# Lecture 8. Perceptron and Support Vector Machine

Prof. Alan Yuille

Summer 2014

## Outline

1. Perceptron
2. Support Vector Machine

## 1 Perceptron

The Perceptron Algorithm dates back to the 1950's. It was very influential (i.e. hyped) - and unrealistic claims were made about its effectiveness. But it is very important as a starting point for one style of machine learning.

The Perceptron was criticized (Minksy and Papert) because it was not all to represent all decision rules – i.e. you cannot always separate data into positive and negative by using a plane. But, from the point of view of generalization, it is good that perceptrons cannot learn everything! In technical language, this means that perceptrons have *limited capacity* and this enables good generalization for some types of data.

Perceptrons can only represent a restricted set of decision rules (e.g. separation by hyperplane). This is a limitation and a virtue. If we can find a separating hyperplane, then it is probably not due to chance alignment of the data (provided $n > (d+1)$), and so it is likely to generalize. In Learning Theory (Vapnik) the quantity $(d+1)$ is the VC dimension of perceptrons and is a measure of the *capacity* of perceptrons. There is a hypothesis space of classifiers – the set of all perceptrons in this case – and this hypothesis space has a *capacity* which is $d+1$ for perceptrons. To ensure generalization, you need much more training data than the capacity of the hypothesis space that you are using. We will return to this is later lectures.

An alternative is to use Multilevel Perceptron, see previous lecture.

### 1.1 Linear Classifiers

A dataset contains $N$ samples: $\{ (x_\mu, y_\mu) : \mu = 1 \text{ to } N \}$, $y_\mu \in \{\pm 1\}$
Can we find a linear classifier that separates the positive and negative examples?

E.g., a plane $\vec{a} \cdot \vec{x} = 0$, see figure (1), which separates the data with a decision rule $\hat{y}(\vec{x}) = sign(\vec{a} \cdot \vec{x})$
s.t. $\vec{a} \cdot \vec{x}_\mu \geq 0$, if $y_\mu = +1$
s.t. $\vec{a} \cdot \vec{x}_\mu \leq 0$, if $y_\mu = -1$

Plane goes through the origin ($\vec{a} \cdot \vec{0} = 0$) (special case, this can be relaxed by setting $\vec{\hat{a}} = (\vec{a}, b)$ and $\vec{\hat{x}} = (\vec{x}, 1)$ and working in this higher-dimensional $d + 1$ space.)

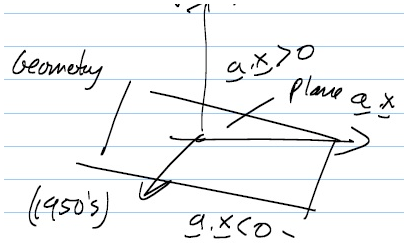


Figure 1: A plane $\vec{a} \cdot \vec{x} =$ separates the space into two regions $\vec{a} \cdot \vec{x} > 0$ and $\vec{a} \cdot \vec{x} < 0$.

## 1.2 The Perceptron Algorithm

First, we need to replace the negative examples by positive examples: if $y_\mu = -1$, set $\vec{x}_\mu \to -\vec{x}_\mu$, $y_\mu \to -y_\mu$.
Require that $\text{sign}(\vec{a} \cdot \vec{x}_\mu) = y_\mu$, which is equivalent to $\text{sign}(-\vec{a} \cdot \vec{x}_\mu) = -y_\mu$).

The perceptron algorithm reduces to finding a plane s.t. $(\vec{a} \cdot \vec{x}_\mu) \geq 0$, for $\mu = 1, ..., N$.

Note: the vector $\vec{a}$ need not be unique. It is better to try to maximize the margin (see later this lecture), which requires finding $\vec{a}$ with $|\vec{a}| = 1$, so that $(\vec{a} \cdot \vec{x}_\mu) \geq m, \ \forall \, \mu = 1, ..., N$ for the maximum value of $m$.

From a geometrical point of view, we can make the following claim.

Claim: If $\vec{a}$ is a unit vector $|\vec{a}| = 1$, then $\vec{a} \cdot \vec{y}$ is the sign distance of $\vec{y}$ to the plane $\vec{a} \cdot \vec{x} = 0$, see figure (2). I.e. $\vec{a} \cdot \vec{y} > 0$, if $\vec{y}$ is above plane; and $\vec{a} \cdot \vec{y} < 0$, if $\vec{y}$ is below plane.

Proof: write $\vec{y} = \lambda \vec{a} + \vec{y}_p$, where $\vec{y}_p$ is the projection of $\vec{y}$ into the plane. By definition $\vec{a} \cdot \vec{y}_p = 0$, hence $\lambda = (\vec{a} \cdot \vec{y})/(\vec{a} \cdot \vec{a}) = (\vec{a} \cdot \vec{y})$, if $|\vec{a}| = 1$.

The perceptron algorithm has the following steps:
Initialize: $\vec{a}(0) = 0$
loop over $\mu = 1$ to $N$
If $\vec{x}_\mu$ is misclassified (i.e. $sign(\vec{a} \cdot \vec{x}_\mu) < 0$), set $\vec{a} \to \vec{a} + \vec{x}_\mu$,
Repeat until all samples are classified correctly.

Note: instead of changing the signs of the data points to require that $\text{sign}(\vec{a} \cdot \vec{x}_\mu) = y_\mu$, the algorithm can be modified to update the weights by the rule: $\vec{a} \to \vec{a} + y\vec{x}_\mu$ (if the datapoint is missclassified). This means that positive missclassified examples update the weights by $\vec{a} \mapsto \vec{a} + \vec{x}_\mu$ (as above) and negative missclassified examples update the weights

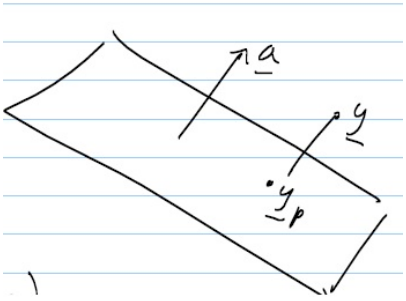


Figure 2: $\vec{y}_p$ is the projection of point $\vec{y}$ onto the plane $\vec{a} \cdot \vec{x} = 0$. This means that $\vec{y} - \vec{y}_p \propto \vec{a}$, i.e. the vector joining $\vec{y}$ and $\vec{y}_p$ is parallel to the normal $\vec{a}$ to the plane. If $|\vec{a}| = 1$, then the sign projection is $\vec{a} \cdot (\vec{y} - \vec{y}_p) = \vec{a} \cdot \vec{y}$ (since $\vec{y}_p$ lies on the plane and so $\vec{a} \cdot \vec{y}_p = 0$). The sign projection is positive $\vec{a} \cdot \vec{y} > 0$ if $\vec{y}$ lies above the plane and is negative $\vec{a} \cdot \vec{y} < 0$ if $\vec{y}$ lies below the plane.

by $\vec{a} \mapsto \vec{a} - \vec{x}_\mu$. Later in the lecture we will see that similar update algorithms arise from more sophisticated formulations (like max margin).

### 1.3 Novikov's Theorem (advanced topic)

Novikov's Theorem: The perceptron algorithm will converge to a solution weight $\vec{a}$ that classifies all the samples correctly (provided this is possible).

Proof.
Let $\hat{\vec{a}}$ be a separating weight ($m > 0$)
Let $m = \min_\mu \quad \hat{\vec{a}} \cdot \vec{x}_\mu$
Let $\beta^2 = \max_\mu |\vec{x}_\mu|^2$

Suppose that $\vec{x}_t$ is misclassified at time $t$ so $\vec{a} \cdot \vec{x}_t < 0$. Then $\vec{a}_{t+1} - (\beta^2/m)\hat{\vec{a}} = \vec{a}_t - (\beta^2/m)\hat{\vec{a}} + \vec{x}_t$.

$\|\vec{a}_{t+1} - \frac{\beta^2}{m}\hat{\vec{a}}\|^2 = \|\vec{a}_t - \frac{\beta^2}{m}\hat{\vec{a}}\|^2 + \|\vec{x}_t\|^2 - 2(\vec{a}_t - \frac{\beta^2}{m}\hat{\vec{a}}) \cdot \vec{x}_t$

Using $\|\vec{x}_t\|^2 \leq \beta^2$, $\vec{a}_t \cdot \vec{x}_t < 0, -\hat{\vec{a}} \cdot \vec{x}_t < -m$, it follows that $\|\vec{a}_{t+1} - \frac{\beta^2}{m}\hat{\vec{a}}\|^2 \leq \|\vec{a}_t - \frac{\beta^2}{m}\hat{\vec{a}}\|^2 + \beta^2 - 2\frac{\beta^2 m}{m}$. Hence $\|\vec{a}_{t+1} - \frac{\beta^2}{m}\hat{\vec{a}}\|^2 \leq \|\vec{a}_t - \frac{\beta^2}{m}\hat{\vec{a}}\|^2 - \beta^2$.

So, each time we update a weight, we reduce the quality $\|\vec{a}_t - \frac{\beta^2}{m}\hat{\vec{a}}\|^2$ by a fixed amount $\beta^2$. But $\|\vec{a}_0 - \frac{\beta^2}{m}\hat{\vec{a}}\|^2$ is bounded above by $\frac{\beta^4}{m^2}\|\hat{\vec{a}}\|^2$. So, we can update the weight at most $\frac{\beta^2}{m^2}|\hat{\vec{a}}^2|$ times. This guarantees convergence.