Objectives

- Perceptrons Continued
- Introduction to SVM
- XPLs

---

1 Perceptrons Continued

The Perceptron[1] is part of a family of machine learning algorithms known as *supervised learning* algorithms.

1.1 Supervised vs Unsupervised Learning Algorithms

A *supervised learning* algorithm takes in labeled data, from which relationships between the desired output corresponding to the desired inputs are already known, from which the algorithm can make predictions on new, unlabeled data.

Meanwhile, an *unsupervised learning* algorithm takes in unlabeled data, from which it is expected to find relationships and patterns among the data. For our current purposes, let's focus on the different categories of *supervised learning* algorithms.

1.2 Types of Supervised Learning Algorithms

The two primary types of supervised learning algorithms are *regression* algorithms and *classification* algorithms.

1.2.1 Regression Algorithms

Regression algorithms, given an input x, maps it to a y, such that

$$y \in \mathbb{R}.$$

That is, x is mapped to a real number.

1.2.2 Classification Algorithms

Classification algorithms, given an input x, maps it to a y, such that

$$y \in \{C_1, C_2, \ldots, C_n\}.$$

That is, x is mapped to C_i, out of n possible categories. Coincidentally, the Perceptron algorithm belongs to this category. More specifically, it is a *binary classification* algorithm, where

$$y \in \{-1, 1\}.$$

1.3 Perceptrons and Linear Separability

The Perceptron algorithm assumes that the data it takes in is *linearly separable*.

A set of data D, such that $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,, $x_i \in \mathbb{R}^p$,, and $y \in \{-1, 1\}$. is *linearly separable* if and only if one can draw a hyperplane between the two sets, such that

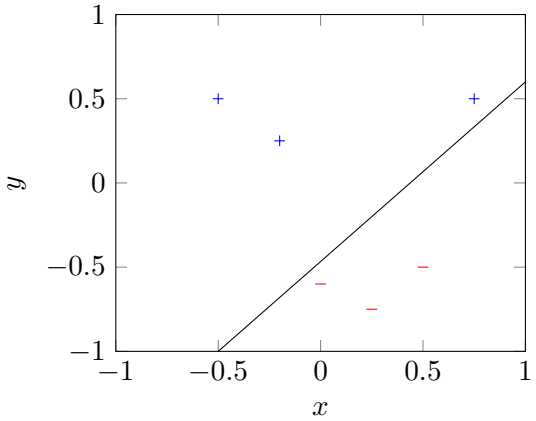
$$y_i(w^\perp x + b) > 0,$$

$$\forall x_i, y_i.$$

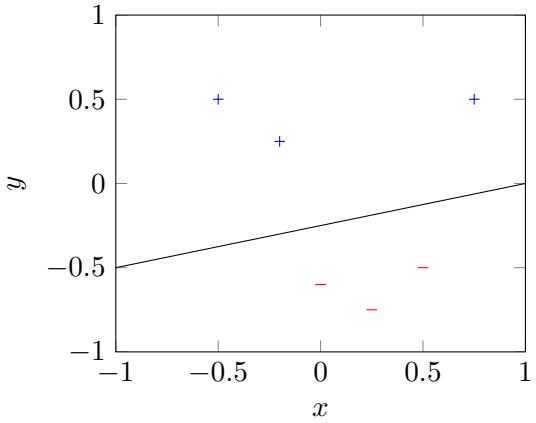
Recall that the sign of $w^\perp x + b$ tells us whether vector x is 'above', 'below', or 'on' the hyperplane. Assuming the hyperplane obtained via the Perceptron is a solution, then the sign should match the sign of the category, so if we were to multiply the category with the equation, we'll either be multiplying two positive or negative numbers together, and we should therefore always obtain a positive number.

1.4 The Set of all Decision Functions

Assuming that the set given to it is linearly separable, the Perceptron algorithm is guaranteed to obtain *a solution*, a hyperplane that successfully splits the two classes. However, as we can see here, this solution is not guaranteed to be the *optimal solution*. Consider the following dataset and decision boundary:



As it can be seen, while the feature vectors in the dataset are correctly categorized, the way the plane lies leaves very little room for error, and appears prone to misclassifications.



Meanwhile, this hyperplane splits the two halves of the feature space way more evenly, leaving more of a margin between the closest feature vectors to the plane. There is an infinite number of possible solutions, an infinite number of possible hyperplanes, and therefore an infinite number of possible *decision functions*, which take the form

$$h(x) = sign(w^{\perp}x + b).$$

If the hyperplane is the set of all points such that $w^{\perp}x + b = 0$, or, the set of all points in the hyperplane, then the sign of $w^{\perp}x + b$ when non-zero tells us whether or not we are 'above' or 'below' the plane, and as a result, which category we belong to. From this, we can define the set of all possible decision functions,

$$H = \{h_1, h_2, \ldots, h_n\}.$$

1.5 Limitations of perceptron

- Fails on non-linearly separable data.
- Single-layer only.

2 Introduction to SVM

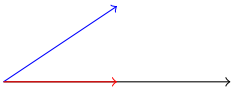
As the previous section shows, an 'optimal' hyperplane maximizes the margin between the closest feature vector and the plane. To better maximize this margin, we'll introduce a new machine learning algorithm called the *Support Vector Machine*[2][3], or *SVM* for short.

The SVM is what is known as a *maximum margin classifier*, which means that it is *guaranteed* to find the maximum margin separating hyperplane, which is what we have decided to define as the most 'optimal' hyperplane for our purposes.

2.1 SVM Prerequisites

Before we delve into the specifics of the SVM algorithm, we must define an important linear algebra operation: *projection*.

In the most literal sense, 'projecting' one vector onto another gives us the magnitude of the first vector in the direction of another, as if you sat the first above the second in the same plane and shined a light over it- the projection is it's 'shadow' (like the example below).



Let u and v be two vectors of some kind, and let us project the first onto the second. If a vector can be defined as a unit direction vector multiplied by some magnitude, we can take the direction of the resulting projection to be the unit vector of the vector we're projecting onto, $\hat{v}$ in this case.

To obtain the magnitude, let us examine the definition of the dot product.

$$u \cdot v = |u||v|cos\theta$$

The dot product can be defined as the magnitudes of both vectors, multiplied by the cosine of the angle between them. Dividing both sides by $|v|$, we obtain

$$\frac{u \cdot v}{|v|} = |u|cos\theta.$$

As it so happens, $|u|cos\theta$ represents the magnitude of u in the direction of v. For example, if u and v are parallel, the angle between them is zero, the cosine of zero is 1, and we get the full magnitude of u. If u and v are perpendicular, the angle between them is $\frac{\pi}{2}$, and the cosine of that is zero, so there is no magnitude of u in the direction of v.

Putting both together, we can define projection to be

$$proj_v u = (\frac{u \cdot v}{|v|})\hat{v} = (\frac{u \cdot v}{|v|})\frac{v}{|v|}.$$

From this, we can also obtain the component of u *perpendicular* to v, $u^{\perp}$, by subtracting the projection from u,

$$u^{\perp} = u - proj_v u.$$

3 XPL's

1.) Modify the perception code to add a counter to see how many updates it takes to obtain any solution.

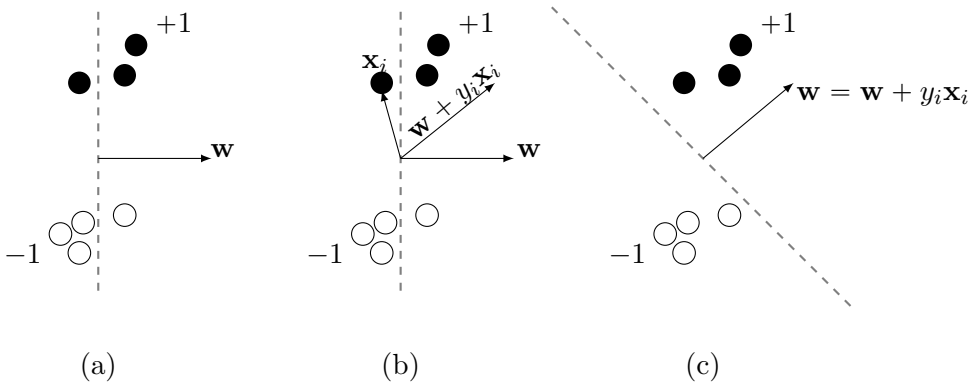


Figure 1: Illustration of a one step update using Algorithm **??**. (a) The hyperplane defined by $\mathbf{w}$ misclassifies one hollow circle and one filled circle. (b) Hyperplane is updated since $\mathbf{x}_i$ was misclassified. (c) Updated hyperplane, $\mathbf{w} + y_i\mathbf{x}_i$, that separates the two classes.

References

[1] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. Technical report, Cornell Aeronautical Laboratory, 1958.

[2] Bernhard Schölkopf and Alexander J. Smola. Learning with kernels: Support vector machines, regularization, optimization, and beyond. *MIT Press*, 2002.

[3] Vladimir N. Vapnik. *Statistical Learning Theory.* Wiley, 1998.

References

[1] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. Technical report, Cornell Aeronautical Laboratory, 1958.

[2] Bernhard Schölkopf and Alexander J. Smola. Learning with kernels: Support vector machines, regularization, optimization, and beyond. *MIT Press*, 2002.

[3] Vladimir N. Vapnik. *Statistical Learning Theory.* Wiley, 1998.