Linear Predictors, Part 1

September 20, 2021

When compared with nearest-neighbor predictors, linear predictors are near the opposite end of the spectrum of machine learning algorithms. They have a very restricted hypothesis space and are resilient to overfitting as a consequence. They are trained by function optimization and the optimization problem is convex. They are very fast to use at test time.

In contrast with 1-nearest-neighbor predictors, for which the regressor and the classifier are the same, the linear regressor and the linear classifier introduced below are different from each other. They are also "linear" in two different ways. Specifically, the linear regressor (Section 1) fits an affine function (a linear function plus a constant, $h(\mathbf{x}) = b + \mathbf{w}^T\mathbf{x}$) to the available data points.[1] In addition, the linear regressor is "linear" also in that the solution is found by solving a linear system of equations.

The linear classifier introduced in Section 2 is binary (that is, the label space Y has two elements), and it is called the *logistic-regression classifier*. It uses a hyperplane as a class separator (decision boundary) in the data space X. Hyperplanes are represented by affine functions, hence the qualifier "linear" for this classifier.

While nearest-neighbor classifiers can handle label sets Y with any number of elements, linear classifiers need a bit of work to adapt to the *multi-class* case $|Y| > 2$, as shown in Part 2 of the notes on linear predictors.

1 The Least-Squares Linear Regressor

Linear regression requires little discussion, given our previous treatment of polynomial data fitting. Indeed, a linear regressor simply fits an affine polynomial

$$h_{\mathbf{v}}(\mathbf{x}) = b + \mathbf{w}^T\mathbf{x} \quad \text{for} \quad \mathbf{x} \in \mathbb{R}^d$$

to the training set

$$T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\} \subset \mathbb{R}^d \times \mathbb{R} \, .$$

In this expression, we defined the *parameter vector*

$$\mathbf{v} = \begin{bmatrix} b \\ \mathbf{w} \end{bmatrix} \quad \text{where } b \in \mathbb{R} \text{ and } \mathbf{w} \in \mathbb{R}^d,$$

and the hypothesis space $\mathcal{H}$ is the set

$$\mathcal{H} = \{h_{\mathbf{v}} \ : \ \mathbf{v} \in \mathbb{R}^{d+1}\}$$

[1] Technically, affine functions are not linear, since if f is affine, then $h(\mathbf{x}+\mathbf{y}) \neq h(\mathbf{x}) + h(\mathbf{y})$ in general. However, the mathematical literature often uses the word "linear" to denote an affine function.

of all affine functions on $\mathbb{R}^d$. The graph of an affine function on $\mathbb{R}^d$ is a hyperplane in $\mathbb{R}^{d+1}$. For instance, when $d = 1$, the graph of $y = h_{\mathbf{v}}(x) = b + wx$ is a line on the (x, y) plane and is identified by two parameters $\mathbf{v} = (b, w)$.

As in any fitting problem, one must specify a loss function $\ell(y, y')$ that measures the cost incurred when $h(\mathbf{x}) = y'$ while the true value associated to $\mathbf{x}$ is y. The parameters of the regressor are then determined so as to minimize the *empirical risk* on T:

$$L_T(\mathbf{v}) = \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, h_{\mathbf{v}}(\mathbf{x}_n))$$

over all choices of $\mathbf{v} \in \mathbb{R}^{d+1}$.

We choose again the quadratic loss

$$\ell(y, y') = (y - y')^2$$

not because this is particularly appropriate (we don't even know what the underlying task is!), but because it leads to a simple solution. Specializing our previous discussion of polynomial data fitting to the affine case (multivariate polynomial of degree $k = 1$), the vector $\hat{\mathbf{v}}$ of $d+1$ real-valued parameters that minimizes the empirical risk on T can be found by solving the system

$$A\mathbf{v} = \mathbf{a} \tag{1}$$

in the Least-Squares sense:

$$\hat{\mathbf{v}} = \arg\min_{\mathbf{v}} \|A\mathbf{v} - \mathbf{a}\|^2 \,. \tag{2}$$

In these expressions,

$$A = \begin{bmatrix} 1 & \mathbf{x}_1^T \\ \vdots & \\ 1 & \mathbf{x}_N^T \end{bmatrix} \quad \text{and} \quad \mathbf{a} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} .$$

As you know from linear algebra, the solution can be found by solving the *normal equations*

$$A^T A\mathbf{v} = A^T \mathbf{a} \,,$$

a square, $(d+1) \times (d+1)$ system that is invertible if and only if A has rank $r = d + 1$.

Alternatively, and more generally and with greater numerical accuracy, the solution can be found by computing the pseudo-inverse $A^\dagger$ of A. When $r < d+1$ (that is, when A is not full rank), then multiple solutions exist, and the solution computed through the pseudo-inverse is the solution of smallest norm.[2] However, there are typically more training samples than parameters, that is $N > d+1$, and, since the entries in A and $\mathbf{a}$ come from (typically noisy) measurements, the matrix A is likely to be full rank in practice, $r = d + 1$.

Figure 1 illustrates linear regression in one dimension, $d = 1$. The quadratic loss is sensitive to outliers. For instance, removing just two samples out of the 1379 training samples fitted in Figure 1 (a) changes the fitting line quite a bit. Either way, is a linear fit good for this data? Compare with kNN regression, for a different hypothesis space, and with Figure 1(b), where a line is fit to smaller homes in a narrower price range and in a single neighborhood.

[2] More on the pseudo-inverse can be found in Chatpter 2 of the linear systems refresher on the class syllabus page. Also, please note that "solution of smallest norm" here refers to minimizing the norm of $\mathbf{v}$ among all $\mathbf{v}$ that minimize the residual in equation 2.

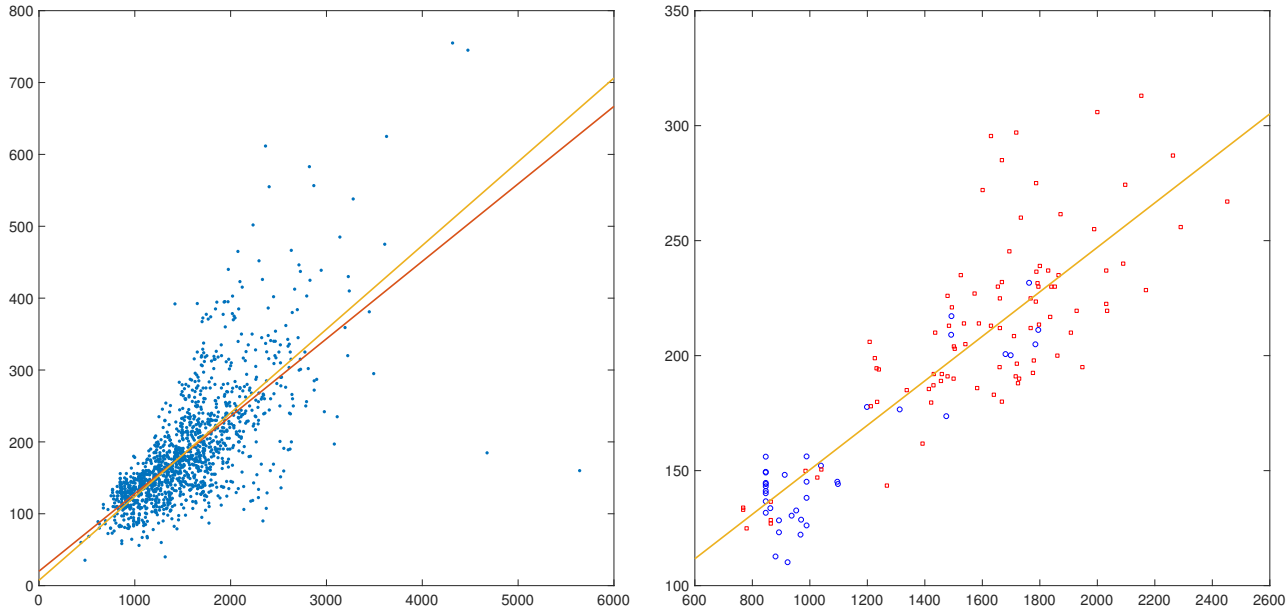


Figure 1: (a) An example of linear regression on the Ames, Iowa home data used in an earlier note. The horizontal axis denotes the gross living area in square feet of a set of 1379 homes in Ames, Iowa, between 2006 and 2010. The vertical axis is the sale price in thousands of dollars [1]. The dark orange line fits all the data points. The yellow line fits all the data points except for the two homes larger than 4500 square feet. (b) Line fit to homes in a smaller range of sizes and prices in a single neighborhood of Ames, College Circle. The colors and styles of the markers are irrelevant here, and will come up in later Figures.

All that the regression lines in Figure 1 show is that price grows with size, which is to be expected. The regression in Figure 1(b) looks reasonably good, in that the points are relatively tightly clustered around the yellow line. There are of course several ways to measure the quality of the fit. One of them is the *residual risk*, that is, the risk $L_T(\hat{\mathbf{v}})$ of the regressor. This is called the "residual" risk because it is the risk remaining even after the optimal linear regressor has been found. The residual risk is the sum of squares $\|A\hat{\mathbf{v}} - \mathbf{a}\|^2$ (see equation 2), and has the unit of thousands of dollars squared. The corresponding standard deviation is more meaningful (thousands of dollars), so one often reports the square root of the residual risk.

For the fit in Figure 1(a), the square root of the residual risk is \$55,800. For the fit in Figure 1(b) it is significantly lower, \$23,600. One way to explain this discrepancy is that the size of a home is generally not a good predictor of its price. As any realtor will tell you, location is a much more important price factor than size. The homes plotted in panel (b) of the Figure are in about the same location, the College Circle neighborhood, and, *given* location, size is a better predictor of price than it is in a broader context.

Note that the only parameters in linear regression are those computed by the training algorithm, which fits the parameters to the training set T, since we are told that an affine function (polynomial of degree 1) is desired. In contrast, the parameter k for k-nearest-neighbors regression must be set separately, and is therefore called a *hyper-parameter*. You have seen ways to select hyper-parameters earlier in the course.

2 The Binary Logistic-Regression Classifier

A binary linear classifier is a classifier for a binary problem whose decision boundary is a hyperplane. "Binary" means that there are two classes,

$$Y = \{c_0, c_1\} .$$

A first idea for building a binary classifier based on linear fitting techniques is to run the linear regressor on the $(\mathbf{x}, y)$ data samples in the training set to obtain a linear regressor $y \approx s(\mathbf{x})$, which we can call the *score*. This fitting presents no technical difficulty if c_0 and c_1 have numerical values, say,

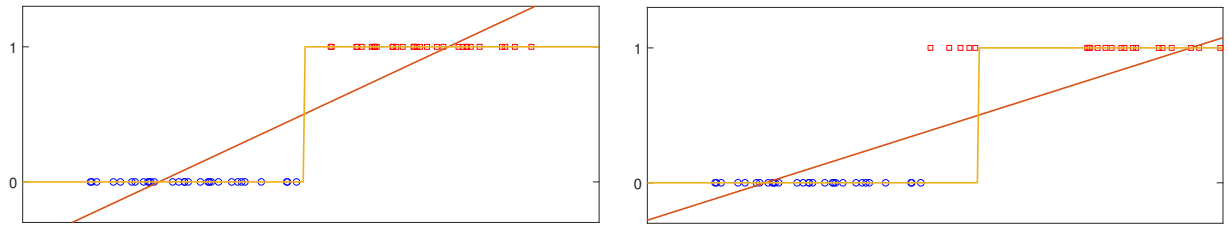
$$c_0 = 0 \quad \text{and} \quad c_1 = 1 .$$

Otherwise, one can remap c_0 to 0 and c_1 to 1 without altering the essence of the problem. The resulting function is linear, so unless it is constant, it can assume values 0 and 1 only on two hyperplanes[3] in the data space X, so the score cannot be a classifier by itself. However, if the data in the training set are even just very roughly linearly separable, we can expect $s(\mathbf{x})$ to take on greater values where the density of values $y_n = 1$ is greater than that of values $y_n = 0$, and *vice versa*. The score function can then be transformed into a classifier by thresholding at 1/2:

$$h(\mathbf{x}) = \begin{cases} c_0 & \text{if } s(\mathbf{x}) \geq 1/2 \\ c_1 & \text{otherwise} \end{cases}$$

as shown in Figure 2 (a) when $d = 1$.

[3]The isocontours of an affine function are hyperplanes. For instance, when $d = 2$, the score $s(\mathbf{x})$ is constant along suitable lines on the plane. There is one line where the score is 0 and one where the score is 1. These lines have (affine) equations $s(\mathbf{x}) = 0$ and $s(\mathbf{x}) = 1$.



(a) (b)

Figure 2: A score function (dark orange) obtained by linear regression from the blue circles and red squares, assuming that blue circles map to zero and red squares map to one. The yellow plot is the classifier obtained by thresholding the score function at $y = 1/2$. (a) The training samples are evenly distributed in x, and the classifier's decision boundary does well on this linearly separable training set. (b) The red squares are unevenly distributed in x, and the same threshold yields a classifier that mislabels several samples, even if the training set is still linearly separable.

This method for turning a regressor into a classifier, where the classifier is a thresholded version of the regressor, is quite general, and the resulting classifier is called *score-based*, as we saw in an earlier note.

The specific idea of using a linear regressor as the score, however, is not great, because the decision hyperplane depends on the distribution of the data in ways that ought to be irrelevant for classification, as we discuss next. This issue arises even when the data space X is one-dimensional, $d = 1$, and is illustrated in Figure 2.

The source of this problem is the fact that a binary classifier is a function that takes on one of two values, 0 and 1. If the classifier is linear, this function is a step function: It is equal to zero on one side of a hyperplane in X, and to one on the other side. In one dimension, a classifier looks like the yellow step function plotted in Figure 2 (a): A hyperplane is a single point when $d = 1$, and that point in the Figure is the value of x where the rise of the step occurs.

In contrast, an affine function (dark orange plot in Figure 2 (a)) is entirely inadequate for approximating a step function, because it grows indefinitely in both directions (towards positive infinity on one side and negative infinity on the other). As a result, training samples that are far from where the rise of the step ought to be placed will skew the score function in ways that have nothing to do with the optimal position of the rise. The position of the rise is the only factor that affects the classifier's error rate, and therefore training points close to the rise ought to have more impact than those that are far away.

Clearly, a better solution would be to fit a step function to the training set, rather than an affine function. It can be shown that the problem of fitting a step function is a linear program in the linearly separable case, and therefore admits an efficient solution. Unfortunately, real data is only rarely linearly separable, and the step-fitting problem can be shown to be NP-hard in that case.

To circumvent this difficulty, the logistic-regression classifier replaces the step with a smooth step-like function. Together with a judicious choice for the loss function, this change results in a convex optimization problem that can be solved efficiently. Moreover, these choices can be justified on the basis of probabilistic considerations, given suitable assumptions on the distribution of the training data. We follow this treatment below, after a remark about the type of score function

needed for the classifier to have a linear decision boundary.

Score Functions for Linear Boundaries When the data space X has one dimension, $d = 1$, as in Figure 2, we only need the score function to have a single zero crossing to ensure a single point as a decision boundary: Everything on one side of the zero crossing is classified as c_0, and everything else as c_1.

In more than one dimension, $d > 1$, the shape of the boundary depends on the form of the score function in greater detail. Since we want the boundary to be a hyperplane (because we look for a linear classifier), we must choose the set of possible score functions $s(\mathbf{x})$ so that the solution to the equation

$$s(\mathbf{x}) = 1/2 \tag{3}$$

is always a hyperplane in X. This can be achieved by defining s to be the composition of an affine function $a : \mathbb{R}^d \to \mathbb{R}$ and a monotonic function $\sigma : \mathbb{R} \to \mathbb{R}$ that crosses the value $1/2$:

$$s(\mathbf{x}) = \sigma(a(\mathbf{x})) = \sigma(c + \mathbf{u}^T\mathbf{x}) \quad \text{for some real number } a \text{ and real-valued vector } \mathbf{u} \in \mathbb{R}^d.$$

The zero crossing of $a(\mathbf{x})$ is a hyperplane, and a monotonic function σ composed with a can only move that hyperplane parallel to itself, without changing its shape or orientation. Specifically, if $\alpha = \sigma^{-1}(1/2)$ is the unique (because of the monotonicity of σ) number for which $\sigma(\alpha) = 1/2$, then equation 3 is equivalent to

$$c + \mathbf{u}^T\mathbf{x} = \alpha \ ,$$

which is a hyperplane in the data space X, as desired. (**End of Remark**).

As a motivating example for an approach to logistic-regression classification based on probabilities, the plot in Figure 3(a) shows a training set

$$T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$$

related the the Ames home data we saw earlier. Each of 127 homes costing less than \$400,000 and smaller than 3,000 square feet of gross living area in the College Circle neighborhood in included. A home is represented by a red square if the home was of high quality and in relatively good condition when it was sold, and by a blue circle otherwise. Quality and condition are assessed by house inspectors.

Intuition suggests that while neither sale price nor size are good predictors of home quality and condition by themselves, their combination may be more informative, since a large house in good conditions is likely to sell at a higher price than a small house in the same conditions. Of course, there are many other factors at play, location being perhaps the most important: This is why we limit this analysis to homes from a single neighborhood, to keep our reasoning simple.

The coordinates of a home on the plot are its size in square feet (x_1) and its sale price in thousands of dollars (x_2). Thus, now both size and price are considered part of the data space X, and the label y encodes condition. Please contrast this with the situation we encountered for linear regression on this data, where there was only one x (size of the home) and price was the target variable y.

The data at our disposal[4] does not include a quantitative measure of the realtor's quality and condition assessment (which would be some real-valued "quality index"), but only whether the home

[4]At least in this example.

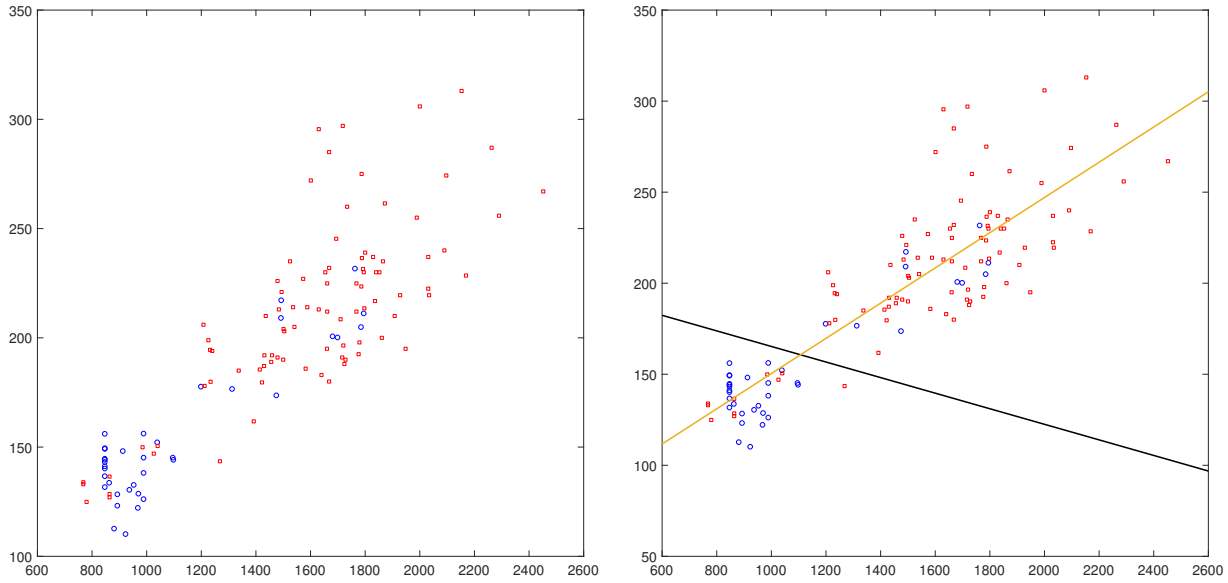


Figure 3: (a) A training set constructed from the Ames home data [1]. The horizontal axis is the size of the house, measured in square feet of gross living area. The vertical axis is the sale price in thousands of dollars. A data point is a blue circle if the overall quality and condition of the house is below average, as established by a professional assessor, and it is a red square otherwise. Only homes from the College Circle are included. (b) The black line is the boundary for the logistic-regression classifier trained on the data in (a). The gold-colored line is the regression line between the size and price of the homes in (a), and is the same line as in Figure 1(b).

is "good" (red squares) or "bad:" We want a classifier that reproduces the realtor's assessment, not a regressor. If we are given size and sale price of a new home, we want to use our classifier to predict whether that is a house in good or poor conditions.

In addition, since we want a *linear* classifier, we are looking for a single hyperplane (in our example, a line) in the data space X (the plane of all home sizes and prices) that separates the two classes (good and bad homes) as well as possible.

In a probabilistic interpretation, a logistic-regression classifier for a binary problem like this assumes that one can quantify the *probability* that a data point $\mathbf{x}$ belongs to either class, and that this probability is an increasing function of the distance of $\mathbf{x}$ from the separating hyperplane, on the appropriate side of it. Specifically, the probability $p = \mathbb{P}[\text{"good home"}]$ is $1/2$ on the separating hyperplane. It increases towards 1 as we move away from the hyperplane on one side of it, and it decreases towards 0 as we move away from the hyperplane on the other side. Of course, the probability $1 - p = \mathbb{P}[\text{"bad home"}]$ behaves complementarily.

To complete this picture, we need three ingredients:

- Find a way to compute the distance Δ of a point $\mathbf{x} \in X$ from a hyperplane χ, and the side of χ on which the point is on.
- Specify a smooth, monotonically increasing function that turns Δ into a probability.

- Define a *differentiable* loss function $\ell(y, y')$ such that the minimum average loss (risk) yields the optimal classifier.

The first point is a matter of geometry. The second is an arbitrary choice. However, choosing the last two ingredients well turns the training of a logistic-regression classifier into a *convex* optimization problem, and this is perhaps the main reason for the success of this type of linear classifier: If a convex function has a minimum, the minimum is global. In addition, smoothness and differentiability, as required above, ensure that this function can be minimized by some of the numerical optimization methods we studied in an earlier note.

2.1 Signed Distance from a Hyperplane

Let the equation of a hyperplane $\chi \in \mathbb{R}^d$ be

$$b + \mathbf{w}^T\mathbf{x} = 0 \tag{4}$$

where $\mathbf{w} \in \mathbb{R}^d$ is a nonzero vector and $b \leq 0$. If $b > 0$, the equation can be multiplied by -1 to make b negative.

If $\mathbf{w}$ were zero, the equation would not represent a hyperplane. Instead, it would represent all of $\mathbb{R}^d$ if $b = 0$, or the empty set if $b \neq 0$ (why?). This is why we require $\mathbf{w} \neq \mathbf{0}$.

If two points $\mathbf{a}_1$, $\mathbf{a}_2$ are on χ, then their difference $\mathbf{c} = \mathbf{a}_1 - \mathbf{a}_2$ is a vector parallel to χ. Conversely, any vector $\mathbf{c}$ parallel to χ can be written in this way: Just pick an arbitrary $\mathbf{a}_1 \in \chi$, and let $\mathbf{a}_2 = \mathbf{a}_1 - \mathbf{c}$. Since $\mathbf{c}$ is parallel to χ, the point $\mathbf{a}_2$ is on χ. In summary, $\mathbf{c}$ is parallel to χ if and only if it can be written in the form $\mathbf{c} = \mathbf{a}_1 - \mathbf{a}_2$ with $\mathbf{a}_1, \mathbf{a}_2 \in \chi$.

In addition, $\mathbf{a}_1$ and $\mathbf{a}_2$, being on χ, satisfy the equation that defines χ:

$$b + \mathbf{w}^T\mathbf{a}_1 = 0 \quad \text{and} \quad b + \mathbf{w}^T\mathbf{a}_2 = 0$$

and therefore, by subtracting these two equations from each other, we obtain

$$\mathbf{w}^T\mathbf{c} = 0 \, .$$

Therefore, *the vector* $\mathbf{w}$ *in equation 4 is perpendicular to the hyperplane* χ *it defines*, because it has zero inner product with any vector $\mathbf{c}$ that is parallel to χ. Let

$$\mathbf{n} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$$

be the unit vector along $\mathbf{w}$ with the same orientation as $\mathbf{w}$ (see Figure 4). The equation of χ can then be rewritten as follows:

$$\mathbf{n}^T\mathbf{x} = \beta \quad \text{where} \quad \beta = -\frac{b}{\|\mathbf{w}\|} \geq 0 \, . \tag{5}$$

The point $\mathbf{x}_0$ on χ that is closest to the origin of space is the intersection of χ with a line through χ and the origin, that is, with the line with parametric equation

$$\mathbf{x} = \alpha\mathbf{n} \quad \text{for} \quad \alpha \in \mathbb{R} \, ,$$

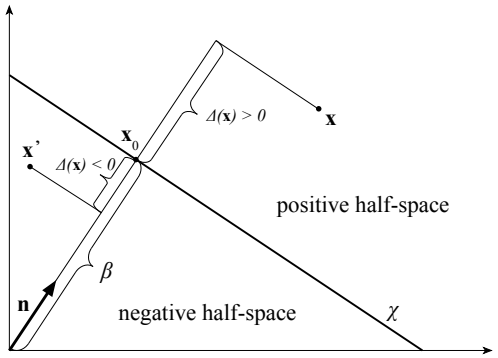


Figure 4: The geometry of a linear decision boundary in two dimensions $d = 2$. The two data points $\mathbf{x}$ and $\mathbf{x}'$ have respectively a positive and negative signed distance from the hyper-plane χ, as determined by the relative position of χ and the unit vector $\mathbf{n}$. This unit vector has the same direction and orientation as $\mathbf{w}$ (not shown, to reduce clutter), and which half-space a point $\mathbf{x}$ is in depends on the sign of $b+\mathbf{w}^T\mathbf{x}$. The point $\mathbf{x}_0$ is the point on χ closest to the origin, and $\Delta(\mathbf{x}_0) = 0$. These relationships hold in any dimension $d \geq 1$.

and since $\mathbf{n}$ has unit norm, α is the distance of any point $\mathbf{x}$ on this line from the origin. Replacing this expression for $\mathbf{x}$ into the equation 5 for χ yields

$$\alpha \mathbf{n}^T\mathbf{n} = \beta \quad \text{that is,} \quad \alpha = \beta \geq 0 \; .$$

We can conclude that if $\mathbf{w}$ is nonzero and $b \leq 0$ in equation 5, the distance of χ from the origin is β, as defined in the same expression, and the point on χ that is closest to the origin is

$$\mathbf{x}_0 = \beta \mathbf{n} \; ,$$

shown in Figure 4 as well. Let now $\mathbf{x}$ be a point in the half-space of X delimited by χ and for which

$$\mathbf{n}^T\mathbf{x} \geq \beta \; .$$

The left-hand side is nonnegative (because β is), and is the length of the projection of $\mathbf{x}$ onto $\mathbf{n}$. Therefore, the distance of $\mathbf{x}$ from χ is

$$\mathbf{n}^T\mathbf{x} - \beta \geq 0 \; .$$

If $\mathbf{x}$ is in the opposite half-space, we have

$$\mathbf{n}^T\mathbf{x} \leq \beta$$

and the distance of $\mathbf{x}$ from χ is

$$\beta - \mathbf{n}^T\mathbf{x} \; .$$

If we now allow for b to have an arbitrary sign, we can summarize this discussion as follows:

If $\mathbf{w}$ is nonzero in equation 4, the distance of χ from the origin is

$$\beta \stackrel{\text{def}}{=} \frac{|b|}{\|\mathbf{w}\|}$$

(a nonnegative number) and the quantity

$$\Delta(\mathbf{x}) \stackrel{\text{def}}{=} \frac{b + \mathbf{w}^T\mathbf{x}}{\|\mathbf{w}\|}$$

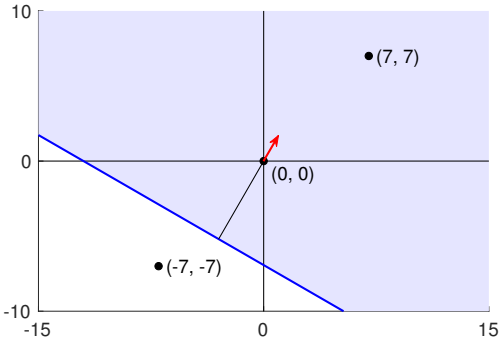
is the *signed distance* of point $\mathbf{x} \in X$ from hyperplane χ. Specifically, the distance of $\mathbf{x}$ from χ is $|\Delta(\mathbf{x})|$, and $\Delta(\mathbf{x})$ is nonnegative if and only if $\mathbf{x}$ is on the side of χ pointed to by $\mathbf{w}$. Let us call that side the *positive half-space* of χ.

It is important to realize that $\Delta(\mathbf{x})$ conveys not only the distance of $\mathbf{x}$ from the separating hyperplane χ (through its magnitude), but also which of the two half-spaces $\mathbf{x}$ is in (through the sign of $\Delta(\mathbf{x})$). As an example, Table 1 illustrates the four possible choices of sign for $\mathbf{w}$ and b in two dimensions ($d = 2$), when the data space X is the real plane and χ is therefore a line. The two cases at the top represent the same partition of X, but with different signs for the half-spaces. The same holds for the two cases at the bottom.

[To be continued in Part 2]

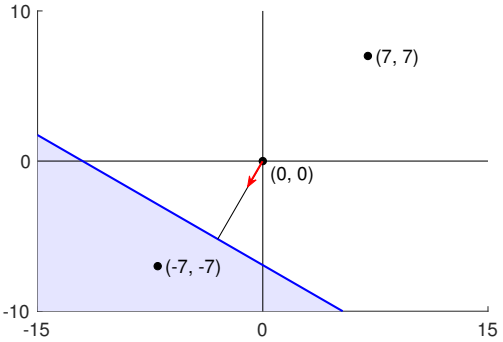
References

[1] D. De Cock. Ames, Iowa: Alternative to the Boston housing data as an end of semester regression project. *Journal of Statistics Education*, 19(3), 2011.



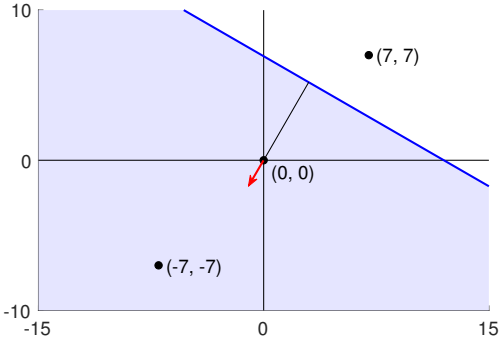
$$b = 12 \quad , \quad \mathbf{w}^T = [1,\ \sqrt{3}]$$

$$\begin{aligned}
\chi &: \quad 12 + x_1 + \sqrt{3}x_2 = 0 \\
\Delta\left([-7,\ -7]^T\right) &= \frac{12 - 1 \cdot 7 - \sqrt{3} \cdot 7}{2} \approx -3.56 \\
\Delta\left([0,\ 0]^T\right) &= \frac{12 + 1 \cdot 0 + \sqrt{3} \cdot 0}{2} = 6 \\
\Delta\left([7,\ 7]^T\right) &= \frac{12 + 1 \cdot 7 + \sqrt{3} \cdot 7}{2} \approx 15.56
\end{aligned}$$



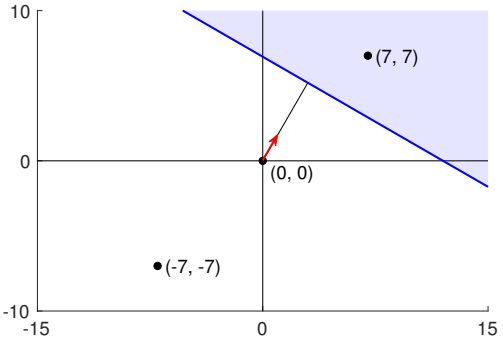
$$b = -12 \quad , \quad \mathbf{w}^T = [-1,\ -\sqrt{3}]$$

$$\begin{aligned}
\chi &: \quad -12 - x_1 - \sqrt{3}x_2 = 0 \\
\Delta\left([-7,\ -7]^T\right) &= \frac{-12 + 1 \cdot 7 + \sqrt{3} \cdot 7}{2} \approx 3.56 \\
\Delta\left([0,\ 0]^T\right) &= \frac{-12 - 1 \cdot 0 - \sqrt{3} \cdot 0}{2} = -6 \\
\Delta\left([7,\ 7]^T\right) &= \frac{-12 - 1 \cdot 7 - \sqrt{3} \cdot 7}{2} \approx -15.56
\end{aligned}$$



$$b = 12 \quad , \quad \mathbf{w}^T = [-1,\ -\sqrt{3}]$$

$$\begin{aligned}
\chi &: \quad 12 - x_1 - \sqrt{3}x_2 = 0 \\
\Delta\left([-7,\ -7]^T\right) &= \frac{12 + 1 \cdot 7 + \sqrt{3} \cdot 7}{2} \approx 15.56 \\
\Delta\left([0,\ 0]^T\right) &= \frac{12 - 1 \cdot 0 - \sqrt{3} \cdot 0}{2} = 6 \\
\Delta\left([7,\ 7]^T\right) &= \frac{12 - 1 \cdot 7 - \sqrt{3} \cdot 7}{2} \approx -3.56
\end{aligned}$$



$$b = -12 \quad , \quad \mathbf{w}^T = [1,\ \sqrt{3}]$$

$$\begin{aligned}
\chi &: \quad -12 + x_1 + \sqrt{3}x_2 = 0 \\
\Delta\left([-7,\ -7]^T\right) &= \frac{-12 - 1 \cdot 7 - \sqrt{3} \cdot 7}{2} \approx -15.56 \\
\Delta\left([0,\ 0]^T\right) &= \frac{-12 + 1 \cdot 0 + \sqrt{3} \cdot 0}{2} = -6 \\
\Delta\left([7,\ 7]^T\right) &= \frac{-12 + 1 \cdot 7 + \sqrt{3} \cdot 7}{2} \approx 3.56
\end{aligned}$$

Table 1: The four different sign choices $(\pm b, \pm \mathbf{w})$ for $b = 12$ and $\mathbf{w} = [1, \sqrt{3}]^T$. The (positive) distance β of the separating hyperplane χ (the blue line) from the origin is $\beta = |b|/\|\mathbf{w}\| = 12/2 = 6$ in all cases. The shaded area is the positive half-space, and the vector $\mathbf{w}$ (in red) points towards it, relative to χ.