Classification

Computational Linguistics: Jordan Boyd-Graber
University of Maryland
PERCEPTRON
Motivation

- **On-line learning:**
  - update parameters with each example
  - no distributional assumption.
  - worst-case analysis (adversarial).
  - mixed training and test.
  - Performance measure: mistake model, regret.
General Online Setting

- For $t = 1$ to $T$:
  - Get instance $x_t \in X$
  - Predict $\hat{y}_t \in Y$
  - Get true label $y_t \in Y$
  - Incur loss $L(\hat{y}_t, y_t)$

- Classification: $Y = \{0, 1\}$, $L(y, y') = |y' - y|$

- Regression: $Y \subset \mathbb{R}$, $L(y, y') = (y' - y)^2$
General Online Setting

- For $t = 1$ to $T$:
  - Get instance $x_t \in X$
  - Predict $\hat{y}_t \in Y$
  - Get true label $y_t \in Y$
  - Incur loss $L(\hat{y}_t, y_t)$

- **Classification**: $Y = \{0, 1\}$, $L(y, y') = |y' - y|$
- **Regression**: $Y \subset \mathbb{R}$, $L(y, y') = (y' - y)^2$

- **Objective**: Minimize total loss $\sum_t L(\hat{y}_t, y_t)$
Perceptron Algorithm

- Online algorithm for classification
- Very similar to logistic regression (but 0/1 loss)
- But what can we prove?
Perceptron Algorithm

\[ \mathbf{w}_1 \leftarrow \mathbf{0}; \]

\textbf{for} \ t \leftarrow 1 \ldots T \textbf{ do}

\hspace{1em} \text{Receive } x_t; \\
\hspace{1em} \hat{y}_t \leftarrow \text{sgn}(\mathbf{w}_t \cdot \mathbf{x}_t); \\
\hspace{1em} \text{Receive } y_t; \\
\hspace{1em} \text{if } \hat{y}_t \neq y_t \text{ then} \\
\hspace{2em} \mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_t \mathbf{x}_t; \\
\hspace{1em} \text{else} \\
\hspace{2em} \mathbf{w}_{t+1} \leftarrow \mathbf{w}_t; \\
\textbf{return } \mathbf{w}_{T+1};

\textbf{Algorithm 1:} Perceptron Algorithm (Rosenblatt, 1958)
Objective Function

- Optimizes

\[ \frac{1}{T} \sum_t \max(0, -y_t (\vec{w} \cdot x_t)) \]  

- Convex but not differentiable
Margin and Errors

- If there's a good margin $\rho$, you’ll converge quickly

$w \cdot x = 0$
Margin and Errors

- If there’s a good margin $\rho$, you’ll converge quickly.
- Whenever you see an error, you move the classifier to get it right.
- Convergence only possible if data are separable.
How many errors does Perceptron make?

- If your data are in a $R$ ball and there is a margin

$$\rho \leq \frac{y_t(\vec{v} \cdot \vec{x}_t)}{||v||}$$

(2)

for some $\vec{v}$, then the number of mistakes is bounded by $R^2/\rho^2$

- The places where you make an error are support vectors
- Convergence can be slow for small margins
Why study Perceptron?

- Simple algorithm
- Bound independent of dimension and tight
- Foundation of deep learning
- Proof techniques helped usher in SVMs
- Generalizes to structured prediction