Boosting Approach to ML
Perceptron, Margins, Kernels

Maria-Florina Balcan
03/18/2015
Recap from last time: Boosting

- General method for improving the accuracy of any given learning algorithm.

- Works by creating a series of challenge datasets s.t. even modest performance on these can be used to produce an overall high-accuracy predictor.

- **Adaboost** one of the top 10 ML algorithms.
  - Works amazingly well in practice.
  - Backed up by solid foundations.
Adaboost (Adaptive Boosting)

Input: $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$; $x_i \in X$, $y_i \in Y = \{-1, 1\}$

weak learning algo $A$ (e.g., Naïve Bayes, decision stumps)

- For $t=1,2, \ldots, T$
  - Construct $D_t$ on $\{x_1, \ldots, x_m\}$
  - Run $A$ on $D_t$ producing $h_t: X \rightarrow \{-1, 1\}$

Output $H_{\text{final}}(x) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(x))$

- $D_1$ uniform on $\{x_1, \ldots, x_m\}$ [i.e., $D_1(i) = \frac{1}{m}$]
- Given $D_t$ and $h_t$ set

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} e^{-\alpha_t}$$ if $y_i = h_t(x_i)$

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} e^{\alpha_t}$$ if $y_i \neq h_t(x_i)$

$$\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right) > 0$$

$D_{t+1}$ puts half of weight on examples $x_i$ where $h_t$ is incorrect & half on examples where $h_t$ is correct
Nice Features of Adaboost

• **Very general**: a meta-procedure, it can use **any** weak learning algorithm!!! (e.g., Naïve Bayes, decision stumps)

• **Very fast** (single pass through data each round) & **simple** to code, no parameters to tune.

• **Grounded in rich theory.**
Analyzing Training Error

**Theorem** \( \epsilon_t = 1/2 - \gamma_t \) (error of \( h_t \) over \( D_t \))

\[
err_S(H_{final}) \leq \exp \left[ -2 \sum_t \gamma_t^2 \right]
\]

So, if \( \forall t, \gamma_t \geq \gamma > 0 \), then \( err_S(H_{final}) \leq \exp[ -2 \gamma^2 T] \)

The training error drops exponentially in \( T \)!!

To get \( err_S(H_{final}) \leq \epsilon \), need only \( T = O \left( \frac{1}{\gamma^2 \log \left( \frac{1}{\epsilon} \right)} \right) \) rounds

**Adaboost is adaptive**

- Does not need to know \( \gamma \) or \( T \) a priori
- Can exploit \( \gamma_t \gg \gamma \)
Generalization Guarantees

Theorem \[ \text{err}_S(H_{\text{final}}) \leq \exp \left[ -2 \sum_t \gamma_t^2 \right] \text{ where } \epsilon_t = 1/2 - \gamma_t \]

How about generalization guarantees?

Original analysis [Freund&Schapire'97]

- \( H \) space of weak hypotheses; \( d = \text{VCdim}(H) \)
- \( H_{\text{final}} \) is a weighted vote, so the hypothesis class is:
  \[ G = \{ \text{all fns of the form } \text{sign}(\sum_{t=1}^T \alpha_t h_t(x)) \} \]

Theorem [Freund&Schapire'97]

\( \forall g \in G, \text{err}(g) \leq \text{err}_S(g) + \tilde{O} \left( \sqrt{\frac{Td}{m}} \right) \) \( T = \# \text{ of rounds} \)

Key reason: \( \text{VCdim}(G) = \tilde{O}(dT) \) plus typical VC bounds.
Generalization Guarantees

**Theorem** [Freund&Schapire’97]

\[ \forall g \in G, \text{err}(g) \leq \text{err}_S(g) + \tilde{O}\left(\sqrt{\frac{Td}{m}}\right) \]

where \(d=\text{VCdim}(H)\)
Generalization Guarantees

• Experiments showed that the test error of the generated classifier usually does not increase as its size becomes very large.

• Experiments showed that continuing to add new weak learners after correct classification of the training set had been achieved could further improve test set performance!!!
Generalization Guarantees

• Experiments showed that the test error of the generated classifier usually *does not increase* as its size becomes very large.

• Experiments showed that continuing to add weak learners after correct classification of the training set had been achieved could further *improve* test set performance!!

• These results seem to contradict FS’97 bound and Occam’s razor (un to achieve good test error the classifier should be as simple as

\[ \forall g \in G, \text{err}(g) \leq \text{err}_S(g) + O\left(\sqrt{\frac{T}{m}}\right) \]
How can we explain the experiments?

R. Schapire, Y. Freund, P. Bartlett, W. S. Lee. present in “Boosting the margin: A new explanation for the effectiveness of voting methods” a nice theoretical explanation.

Key Idea:

Training error does not tell the whole story.

We need also to consider the classification confidence!!
Boosting didn’t seem to overfit...(!)

...because it turned out to be increasing the margin of the classifier

Error Curve, Margin Distr. Graph - Plots from [SFBL98]
Classification Margin

- **H space of weak hypotheses.** The convex hull of H:

\[
\text{co}(H) = \{f = \sum_{t=1}^{T} \alpha_t h_t, \alpha_t \geq 0, \sum_{t=1}^{T} \alpha_t = 1, h_t \in H\}
\]

- Let \( f \in \text{co}(H), f = \sum_{t=1}^{T} \alpha_t h_t, \alpha_t \geq 0, \sum_{t=1}^{T} \alpha_t = 1. \)

The majority vote rule \( H_f \) given by \( f \) (given by \( H_f = \text{sign}(f(x)) \)) predicts wrongly on example \((x, y)\) iff \( yf(x) \leq 0.\)

**Definition:** margin of \( H_f \) (or of \( f \)) on example \((x, y)\) to be \( yf(x).\)

\[
yf(x) = y \sum_{t=1}^{T} [\alpha_t h_t(x)] = \sum_{t=1}^{T} [y \alpha_t h_t(x)] = \sum_{t: y = h_t(x)} \alpha_t - \sum_{t: y \neq h_t(x)} \alpha_t
\]

The margin is positive iff \( y = H_f(x).\)

See \(|yf(x)| = |f(x)|\) as the strength or the confidence of the vote.

\[
\begin{array}{ccc}
-1 & \text{Low confidence} & 1 \\
\text{High confidence, incorrect} & \text{High confidence, correct}
\end{array}
\]
Boosting and Margins

**Theorem:** $\text{VCdim}(H) = d$, then with prob. $\geq 1 - \delta$, $\forall f \in \text{co}(H)$, $\forall \theta > 0$,

$$
\Pr_D[yf(x) \leq 0] \leq \Pr_S[yf(x) \leq \theta] + O\left(\frac{1}{\sqrt{m}} \sqrt{\frac{d \ln^2 m}{\theta^2 d} + \ln \frac{1}{\delta}}\right)
$$

**Note:** bound does not depend on $T$ (the # of rounds of boosting), depends only on the complex. of the weak hyp space and the margin!
Boosting and Margins

Theorem: $\text{VCdim}(H) = d$, then with prob. $\geq 1 - \delta$, $\forall f \in \text{co}(H)$, $\forall \theta > 0$,

$$\Pr_D[yf(x) \leq 0] \leq \Pr_S[yf(x) \leq \theta] + O\left(\frac{1}{\sqrt{m}} \sqrt{\frac{d \ln^2 m}{\theta^2} + \ln \frac{1}{\delta}}\right)$$

- If all training examples have large margins, then we can approximate the final classifier by a much smaller classifier.

- Can use this to prove that better margin $\Rightarrow$ smaller test error, regardless of the number of weak classifiers.

- Can also prove that boosting tends to increase the margin of training examples by concentrating on those of smallest margin.

- Although final classifier is getting larger, margins are likely to be increasing, so the final classifier is actually getting closer to a simpler classifier, driving down test error.
Boosting and Margins

Theorem: $\text{VCdim}(H) = d$, then with prob. $\geq 1 - \delta$, $\forall f \in \text{co}(H)$, $\forall \theta > 0$,

$$\Pr_D[yf(x) \leq 0] \leq \Pr_S[yf(x) \leq \theta] + O \left( \frac{1}{\sqrt{m}} \sqrt{\frac{d \ln^2 m}{\theta^2 d}} + \ln \frac{1}{\delta} \right)$$

Note: bound does not depend on $T$ (the # of rounds of boosting), depends only on the complexity of the weak hyp space and the margin!
Boosting, Adaboost Summary

- Shift in mindset: goal is now just to find classifiers a bit better than random guessing.
- Backed up by solid foundations.
- Adaboost work and its variations well in practice with many kinds of data (one of the top 10 ML algos).
- More about classic applications in Recitation.

- Relevant for big data age: quickly focuses on “core difficulties”, so well-suited to distributed settings, where data must be communicated efficiently [Balcan-Blum-Fine-Mansour COLT'12].
Interestingly, the usefulness of margin recognized in Machine Learning since late 50's.

Perceptron [Rosenblatt'57] analyzed via geometric (aka $L_2$, $L_2$) margin.

Original guarantee in the online learning scenario.
The Perceptron Algorithm

- Online Learning Model
- Margin Analysis
- Kernels
The Online Learning Model

• Example arrive **sequentially**.
• We need to make a prediction. Afterwards observe the outcome.

For $i = 1, 2, \ldots, $:

\[ \text{Example } x_i \rightarrow \text{Prediction } h(x_i) \rightarrow \text{Observe } c^*(x_i) \]

**Mistake bound model**

• Analysis wise, make **no distributional assumptions**.
• **Goal**: **Minimize** the number of mistakes.
The Online Learning Model. Motivation

- Email classification (distribution of both spam and regular mail changes over time, but the target function stays fixed - last year's spam still looks like spam).

- Recommendation systems. Recommending movies, etc.

- Predicting whether a user will be interested in a new news article or not.

- Add placement in a new market.
Linear Separators

• Instance space $X = \mathbb{R}^d$

• Hypothesis class of linear decision surfaces in $\mathbb{R}^d$.

• $h(x) = w \cdot x + w_0$, if $h(x) \geq 0$, then label $x$ as $+$, otherwise label it as $-$

Claim: WLOG $w_0 = 0$.

Proof: Can simulate a non-zero threshold with a dummy input feature $x_0$ that is always set up to 1.

• $x = (x_1, \ldots, x_d) \rightarrow \tilde{x} = (x_1, \ldots, x_d, 1)$

• $w \cdot x + w_0 \geq 0$ iff $(w_1, \ldots, w_d, w_0) \cdot \tilde{x} \geq 0$

where $w = (w_1, \ldots, w_d)$
Linear Separators: Perceptron Algorithm

- Set $t=1$, start with the all zero vector $w_1$.
- Given example $x$, predict positive iff $w_t \cdot x \geq 0$
- On a mistake, update as follows:
  - Mistake on positive, then update $w_{t+1} \leftarrow w_t + x$
  - Mistake on negative, then update $w_{t+1} \leftarrow w_t - x$

**Note:** $w_t$ is weighted sum of incorrectly classified examples

$$w_t = a_{i_1}x_{i_1} + \cdots + a_{i_k}x_{i_k}$$

$$w_t \cdot x = a_{i_1}x_{i_1} \cdot x + \cdots + a_{i_k}x_{i_k} \cdot x$$

Important when we talk about kernels.
Perceptron Algorithm: Example

Example: \((-1,2) – \times\)
- \((1,0) + \checkmark\)
- \((1,1) + \times\)
- \((-1,0) – \checkmark\)
- \((-1, -2) – \times\)
- \((1, -1) + \checkmark\)

**Algorithm:**
- Set \(t=1\), start with all-zeroes weight vector \(w_1\).
- Given example \(x\), predict positive iff \(w_t \cdot x \geq 0\).
  - On a mistake, update as follows:
    - Mistake on positive, update \(w_{t+1} \leftarrow w_t + x\)
    - Mistake on negative, update \(w_{t+1} \leftarrow w_t - x\)

\(w_1 = (0,0)\)
\(w_2 = w_1 - (-1,2) = (1, -2)\)
\(w_3 = w_2 + (1,1) = (2, -1)\)
\(w_4 = w_3 - (-1,-2) = (3,1)\)
**Geometric Margin**

**Definition:** The **margin** of example $x$ w.r.t. a linear sep. $w$ is the distance from $x$ to the plane $w \cdot x = 0$ (or the negative if on wrong side)

Margin of positive example $x_1$

Margin of negative example $x_2$
**Geometric Margin**

Definition: The margin of example $x$ w.r.t. a linear sep. $w$ is the distance from $x$ to the plane $w \cdot x = 0$ (or the negative if on wrong side).

Definition: The margin $\gamma_w$ of a set of examples $S$ wrt a linear separator $w$ is the smallest margin over points $x \in S$. 
**Geometric Margin**

**Definition:** The margin of example $x$ w.r.t. a linear separator $w$ is the distance from $x$ to the plane $w \cdot x = 0$ (or the negative if on wrong side).

**Definition:** The margin $\gamma_w$ of a set of examples $S$ wrt a linear separator $w$ is the smallest margin over points $x \in S$.

**Definition:** The margin $\gamma$ of a set of examples $S$ is the maximum $\gamma_w$ over all linear separators $w$. 
Theorem: If data has margin $\gamma$ and all points inside a ball of radius $R$, then Perceptron makes $\leq \left(\frac{R}{\gamma}\right)^2$ mistakes.

(Normalized margin: multiplying all points by 100, or dividing all points by 100, doesn't change the number of mistakes; algo is invariant to scaling.)
**Theorem:** If data has margin $\gamma$ and all points inside a ball of radius $R$, then Perceptron makes $\leq \left(\frac{R}{\gamma}\right)^2$ mistakes.

**Update rule:**
- Mistake on positive: $w_{t+1} \leftarrow w_t + x$
- Mistake on negative: $w_{t+1} \leftarrow w_t - x$

**Proof:**

**Idea:** analyze $w_t \cdot w^*$ and $\|w_t\|$, where $w^*$ is the max-margin sep, $\|w^*\| = 1$.

**Claim 1:** $w_{t+1} \cdot w^* \geq w_t \cdot w^* + \gamma$.  \hspace{1cm} (because $l(x)x \cdot w^* \geq \gamma$)

**Claim 2:** $\|w_{t+1}\|^2 \leq \|w_t\|^2 + R^2$. \hspace{1cm} (by Pythagorean Theorem)

After $M$ mistakes:

$w_{M+1} \cdot w^* \geq \gamma M$ \hspace{1cm} (by Claim 1)

$\|w_{M+1}\| \leq R\sqrt{M}$ \hspace{1cm} (by Claim 2)

$w_{M+1} \cdot w^* \leq \|w_{M+1}\|$ \hspace{1cm} (since $w^*$ is unit length)

So, $\gamma M \leq R\sqrt{M}$, so $M \leq \left(\frac{R}{\gamma}\right)^2$. 

---

**Diagram**

- $w_t$ and $w_{t+1}$ are vectors.
- $x$ is a data point.
- $w^*$ is the max-margin vector.
- $\|w_t\|$ and $\|w_{t+1}\|$ are the lengths of the vectors.
- $\gamma$ represents the margin.
- $R$ represents the radius of the ball.
Perceptron Extensions

- Can use it to find a consistent separator (by cycling through the data).
- One can convert the mistake bound guarantee into a distributional guarantee too (for the case where the $x_i$s come from a fixed distribution).
- Can be adapted to the case where there is no perfect separator as long as the so called hinge loss (i.e., the total distance needed to move the points to classify them correctly large margin) is small.
- Can be kernelized to handle non-linear decision boundaries!
Perceptron Discussion

• Simple online algorithm for learning linear separators with a nice guarantee that depends only on the geometric (aka $L_2, L_2$) margin.

• It can be kernelized to handle non-linear decision boundaries --- see next class!

• Simple, but very useful in applications like Branch prediction; it also has interesting extensions to structured prediction.