i+1}\| &= \|u\| \|w_{i+1}\| \geq u \cdot w_{i+1} \geq i \delta \\
    i &\leq R^2 / \delta^2
\end{align*}
\]
Convergence Bound $R^2/\delta^2$

- is independent of:
  - dimensionality
  - number of examples
  - starting weight vector
  - order of examples
  - constant learning rate
- and is dependent of:
  - separation difficulty
  - feature scale

- but test accuracy is dependent of:
  - order of examples (shuffling helps)
  - variable learning rate (1/total#error helps)

- can you still prove convergence?
Hardness
margin vs. size

hard

easy
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**
• XOR - not linearly separable
• Nonlinear separation is trivial
• Caveat from “Perceptrons” (Minsky & Papert, 1969)
  Finding the minimum error linear separator is NP hard (this killed Neural Networks in the 70s).
Brief History of Perceptron

1959
Rosenblatt invention

1962
Novikoff proof

1969*
Minsky/Papert book killed it

1969*
Freund/Schapire voted/avg: revived

1997
Cortes/Vapnik SVM

1999
Freund/Schapire voted/avg: revived

2002
Collins structured

2003
Crammer/Singer MIRA

2005*
McDonald/Crammer/Pereira structured MIRA

2005*
McDonald/Crammer/Pereira structured MIRA

2006
Singer group aggressive

2007–2010*
Singer group Pegasos

2007–2010*
Singer group Pegasos

AT&T Research

ex-AT&T and students

*mentioned in lectures but optional (others papers all covered in detail)

DEAD

online

minibatch

batch

+max margin
+kernels
+soft-margin

conservative updates

inseparable case

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch

online approx.

max margin

subgradient descent

minibatch
Extensions of Perceptron

- Problems with Perceptron
  - doesn’t converge with inseparable data
  - update might often be too “bold”
  - doesn’t optimize margin
  - is sensitive to the order of examples

- Ways to alleviate these problems
  - voted perceptron and average perceptron
  - MIRA (margin-infused relaxation algorithm)
  - passive-aggressive
Voted/Avged Perceptron

- motivation: updates on later examples taking over!
- voted perceptron (Freund and Schapire, 1999)
  - record the weight vector after each example
  - (not just after each update)
  - and vote on a new example
  - shown to have better generalization power
- averaged perceptron (from the same paper)
  - an approximation of voted perceptron
  - just use the average of all weight vectors
  - can be implemented efficiently
Voted/Avged Perceptron

d = 1 (low dim - less separable)
Voted/Avged Perceptron

$d = 6$ (high dim - more separable)
• perceptron often makes too bold updates
• but hard to tune learning rate
• the smallest update to correct the mistake?

\[ w_{i+1} = w_i + \frac{y_i - w_i \cdot x_i}{\|x_i\|^2} x_i \]

easy to show:
\[ y_i (w_{i+1} \cdot x_i) = y_i (w_i + \frac{y_i - w_i \cdot x_i}{\|x_i\|^2} x_i) \cdot x_i = 1 \]
Aggressive MIRA (AMIRA)

- aggressive version of MIRA
- also update if correct but margin not big enough
- functional margin: \( y_i (w \cdot x_i) \)
- geometric margin: \( \frac{y_i (w \cdot x_i)}{\|w\|} \)

update if functional margin is \( <= p \) \((0<=p<1)\)
- update rule is same as MIRA
- called AMIRAp or \( p \)-aggressive MIRA. (MIRA: \( p=0 \))
- larger \( p \) leads to a larger geometric margin
- but slower convergence

what if we replace functional here by geometric?
Table 3. Error rates on MNIST dataset. Both ROMMA and Aggressive ROMMA use a scale of 1100. The numbers in parentheses denote the aggressive parameters for AMIRA.

<table>
<thead>
<tr>
<th>Epoch</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perceptron</td>
<td>2.98%</td>
<td>2.32%</td>
<td>1.94%</td>
<td>1.88%</td>
</tr>
<tr>
<td>Perceptron(avg.)</td>
<td>2.16%</td>
<td>1.85%</td>
<td>1.73%</td>
<td>1.69%</td>
</tr>
<tr>
<td>ROMMA</td>
<td>2.48%</td>
<td>1.96%</td>
<td>1.79%</td>
<td>1.77%</td>
</tr>
<tr>
<td>aggr-ROMMA</td>
<td><strong>2.14%</strong></td>
<td>1.82%</td>
<td>1.71%</td>
<td>1.67%</td>
</tr>
<tr>
<td>MIRA</td>
<td>2.56%</td>
<td>2.03%</td>
<td>1.74%</td>
<td>1.70%</td>
</tr>
<tr>
<td>bin AMIRA(0.1)</td>
<td>2.20%</td>
<td><strong>1.78%</strong></td>
<td>1.67%</td>
<td>1.64%</td>
</tr>
</tbody>
</table>
Demo

• perceptron vs. 0.2-aggressive vs. 0.9-aggressive
Demo

• perceptron vs. 0.2-aggressive vs. 0.9-aggressive
• why does this dataset so **slow** to converge?
  • perceptron: 22, p=0.2: 87, p=0.9: 2,518 epochs

answer: margin shrinks in augmented space!

small margin in 2D
big margin in 1D
Demo

- perceptron vs. 0.2-aggressive vs. 0.9-aggressive
- why does this dataset so fast to converge?
- perceptron: 3, p=0.2: 1, p=0.9: 5 epochs

answer: margin shrinks in augmented space!

ok margin in 2D

big margin in 1D
Multiclass Classification

- one weight vector ("prototype") for each class:
  \[ w = (w^{(1)}, w^{(2)}, \ldots, w^{(M)}), \]
- multiclass decision rule:
  \[ \hat{y} = \arg\max_{z \in 1\ldots M} w^{(z)} \cdot x \]
  (best agreement w/ prototype)

Q1: what about 2-class?
Q2: do we still need augmented space?
Multiclass Perceptron

- on an error, penalize the weight for the wrong class, and reward the weight for the true class
Convergence of Multiclass

update rule:
\[ w \leftarrow w + \Delta \Phi(x, y, z) \]

separability:
\[ \exists u, \text{ s.t. } \forall (x, y) \in D, z \neq y \]
\[ u \cdot \Delta \Phi(x, y, z) \geq \delta \]

where \( w^{(i)} \) is used to calculate the functional margin for training example with label \( i \);

for a given training example \( x \) and a label \( y \), we define feature map function \( \Phi \) as
\[ \Phi(x, y) = (0^{(1)}, \ldots, 0^{(y-1)}, x, 0^{(y+1)}, \ldots, 0^{(M)}) \]
such that \( w \cdot \Phi(x, y) = w^{(y)} \cdot x \).

We also define that, with a given training example \( x \), the difference between two feature vectors for labels \( y \) and \( z \) as \( \Delta \Phi \):
\[ \Delta \Phi(x, y, z) = \Phi(x, y) - \Phi(x, z) \].
Useful Engineering Tips:

- shuffling at each epoch helps a lot
- **variable** learning rate often helps (constant: useless)
  - 1/(total#updates) or 1/(total#examples) helps
- any requirement in order to converge?
  - how to prove convergence now?
- centering of each feature dim helps
  - why? => R smaller, margin bigger
- unit variance also helps (why?)
  - 0-mean, 1-var => each feature ≈ a unit Gaussian
Useful Engineering Tips:
feature bucketing (binning/quantization), categorical=>binary

- HW1 Adult income dataset: <=50K, or >50K?
  - age: older means more $$$?
    - bin: Young (0-25), Middle-aged (26-45), Senior (46-65) and Old (66+).
- educational level: 1 to 9 (i think higher is better)
- hours-per-week: more hours means more $$$?
  - bin: Part-time (0-25), Full-time (25-40), Over-time (40-60) and Too-much (60+).
- native-country: split into X binaries for X countries
- gender: binary; no need to split into two binaries!
- type-of-work or position: split into many binaries
- ...

...