Online Learning
Perceptron Algorithm

Machine Learning – CSE446
Carlos Guestrin
University of Washington

April 29, 2013

Challenge 1: Complexity of Computing Gradients in LR

features

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left\{ -\lambda w_i^{(t)} + \sum_{j=1}^{N} x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \overbrace{\mathbf{w}^{(t)})]}^{O(k)} \right\}$$

$O(Nk)$

if lots of data ... N is very large

→ very slow.

We talked about SGD instead

small change after each data point

Challenge 2: Data is streaming

- Assumption thus far: **Batch data**

  Have data, will machine learn

- But, e.g., in click prediction for ads is a streaming data task:
  - User enters query, and ad must be selected:
    - Observe $\mathbf{x}^j$, and must predict y^j

  Q → webpage → x^j → predict $\hat{y}$ — will the user click → which ads have high click prob.

  x^j: features (page, Q user, ad)

  - User either clicks or doesn't click on ad:
    - Label y^j is revealed afterwards
      - Google gets a reward if user clicks on ad

  $\mathbb{1}(\hat{y} \neq y^j)$

  reward is like 0/1 loss in classification

  - Weights must be updated for next time:

  $w^{(t+1)} \leftarrow w^{(t)} + \Delta$ ← what's Δ?

Online Learning Problem

- At each time step t:
  - Observe features of data point:
    - Note: many assumptions are possible, e.g., data is iid, data is adversarially chosen… details beyond scope of course

  $x^{(t)} \leftarrow$ (page, user, ad)

  $x^{(t)} = \begin{pmatrix} 1 \\ \text{page features} \\ \text{user features} \\ \text{ad features} \end{pmatrix}$

  - Make a prediction: $\hat{y}$ hopefully $\approx y^{(t)}$ ← unknown
    - Note: many models are possible, we focus on linear models
    - *For simplicity, use vector notation*

  $w_0 + \sum_i w_i^{(t)} x_i^{(t)} > 0$? ⇒ click

  $w^{(t)} \cdot x^{(t)} > 0$?

  $\sum_{i=0} w_i^{(t)} x_i^{(t)} > 0$?

  $x_0^{(t)} = 1 \quad \forall t$

  - Observe true label:
    - Note: other observation models are possible, e.g., we don't observe the label directly, but only a noisy version... Details beyond scope of course

  observe $y^{(t)}$ → clicked / not clicked

  - Update model:

  $w^{(t+1)} \leftarrow w^{(t)} + \Delta^{(t)}_{\text{something}}$ ← what's Δ?

The Perceptron Algorithm [Rosenblatt '58, '62]

- Classification setting: y in {-1,+1}
- Linear model
  - Prediction: $\hat{y} = \text{Sign}(w \cdot x)$
- Training: $w^{(0)} = 0$ or something smarter
  - Initialize weight vector:
  - At each time step:
    - Observe features: $x^{(t)} \leftarrow$ (page, user, ad)
    - Make prediction: $\hat{y} = \text{Sign}(w^{(t)} \cdot x^{(t)})$
    - Observe true class:
      $y^{(t)} \leftarrow$ true label
    - Update model:
      - If prediction is not equal to truth, if make a mistake!

if $\hat{y} = y^{(t)}$
$w^{(t+1)} \leftarrow w^{(t)}$
else
$w^{(t+1)} \leftarrow w^{(t)} + y^{(t)} x^{(t)}$

I made a mistake:
e.g. $y^{(t)} = +1$
$w^{(t)} \cdot x^{(t)} < 0$
but wanted > 0
what u may $w \cdot x^{(t)}$? u $x^{(t)}$!! :)
by adding $x^{(t)}$ to w
I increase $w^{(t+1)} \cdot x^{(t)}$ the most

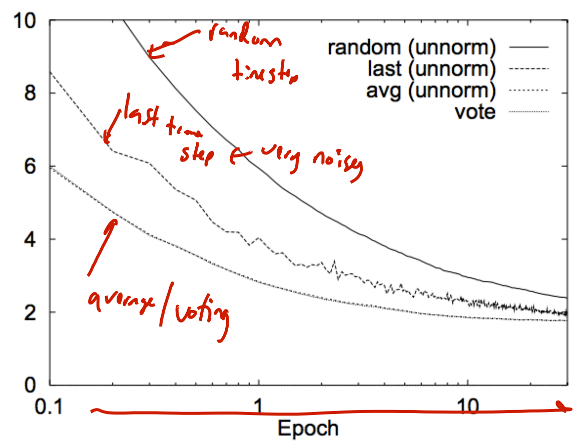
similarly when $y^{(t)} = -1$

Fundamental Practical Problem for All Online Learning Methods: **Which weight vector to report?**

- Suppose you run online learning method and want to sell your learned weight vector… Which one do you sell???
- Last one? $w^{(T)}$? no… very noisy, influenced by last mistake
- Random time step?
- average!! :) $\hat{w} = \frac{1}{T} \sum_{t=1}^{T} w^{(t)}$ (easy to keep track)
- Voting & more advanced methods: how long has this param been good

Choice can make a huge difference!!

error



t

[Freund & Schapire '99]

Mistake Bounds

why does it work??

- Algorithm "pays" every time it makes a mistake:

Loss function for online setting : number of mistakes up to time T

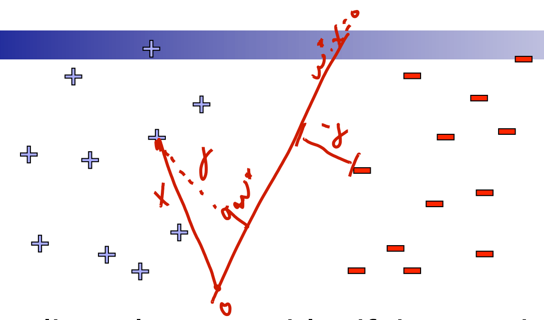
⇒ Google pays for its mistake

in its life time

- How many mistakes is it going to make?

Mistake bound

Linear Separability: More formally, Using Margin



- Data linearly separable, if there exists
  - a vector $\exists\, w^*, \ \|w^*\| = 1$
  - a margin $\gamma > 0$
- Such that all points are γ far or more from $w^* \cdot x = 0$

$\forall t$ if $y^{(t)} = +1 \quad w^* \cdot x^{(t)} > \gamma$

$y^{(t)} = -1 \quad w^* \cdot x^{(t)} < -\gamma$

$\underbrace{y^{(t)}\, w^* \cdot x^{(t)} > \gamma}_{\text{linearly separable, margin } \gamma}$

Perceptron Analysis: Linearly Separable Case

- Theorem [Block, Novikoff]:
  - Given a sequence of labeled examples: $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}) \ldots\ldots (x^{(T)}, y^{(T)})$

    examples need not be i.i.d. or random..
  - Each feature vector has bounded norm:

    $\forall t \ \|x^{(t)}\| \le R$ $\qquad$ w^* is unknown!
  - If dataset is linearly separable:

    $\exists\, w^*, \ \|w^*\| = 1 \quad \forall t \quad y^{(t)}\, w^* \cdot x^{(t)} \geq \gamma, \ \text{for } \gamma > 0$
- Then the number of mistakes made by the online perceptron on ~~this~~ any sequence is bounded by

$$\left(\frac{R}{\gamma}\right)^2$$

wow!! ¡

constant, doesn't depend on T or dimensionality of X !! :)

$a \cdot b \le ||a|| \cdot ||b||$

Perceptron Proof for Linearly Separable case

γ

- Every time we make a mistake, we get gamma closer to w*:
  - Mistake at time t: $w^{(t+1)} = w^{(t)} + y^{(t)} x^{(t)}$
  - Taking dot product with w*: $w^* \cdot w^{(t+1)} = w^* \cdot (w^{(t)} + y^{(t)} x^{(t)})$
    $= w^* \cdot w^{(t)} + y^{(t)} w^* x^{(t)}$
    mistake $\ge w^* \cdot w^{(t)} + \gamma$; $\ge \gamma$ linearly separable
  - Thus after m mistakes:
    by induction: $w^* \cdot w^{(t+1)} \ge m\gamma$
- Similarly, norm of $w^{(t+1)}$ doesn't grow too fast:
  - $$||\mathbf{w}^{(t+1)}||^2 = ||\mathbf{w}^{(t)}||^2 + \underbrace{2y^{(t)}(\mathbf{w}^{(t)} \cdot \mathbf{x}^{(t)})}_{<0 \text{ mistake}} + ||\mathbf{x}^{(t)}||^2 \le R^2$$
  - Thus, after m mistakes:
    $||w^{(t+1)}||^2 \le m R^2$
- Putting all together:

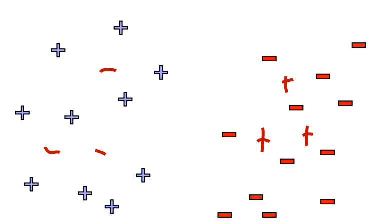
$$m\gamma \le w^* \cdot w^{(t+1)} \le \underbrace{||w^*||}_{=1} \cdot ||w^{(t+1)}|| \le \sqrt{m} R$$

wow!! :)

$$\Rightarrow \quad m\gamma \le \sqrt{m}\, R \quad \Rightarrow \quad m \le \left(\frac{R}{\gamma}\right)^2$$

Beyond Linearly Separable Case

- Perceptron algorithm is super cool!
  - No assumption about data distribution!
    - Could be generated by an oblivious adversary, no need to be iid
  - Makes a fixed number of mistakes, and it's done for ever! ← $\left(\frac{R}{\gamma}\right)^2$
    - Even if you see infinite data



- However, real world not linearly separable
  - Can't expect never to make mistakes again
  - Analysis extends to non-linearly separable case
  - Very similar bound, see Freund & Schapire
  - Converges, but ultimately may not give good accuracy (make many many many mistakes)

we need features that make data as linearly separable as possible

What you need to know

- Notion of online learning
- Perceptron algorithm
- Mistake bounds and proof
- In online learning, report averaged weights at the end