Let $y = \left\{0, 1\right\}$ and $X_1, \ldots, X_n$ be the features. Then

$$\mathcal{D} = \left\{ \begin{bmatrix} \vec{X}_1 \\ \vec{X}_2 \\ \vdots \\ \vec{X}_n \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \right\}$$

Note that we use $p+1$ because there is a bias of 1 in our model.
We need $y^* = g(\vec{x}^*)$. Use $g = \mathcal{A}(\mathcal{H}, \mathcal{D})$ where

$$\mathcal{H} = \left\{ \mathbb{1}_{\vec{w}\cdot\vec{x}>0}, \vec{w} \in \mathbb{R}^{p+1} \right\}$$

Perceptron Learning Algorithm:

1. Initialize $\vec{w}^{t=0} = \vec{0}$ or random
2. Calculate $\hat{y}_i = \mathbb{1}_{\vec{w}^t\cdot\vec{x}>0}$
3. Update all weights from $j = 1, \ldots, p+1$

$$\begin{aligned} w_1^{t=1} &= w_1^{t=0} + (y_i - \hat{y}_i)1 \\ w_2^{t=1} &= w_2^{t=0} + (y_i - \hat{y}_i)x_{0,1} \\ &\vdots \\ w_{p+1}^{t=1} &= w_{p+1}^{t=0} + (y_i - \hat{y}_i)x_{p+1,1} \end{aligned}$$

4. Repeat steps 2 and 3 for all $i \in \left\{1, \ldots, n\right\}$
5. Repeat steps 2 through 4 until a threshold error is reached or a maximum number of iterations.

Note: If $\mathcal{D}$ is linearly separable ($\exists\vec{w}$ such that $\mathbb{1}_{\vec{w}\cdot\vec{x}>0}$) yields no errors in $\mathcal{D}$, then the algorithm will find it.

Perceptron Diagrams: