CSE 151A - Discussion 06

Quick Review

Variations on Gradient Descent
Goal : Take advantage of the decomposability of our objective functions

- Full Batch $\rightarrow \nabla R(\vec{w}) = \sum_{i=1}^{n} \nabla \ell(\vec{w}; \vec{x}^{(i)}, y_i)$

  Update $\vec{w}$ after processing all n points $\rightarrow O(nd)$ runtime for a single step

- Mini Batch $\rightarrow \nabla R(\vec{w}) \approx \sum_{i \in B} \nabla \ell(\vec{w}; \vec{x}^{(i)}, y_i)$, where B is a set of $n' \leq n$ points

  Update $\vec{w}$ after processing n' points in minibatch $\rightarrow O(n'd)$ runtime for a single step

- Stochastic $\rightarrow \nabla R(\vec{w}) \approx \nabla \ell(\vec{w}; \vec{x}^{(i)}, y_i)$

  Update $\vec{w}$ after processing just one point, $\vec{x}^{(i)} \rightarrow O(d)$ runtime for a single step

Perceptrons
Goal : Learn a linear decision boundary to make classifications
Key property : Converges only when the data is linearly separable
Perceptron Algorithm : loop over misclassified points $i \in M$ and perform the following update:

$$\vec{w}^{(t)} = \vec{w}^{(t-1)} - \alpha \begin{cases} Aug(\vec{x}^{(i)}), & \vec{w}^{(t-1)} \cdot Aug(\vec{x}^{(i)})) \geq 0 \\ -Aug(\vec{x}^{(i)}), & \vec{w}^{(t-1)} \cdot Aug(\vec{x}^{(i)})) < 0 \end{cases}$$

Prediction Rule :

$$\text{prediction} = \begin{cases} 1 & \text{if } \vec{w} \cdot Aug(\vec{x}) \geq 0 \\ -1 & \text{if } \vec{w} \cdot Aug(\vec{x}) < 0 \end{cases}$$

Support Vector Machines
Goal : Maximize the margin of a linear decision boundary
Support Vector : A training point $\vec{x}^{(i)}$ such that $y_i \vec{w} \cdot Aug(\vec{x}^{(i)}) = 1$

- Hard Margin
  Assumption : Data is linearly separable
  Goal : Minimize $||\vec{w}||^2$ subject to $y_i \vec{w} \cdot Aug(\vec{x}^{(i)}) \geq 1$ for all i
  $\vec{w} = \sum_{i \in S} y_i \alpha_i Aug(\vec{x}^{(i)})$ where S is the set of support vectors

- Soft Margin
  Assumption : Data may not be linearly separable
  Goal : Minimize $||w||^2 + C \sum_{i=1}^{n} \xi_i$ subject to $y_i \vec{w} \cdot Aug(\vec{x}^{(i)}) \geq 1 - \xi_i$ for all i (and $\xi_i \geq 0, C \geq 0$)
  $\xi_i = 1 - y_i \vec{w} \cdot Aug(\vec{x}^{(i)})$ for a misclassified $\vec{x}^{(i)}$
  C : slack parameter (as C increases, we allow less slack, harden the margin, avoid misclassifications)

Problem 1.

Suppose we want to learn a perceptron over data points in two dimensions (i.e. each $\vec{x} = (x_1, x_2)^T$).
Write an equation for the decision boundary in slope-intercept form.
Hint : Your result should resemble the form $x_2 = A \cdot x_1 + B$ where A and B are in terms of $\vec{w} = (w_0, w_1, w_2)^T$.

Problem 2.

While running the perceptron algorithm, suppose that at the start of some arbitrary step t the value of $\vec{w}$ is given as $\vec{w}^{(t)} = (w_0^{(t)}, w_1^{(t)}, w_2^{(t)})^T$, and that the value of the input is given as $\vec{x} = (x_0, x_1, x_2)^T = (1, 0, -1)^T$. Determine, for each component of $\vec{w}^{(t+1)} = (w_0^{(t+1)}, w_1^{(t+1)}, w_2^{(t+1)})^T$, if the value is $>$, $<$, or $=$ the corresponding value of $\vec{w}^{(t)}$ after applying the update rule in each of the following circumstances:

a) False Positive (we predict $y = 1$ but the correct label is $y = -1$).

$w_0^{(t+1)} \ \square \ w_0^{(t)}$

$w_1^{(t+1)} \ \square \ w_1^{(t)}$

$w_2^{(t+1)} \ \square \ w_2^{(t)}$

b) Correct Positive Prediction (we predict $y = 1$ and the correct label is $y = 1$).

$w_0^{(t+1)} \ \square \ w_0^{(t)}$

$w_1^{(t+1)} \ \square \ w_1^{(t)}$

$w_2^{(t+1)} \ \square \ w_2^{(t)}$

c) False Negative (we predict $y = -1$ but the correct label is $y = 1$).

$w_0^{(t+1)} \ \square \ w_0^{(t)}$

$w_1^{(t+1)} \ \square \ w_1^{(t)}$

$w_2^{(t+1)} \ \square \ w_2^{(t)}$

d) Correct Negative Prediction (we predict $y = -1$ and the correct label is $y = -1$).

$w_0^{(t+1)} \ \square \ w_0^{(t)}$

$w_1^{(t+1)} \ \square \ w_1^{(t)}$

$w_2^{(t+1)} \ \square \ w_2^{(t)}$

Problem 3.

Knowing that the training data is linearly separable, describe a situation where a Soft Margin SVM would be preferable to a Hard Margin SVM.

Problem 4.

Extra Problem!

Determine if a perceptron is capable of learning to compute the following logical operators. If so, give a possible value for $\vec{w}$. If not, explain why.
Assume that each input $\vec{x}$ is two dimensional, where x_0 and x_1 can only take on the value of 0 or 1.
(e.g. for the OR operator, a potential input is $\vec{x}^{(i)} = (1, 0)^T$ with $y_i = x_0^{(i)} \vee x_1^{(i)} = 1 \vee 0 = 1$)

- OR
- AND
- XOR