AM 221: Advanced Optimization Spring 2016

Prof. Yaron Singer *Lecture 2 — Wednesday, January 27th*

1 Overview

In our previous lecture we discussed several applications of optimization, introduced basic terminology, and proved Weierstass' theorem (circa 1830) which gives a sufficient condition for existence of an optimal solution to an optimization problem. Today we will discuss basic definitions and properties of convex sets and convex functions. We will then prove the separating hyperplane theorem and see an application of separating hyperplanes in machine learning. We'll conclude by describing the seminal perceptron algorithm (1957) which is designed to find separating hyperplanes.

2 Elements of Convex Analysis

We will primarily consider optimization problems over convex sets – sets for which any two points are connected by a line. We illustrate some convex and non-convex sets in Figure 1.

Definition. *A set S is called a **convex set** if any two points in S contain their line, i.e. for any $\mathbf{x}_1, \mathbf{x}_2 \in S$ we have that $\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2 \in S$ for any $\lambda \in [0,1]$.*

In the previous lecture we saw linear regression an example of an optimization problem, and mentioned that the RSS function has a convex shape. We can now define this concept formally.

Definition. *For a convex set $S \subseteq R^n$, we say that a function $f : S \to \mathbb{R}$ is:*

- ***convex on** S if for any two points $\mathbf{x}_1, \mathbf{x}_2 \in S$ and any $\lambda \in [0,1]$ we have that:*

$$f\left(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2\right) \leq \lambda f(\mathbf{x}_1) + \Big(1-\lambda\Big) f(\mathbf{x}_2).$$

- ***strictly convex on** S if for any two points $\mathbf{x}_1, \mathbf{x}_2 \in S$ and any $\lambda \in [0,1]$ we have that:*

$$f\left(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2\right) < \lambda f(\mathbf{x}_1) + (1-\lambda)\, f(\mathbf{x}_2).$$

We illustrate a convex function in Figure 2. Notice that any linear function is a convex function, but not a strictly convex function. Throughout the course we will see how convex functions provide reasonable models for many real-world objectives and have numerous applications in data science. Importantly, as we will learn in this course, convex functions can be minimized under various constraints in a computationally efficient manner. Why? For convex functions, local optima are also global optima, and this property is extremely helpful for finding optimal solutions. Let's formalize this argument.

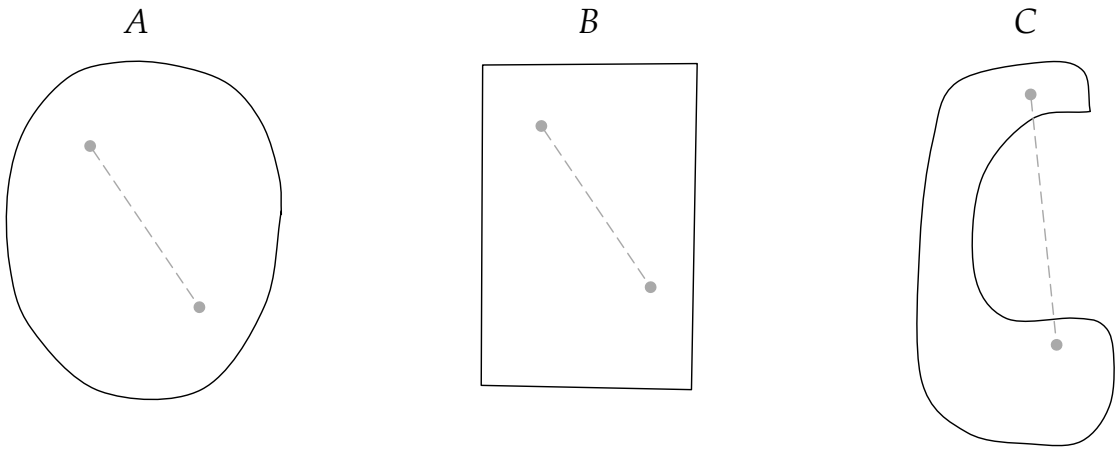

Figure 1: A depiction of convex and non-convex sets. The sets A and B are convex since the straight line between any two points inside them is also in the set. The set C is not convex.

Definition. *For a* maximization *problem* $\bar{\mathbf{x}} \in \mathcal{F}$ *is called a* ***locally optimal solution*** *if* $\exists \epsilon > 0$ *s.t.* $f(\bar{\mathbf{x}}) \geq f(\mathbf{x})$, $\forall \mathbf{x} \in \mathcal{F}$ *s.t.* $\|\bar{\mathbf{x}} - \mathbf{x}\|_2 \leq \epsilon$. *Similarly, for a* minimization *problem* $\bar{\mathbf{x}} \in \mathcal{F}$ *is called an* ***locally optimal solution*** *if* $\exists \epsilon > 0$ *s.t.* $f(\bar{\mathbf{x}}) \leq f(\mathbf{x})$, $\forall \mathbf{x} \in \mathcal{F}$ *s.t.* $\|\bar{\mathbf{x}} - \mathbf{x}\|_2 \leq 0$.

Theorem. *Let* S *be a convex set and* $f : S \to \mathbb{R}$ *a convex function. If* $\bar{\mathbf{x}}$ *be a local optimum of* $\min\{f(\mathbf{x}) \ : \ \mathbf{x} \in S\}$*, then* $\bar{\mathbf{x}}$ *is a global optimum. Furthermore, if the function is strictly convex, then* $\bar{\mathbf{x}}$ *is unique.*

Proof. Let $\mathbf{x}^\star$ be a local optimum and assume for purpose of contradiction that there exists another point $\mathbf{y} \in S$ s.t. $f(\mathbf{y}) < f(\mathbf{x}^\star)$. Consider a strict convex combination $\mathbf{z} = \lambda \mathbf{x}^\star + (1-\lambda)\mathbf{y}$, for some $\lambda \in (0,1)$. From convexity and minimality of $\mathbf{y}$:

$$\begin{aligned} f(\mathbf{z}) &= f(\lambda \mathbf{x}^\star + (1-\lambda)\mathbf{y}) \\ &\leq \lambda f(\mathbf{x}^\star) + (1-\lambda) f(\mathbf{y}) \\ &< \lambda f(\mathbf{x}^\star) + (1-\lambda) f(\mathbf{x}^\star) \\ &= f(\mathbf{x}^\star) \end{aligned}$$

Since $\mathbf{z}$ can be *any* convex combination, it applies to points that are arbitrarily close to $\mathbf{x}^\star$, contradicting x^* being a local optimum. □

Concave functions. Note that if we take the mirror image of the plot above, we have a *concave* function: any two points on the curve lie above the line. That is, $f(\lambda x + (1-\lambda)y) \geq \lambda f(x) + (1-\lambda) f(y)$. So, $f : S \to \mathbb{R}$ is convex function if and only if then $-f(\cdot)$ is a concave function. Thus, if one knows how to minimize a convex function, one can maximize a concave function.

Definition. *For a convex set* $S \subseteq R^n$*, we say that a function* $f : S \to \mathbb{R}$ *is* ***concave on*** S *if for any two points* $\mathbf{x}_1, \mathbf{x}_2 \in S$ *and any* $\lambda \in [0,1]$ *we have that:*

$$f\Big(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2\Big) \geq \lambda f(\mathbf{x}_1) + \Big(1-\lambda\Big) f(\mathbf{x}_2).$$

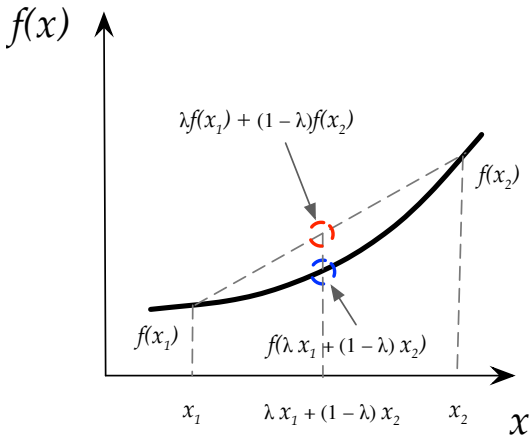

Figure 2: A depiction of a convex function. The red circle indicates the value on the straight line $\lambda f(x^1) + f(1-\lambda)(x^2)$ and the blue indicates the value of the convex function evaluated on the point $\lambda x^1 + (1-\lambda)x^2$, i.e. $f(\lambda x^1 + (1-\lambda)x^2)$.

2.1 The separating hyperplane theorem

We will now prove the separating hyperplane theorem. This theorem is a fundamental result in convex analysis which simply states that for every closed convex set, there is a hyperplane that separates the set with any point outside it. This theorem will be instrumental in proving Farkas' lemma next week which leads to the proof of duality of linear programs, as well as Lagrangian duality that we will cover when we discuss convex optimization. The proof follows two step: first we prove a theorem about the projection of a point on a convex closed set and then apply this result to prove the separating hyperplane theorem. We illustrate these theorems in Figure 3.

Theorem. *Let $C \subseteq \mathbb{R}^n$ be a non-empty closed convex set and $\mathbf{y} \in \mathbb{R}^n \setminus C$. Then:*

(a) There exists a unique point $\mathbf{z} \in C$ s.t. $\mathbf{z} = \operatorname{argmin}_{\mathbf{x}\in C} \|\mathbf{y}-\mathbf{x}\|_2^2$;

(b) $z = \operatorname{argmin}_{\mathbf{x}\in C} \|\mathbf{y}-\mathbf{x}\|$ if and only if $(\mathbf{y}-\mathbf{z})^T(\mathbf{x}-\mathbf{z}) \leq 0,\ \forall \mathbf{x} \in C$.

Proof. The intuition behind the proof of (a) is simple: the distance function is strictly convex and has a unique minimum, which is in C since C is convex and closed. The characterization (b) will follow from the parallelogram law. Let's prove this formally.

(a) Define $P_C(\mathbf{y}) = \operatorname{argmin}_{\mathbf{x}\in C} \|\mathbf{y}-\mathbf{x}\|_2^2$. Without loss of generality, we can assume that C is bounded. This is because if it is not bounded, since it is non-empty there exists some $\mathbf{w} \in S$ and we can simply define $C' = \{\mathbf{x} : \|\mathbf{x}-\mathbf{y}\| \leq \|\mathbf{w}-\mathbf{y}\|\}$ and prove the theorem for C'. From Weirsrass theorem we know that there exists an optimal solution in C. In the problem set you will show that the ℓ_2 distance function is *strictly convex* and that every strictly convex function has a unique optimal solution. Thus, since there exists an optimal solution in Ct, and the objective function is strictly convex we know that $P_C(\mathbf{y})$ is unique.

(b) Recall the law of the parallelogram:

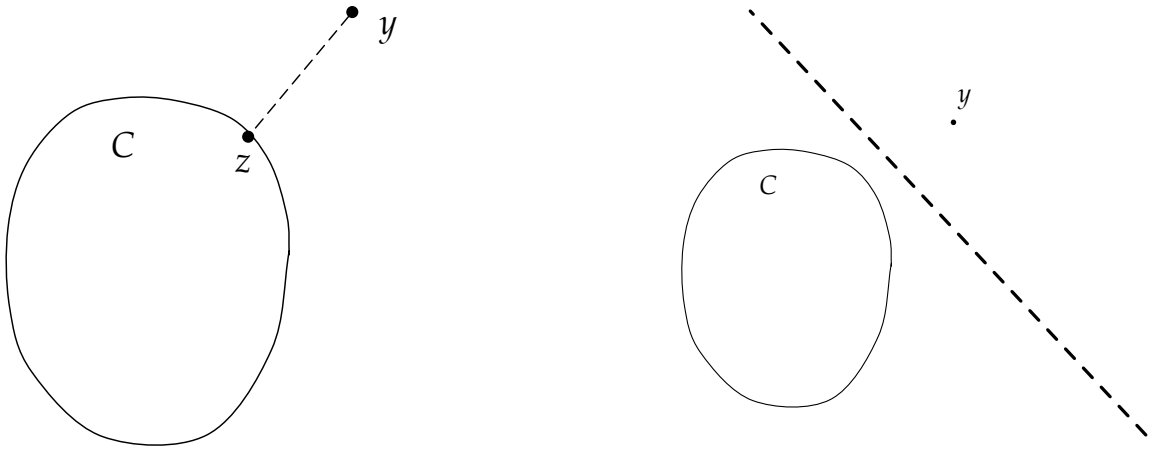

Figure 3: A depiction of a projection of a point $\mathbf{y}$ onto a closed convex set C (left) and a hyperplane (dotted line) separating a point $\mathbf{y}$ and closed convex set C (right).

Fact: law of the parallelogram.

$$\|\mathbf{a}+\mathbf{b}\|^2 = \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2 + 2\mathbf{a}^\intercal\mathbf{b}$$

$$\|\mathbf{a}-\mathbf{b}\|^2 = \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2 - 2\mathbf{a}^\intercal\mathbf{b}$$

For the first direction assume that $(\mathbf{y}-\mathbf{z})^\intercal(\mathbf{x}-\mathbf{z}) \leq 0 \ \forall \mathbf{x} \in C$:

$$\begin{aligned}\|\mathbf{y}-\mathbf{x}\|^2 &= \|\mathbf{y}-\mathbf{z}+\mathbf{z}-\mathbf{x}\|^2 \\ &= \|\mathbf{y}-\mathbf{z}\|^2 + \|\mathbf{z}-\mathbf{x}\|^2 + 2(\mathbf{y}-\mathbf{z})^\intercal(\mathbf{z}-\mathbf{x})\end{aligned}$$

Notice that $\|\mathbf{z}-\mathbf{x}\|^2$ is always positive, and by our assumption $(\mathbf{y}-\mathbf{z})^\intercal(\mathbf{z}-\mathbf{x}) \geq 0$, hence:

$$\|\mathbf{y}-\mathbf{x}\|^2 \geq \|\mathbf{y}-\mathbf{z}\|^2.$$

For the converse, assume that $\mathbf{z} = P_C(\mathbf{y})$. Let $\mathbf{x} \in C$ and for some $\lambda \in [0,1]$ define the point: $\mathbf{z}+\lambda(\mathbf{x}-\mathbf{z})$. Notice that this point is in C since C is convex. By minimality of $\mathbf{z}$ we know that $\|\mathbf{y}-\mathbf{z}\|^2 \leq \|\mathbf{y}-\mathbf{z}+\lambda(\mathbf{x}-\mathbf{z})\|^2$. Thus:

$$\begin{aligned}\|\mathbf{y}-\mathbf{z}\|^2 &\leq \|\mathbf{y}-\mathbf{z}+\lambda(\mathbf{x}-\mathbf{z})\|^2 \\ &= \|\mathbf{y}-\mathbf{z}\|^2 + \|\lambda(\mathbf{x}-\mathbf{z})\|^2 - 2(\mathbf{y}-\mathbf{z})^\intercal\lambda(\mathbf{x}-\mathbf{z}) \\ &= \|\mathbf{y}-\mathbf{z}\|^2 + \lambda^2\|(\mathbf{x}-\mathbf{z})\|^2 - 2\lambda(\mathbf{y}-\mathbf{z})^\intercal(\mathbf{x}-\mathbf{z})\end{aligned}$$

Rearranging we get that:

$$2(\mathbf{y}-\mathbf{z})^\intercal(\mathbf{x}-\mathbf{z}) \leq \lambda\|(\mathbf{x}-\mathbf{z})\|^2$$

Since this holds for any $\lambda \in [0,1]$, it holds for $\lambda = 0$, and thus indeed $(\mathbf{y}-\mathbf{z})^\intercal(\mathbf{x}-\mathbf{z}) \leq 0$. □

We will now apply the characterization of the above theorem to show that between any closed convex set and point outside that set there is a separating hyperplane.

Definition. *Given* $\mathbf{a} \in \mathbb{R}^n \neq \mathbf{0}$ *and* $\alpha \in \mathbb{R}$*, the set of points* $H = \{\mathbf{x} \in \mathbb{R}^n \ : \ \mathbf{a}^\intercal \mathbf{x} = \alpha\}$ *is called the* ***hyperplane*** *defined by* $\mathbf{a}$ *and* α*.*

Definition. *A hyperplane* $H = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{a}^\intercal \mathbf{x} = \alpha\}$ *defines* ***half-spaces*** $H^+ = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{a}^\intercal \mathbf{x} \geq \alpha\}$ *and* $H^- = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{a}^\intercal \mathbf{x} \leq \alpha\}$*.*

Theorem. *Let* $C \subseteq \mathbb{R}^n$ *be a non-empty, closed, convex set and* $\mathbf{y} \in \mathbb{R}^n \setminus C$*. Then, there always exists a hyperplane that separates* $\mathbf{y}$ *and* C*, i.e.* $\exists \mathbf{a} \in \mathbb{R}^n \neq 0, \alpha \in \mathbb{R}$ *s.t.* $\mathbf{a}^\intercal \mathbf{x} \leq \alpha \leq \mathbf{a}^\intercal \mathbf{y}$ $\forall \mathbf{x} \in C$*, i.e.* $C \subseteq H^-, \mathbf{y} \in H^+$*.*

Proof. Let $z = P_C(\mathbf{y})$ and define $\mathbf{a} = \mathbf{y} - \mathbf{z}$ and $\alpha = \mathbf{a}^\intercal \mathbf{z}$. We know that for any $\mathbf{x} \in C$ we have that: $(\mathbf{y} - \mathbf{z})^\intercal (\mathbf{x} - \mathbf{z}) \leq 0$, hence:

$$\begin{aligned} & \mathbf{a}^\intercal (\mathbf{x} - \mathbf{z}) \leq 0 \\ \iff & \mathbf{a}^\intercal \mathbf{x} \leq \mathbf{a}^\intercal \mathbf{z} \\ \iff & \mathbf{a}^\intercal \mathbf{x} \leq \alpha \end{aligned}$$

For $\mathbf{y}$ we have that:

$$\begin{aligned} \mathbf{a}^\intercal \mathbf{y} - \alpha &= \mathbf{a}^\intercal \mathbf{y} - \mathbf{a}^\intercal \mathbf{z} \\ &= \mathbf{a}^\intercal (\mathbf{y} - \mathbf{z}) \\ &= (\mathbf{y} - \mathbf{z})^\intercal (\mathbf{y} - \mathbf{z}) \\ &= \|\mathbf{y} - \mathbf{z})\|^2 \\ &> 0 \end{aligned}$$

Therefore $\mathbf{a}^\intercal \mathbf{x} \leq \alpha$ and $\mathbf{a}^\intercal \mathbf{y} > \alpha$, as required. □

3 Linear Classification and the Perceptron Algorithm

Consider a data set which contains the number of courses a student is currently taking, the number of seminars she attends on a regular basis (on average), as well as whether this student is an undergraduate student or graduate student. Suppose the goal is now to build a classifier which given data on the number of courses and seminars the student is attending can predict whether this student is a graduate student or an undergraduate. Notice that unlike linear regression where we tried to predict a real variable (in the previous lecture's example we tried to predict height), here we aim to predict a binary variable (whether a student is a graduate (1) or undergraduate (0)).

Classification. The above example is one of the most fundamental tasks in machine learning known as *classification*. In the most basic setting, we are given data points $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ (called *training data*) where $\mathbf{x} \in \mathbb{R}^d, y_i \in \{0, 1\}$, and the goal is to construct an algorithm which given a new data point $\mathbf{x}_j$ can predict reasonably well whether $\mathbf{x}_j$ belongs to group 0 or group 1.

Linear classification and separating hyperplanes. In order to build a classifier, a classic approach in machine learning is to seek a *separating hyperplane*. This hyperplane would ideally

separate the data points into two separate regions H^+ and H^-, where one region would correspond to one class (e.g. graduate students) and the other to the other class (undergraduates). The separating hyperplane theorem says that if the data points can be represented as two convex sets, such a hyperplane exists.

3.1 The Perceptron algorithm

If we want to use the machinery from the previous section to find a separating hyperplane we need an algorithm that finds $P_C(\mathbf{y})$. Doing so, requires finding the point $\mathbf{z} \in C$ which minimizes the distance function to $\mathbf{y}$. Since the distance function is (strictly) convex, this is a convex optimization problem. Once we learn how to design algorithms for convex optimization we will be able to design an algorithm for finding this point. But at this point, we don't know such algorithms. Luckily, the perceptron algorithm finds this point for us.

A brief description. The algorithm receives a data set of points $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ where $\mathbf{x}_i \in \mathbb{R}^d, y_i \in \{0, 1\}$, and returns a weight vector $\mathbf{w}$ and parameter $\alpha \in \mathbb{R}$ s.t. $\mathbf{w}^\intercal \mathbf{x}_i > \alpha$ for all $i : y_i = 1$ and $\mathbf{w}^\intercal \mathbf{x}_j \leq \alpha$ for all $j : y_j = 0$. The algorithm begins with a data initialization step which transforms $\mathcal{D}$ to a data set $\mathcal{D}'$ in a manner described below.

Data initialization. In order to describe the perceptron algorithm we will perform two initialization steps that will simplify the description of the algorithm. First, rather than find a vector in $\mathbb{R}^d$ and parameter $\alpha \in \mathbb{R}$, we will simply return one vector in $\mathbb{R}^{d+1}$ where the first d entries describe the vector and the $d+1$ entry is the parameter α. To do so we add another entry to each data point $\mathbf{x}_i$ whose value is -1. Doing so ensures that the vector $\mathbf{w}$ of dimension $d+1$ we return respects $\mathbf{w}^\intercal \mathbf{x}'_i > 0 \iff \sum_{j=1}^{d} w_j \cdot x_j > \alpha$. The second initialization step will multiply by -1 the entries of the data points $\mathbf{x}'_i$ with label $y_i = 0$. Doing so ensures us that if all new points $\mathbf{x}'_i$ respect $\mathbf{w}^\intercal \mathbf{x}'_i > 0$ then the points $\mathbf{x}_i$ whose label was $\mathbf{y}_i = 1$ are in H^+ and the points with label 0 are in H^-. We describe the algorithm below.

Algorithm 1 PERCEPTRON

Input: training data $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$

1: $\mathcal{D}' \leftarrow$ INITIALIZEDATA($\mathcal{D}$)
2: $\mathbf{w} \leftarrow \mathbf{0}$
3: **while** $\exists \mathbf{x}'_i \in \mathcal{D}' : \mathbf{w}^\intercal \mathbf{x}'_i \leq 0$ **do**
4: $\quad \mathbf{w} \leftarrow \mathbf{w} + \frac{\mathbf{x}'_i}{\|\mathbf{x}'_i\|}$
5: **end while**
6: **return** $\mathbf{w}$

BONUS MATERIAL (not covered in class): why Perceptron works. It is quite remarkable to think that such a simple algorithm actually finds a separating hyperplane.

Necessary Condition. An important necessary condition for the algorithm to converge is that the dataset is linearly separable, *i.e* there exists a separating hyperplane. After the data initializa-

tion step, this condition can be expressed as:

$$\exists w, \quad \forall i \in [n], \quad \mathbf{w}^\intercal \mathbf{x}'_i > 0$$

We will henceforth that this condition holds.

Theorem. *The perceptron algorithm terminates and finds a separating hyperplane.*

Proof. First, observe that by the condition of the while loop, the fact that the algorithm terminates necessarily means that $\mathbf{w}^\intercal \mathbf{x}'_i > 0$ for all the points $\mathbf{x}'_i \in \mathcal{D}'$. By the second initialization step this implies that we indeed found a separating hyperplane: all points whose label is 1 respect $(w_1, \ldots, w_d)^\intercal \mathbf{x}_i \geq w_{d+1}$ and since for all points with label $y_i = 0$ we have that $\mathbf{w}^\intercal \mathbf{x}'_i = (-1) \cdot \mathbf{w}^\intercal \mathbf{x}_i$ and $(w_1, \ldots, w_d)^\intercal \mathbf{x}_i > w_{d+1}$ this implies that all points with label 0 respect $(w_1, \ldots, w_d)^\intercal \mathbf{x}_i < w_{d+1}$. So what remains is to show that this is indeed achieved in a finite number of steps.

Let $\{\mathbf{x}'_i\}_{i=1}^m$ be the points after the initialization step and define $\mathbf{w}^\star = \text{argmax}_{\mathbf{w}} \min_{i \in [m]} \frac{\mathbf{w}^\intercal \mathbf{x}'_i}{\|\mathbf{x}'_i\| \cdot \|\mathbf{w}\|}$ and $\nu = \min_{i \in [m]} \frac{\mathbf{w}^{\star\intercal} \mathbf{x}'_i}{\|\mathbf{x}'_i\| \cdot \|\mathbf{w}^\star\|}$. Note that by the necessary condition stated above the theorem, we have that $\nu > 0$. For every iteration j of the while loop, let $\mathbf{w}(j)$ denote new the weight created (i.e, $\mathbf{w}(0) = \mathbf{0}$ and $\mathbf{w}(j) = \mathbf{w}(j-1) + \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|}$). The proof will follow from a few simple arguments about the quantities $\mathbf{w}(j)^\intercal \mathbf{w}^\star$ and $\|\mathbf{w}(j)\|$.

First, since $\mathbf{w}^\star$ is a unit vector we have that $\mathbf{w}(j)^\intercal \mathbf{w}^\star \leq \|\mathbf{w}(j)\|$ (as you recently proved in a problem set). At each step $\mathbf{w}(j)^\intercal \mathbf{w}^\star$ increases by at least ν since:

$$\left(\mathbf{w}(j-1) + \frac{\mathbf{x}'_i}{\|\mathbf{x}'_i\|}\right)^\intercal \mathbf{w}^\star = \mathbf{w}(j-1)^\intercal \mathbf{w}^\star + \frac{\mathbf{x}_i'^\intercal \mathbf{w}^\star}{\|\mathbf{x}'_i\|} \geq \mathbf{w}(j-1)^\intercal \mathbf{w}^\star + \nu$$

Notice that $\|\mathbf{w}(j)\|^2$ never increases by more than 1 in a given step since from the parallelogram law and from the fact that in an update step it must be that $\frac{\mathbf{x}_i'^\intercal \mathbf{w}}{\|\mathbf{x}'_i\|} \leq 0$ we get:

$$\|\mathbf{w}(j+1)\|^2 = \left\|\mathbf{w}(j) + \frac{\mathbf{x}'_i}{\|\mathbf{x}'_i\|}\right\|^2 = \|\mathbf{w}(j)\|^2 + \left\|\frac{\mathbf{x}'_i}{\|\mathbf{x}'_i\|}\right\|^2 + 2\frac{\mathbf{w}(j)^\intercal \mathbf{x}'_i}{\|\mathbf{x}'_i\|} \leq \|\mathbf{w}(j)\|^2 + 1$$

Thus, after t steps we have that $\mathbf{w}(t)^\intercal \mathbf{w}^\star \geq \nu \cdot t$ and $\|\mathbf{w}(t)\| \leq \sqrt{t}$. The fact that $\mathbf{w}(j)^\intercal \mathbf{w}^\star \leq \|\mathbf{w}(j)\|$ implies that $t\nu \leq \sqrt{t}$ and hence $t \leq 1/\nu^2$. □

A couple of caveats. Before we conclude it's we should mentioned two drawbacks of the perceptron algorithm. The first is that it necessitates the data set to be separable. That is, in order for this algorithm to work, the set of points $\mathbf{x}_i$ labeled $y_i = 1$ and those labeled $y_i = 0$ must be convex and disjoint from one another. Otherwise, the separating hyperplane theorem does not hold (can you think of some examples?). The second drawback is that in the worst case the running time of the algorithm may be exponential. That is, for a data set of n points the algorithm may need to make on the order of 2^n steps before it terminates. The first drawback is a major concern, and there are variations of the perceptron algorithm so that it can handle data sets that are not separable. The most notable variation is the Support Vector Machine (SVM) algorithm which we will cover later this semester. The second drawback is a "worst case" argument – one can adversarially construct cases in which the running time will be exponential in the size of the data set.

3.2 Neural networks

The perceptron algorithm is a simple example of a *neural network*. Given a data point $\mathbf{x}_j \in \mathbb{R}^d$, for $\mathbf{x}'_j = (\mathbf{x}_j, 1)$ once the weights have been learned, the function $f(\mathbf{x}'_j) = \sum_{j=1}^{d+1} w_i \cdot x'_{ij}$ vaguely mimics a neuron. An artificial neural network is simply a chaining of perceptrons (see further reading for more information).

4 Further Reading

Basic elements of convex analysis are extensively covered in Chapters 2 and 3 of Boyd and Vandenberghe. To read more about the perceptron algorithm and neural networks you may find Chapter 4 in Mitchell's Machine Learning textbook useful: `http://www.cs.cmu.edu/~tom/mlbook.html`