Lecture #5

Decision boundaries for binary classification

Jason G. Fleischer, Ph.D.
Asst. Teaching Professor
Department of Cognitive Science, UC San Diego

jfleischer@ucsd.edu

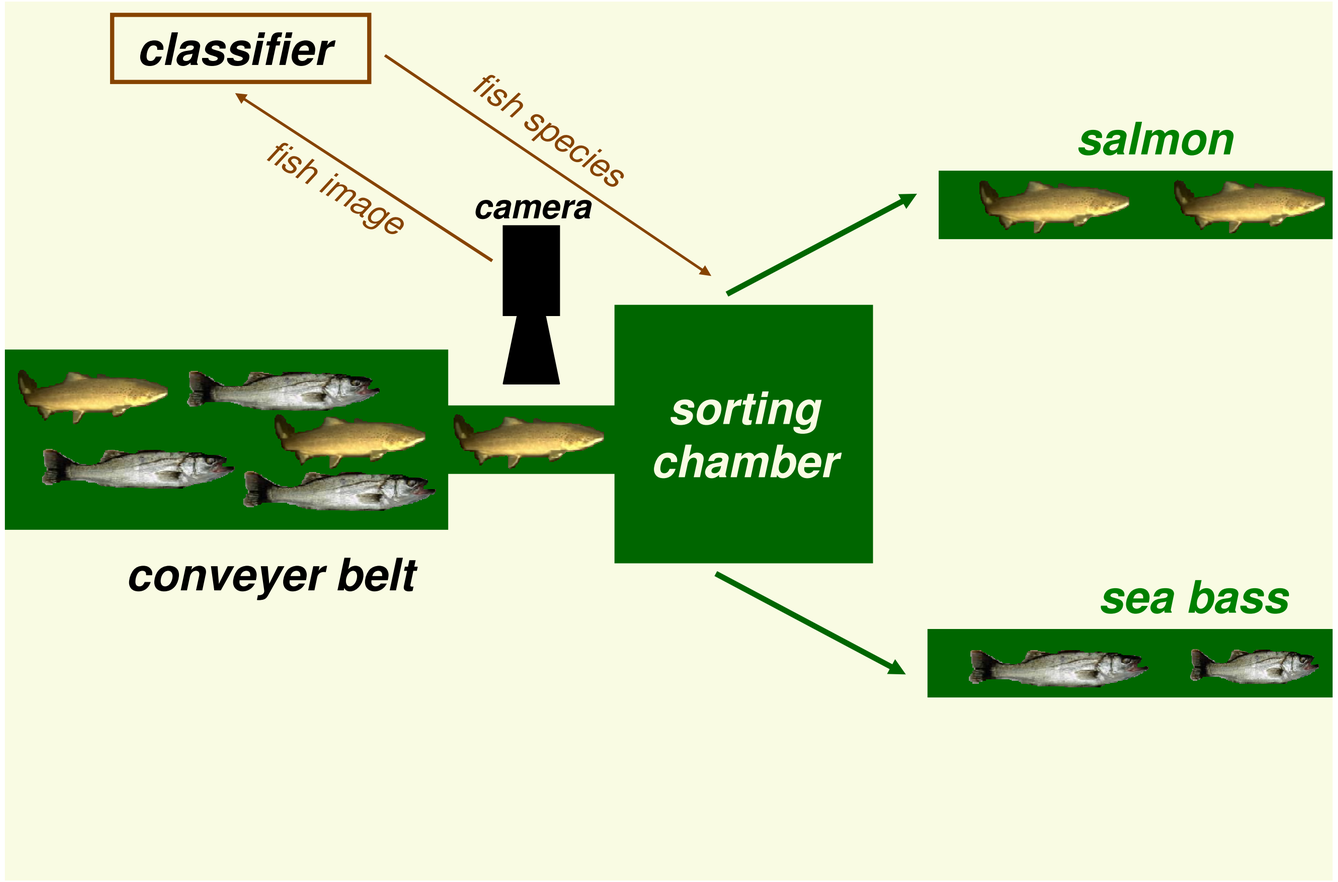
@jasongfleischer

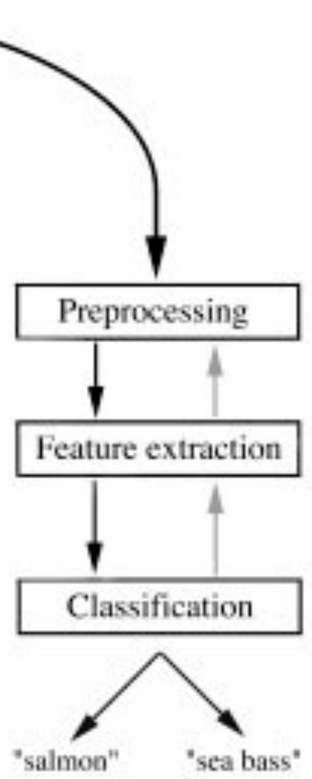
https://jgfleischer.com

Slides in this presentation are from material kindly provided by Zhuowen Tu and Olga Veksler

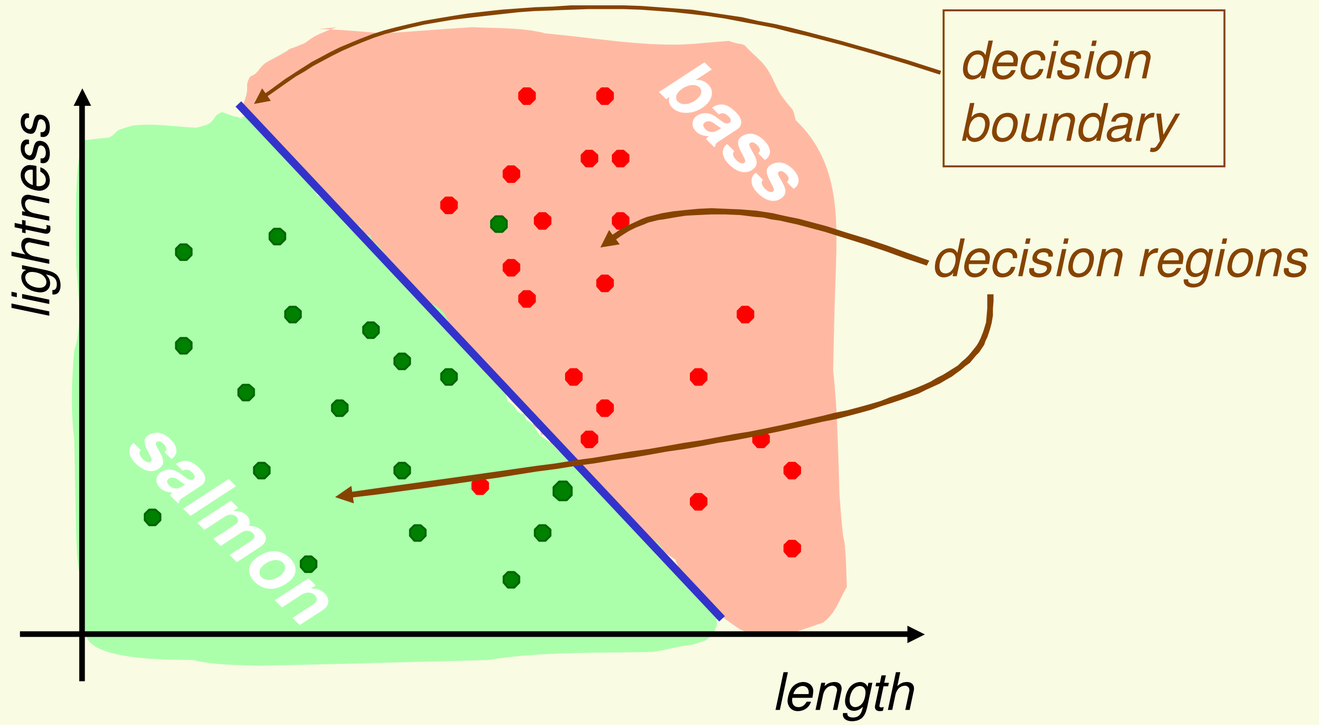
How are you feeling about the course?

An example


classifier
fish species
fish image
camera
salmon
sorting chamber
conveyer belt
sea bass


Preprocessing
Feature extraction
Classification
"salmon"
"sea bass"

- Use both *length* and *lightness* features
- Feature vector [*length*,*lightness*]



- Classification error 4%

Summary of the problem

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

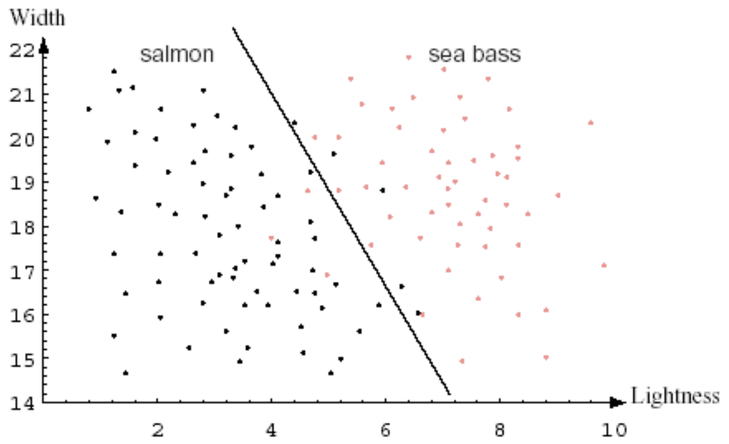
$\mathbf{x} = (x_1, ..., x_m), x_i \in \mathbb{R}, \quad \mathbf{x} \in \mathbb{R}^m$

$y \in \{0, 1\}$ $\quad y = 0$: negative

$y = 1$: positive

Classifier: $f(\mathbf{x}; W) \in \{0, 1\}$

Model parameter to be learned: W



Training error:

$$e_{training} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W))$$

Training errors

Supervised classifiers learn by minimizing the training error (either implicitly or explicitly)

Minimize $e_{training}$

The definition of training error varies depending on the types of classifier.

For binary classification typically its:

Minimize $\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}(y_i \neq f(\mathbf{x}_i; W))$

Whats the worst we can do at binary classification when there are an equal number of each class in the dataset?

Can you do worse if the datasets are not class-balanced? For example, 75% salmon and 25% seabass

Given a input set and a choice of classifier:

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\}$$

Classifier: $f(\mathbf{x}; W) \in \{0, 1\}$

$$e_{training} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W))$$

A. 0
B. 0.25

C. 0.5
D. 0.75
E. 1.0

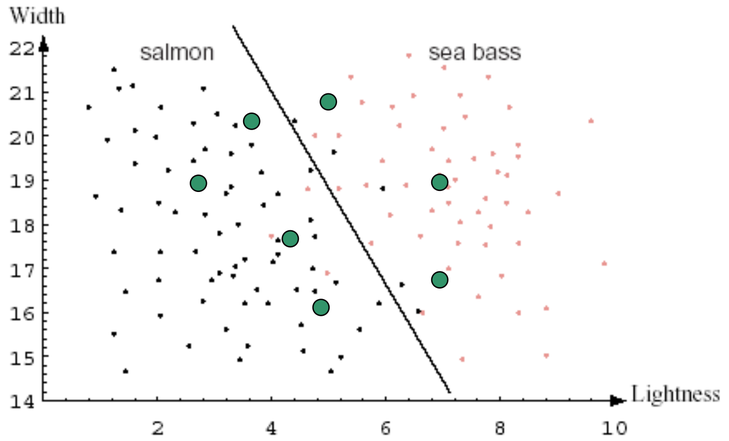
Why?

If $e_{training} > 0.5$

we simply define $f^{(reverse)}(\mathbf{x}_i; W) = 1 - f(\mathbf{x}_i; W)$

Testing

We use the training set to train a classifier $f(\mathbf{x}; W)$.



Given a set of testing data, $S_{testing}$, we make the prediction of each input and evaluate the algorithm.

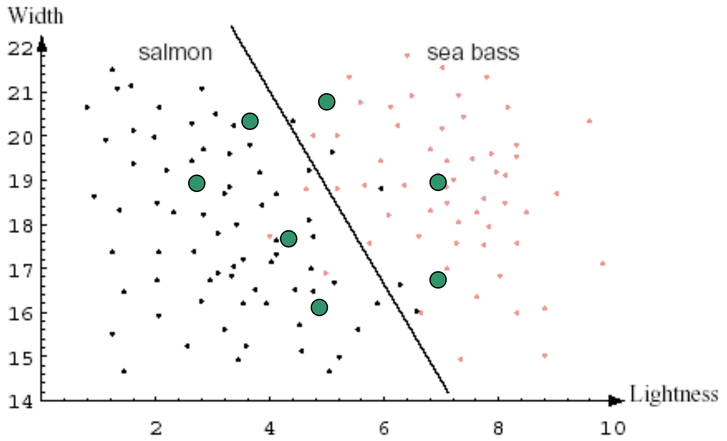
$S_{testing} = \{(\mathbf{x}_i, y_i), i = 1..q\}$.

For each $\mathbf{x}_i$ we want to predict its y_i.

y_i is given to evaluate the quality of a classifier and is not given in reality.

Testing

We use the training set to train a classifier $f(\mathbf{x}; W)$.



$S_{testing} = \{(\mathbf{x}_i, y_i), i = 1..q\}$

$$e_{testing} = \frac{1}{q} \sum_{i=1}^{q} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W))$$

Training Error and Testing Error

$$e_{training} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W)) \qquad e_{training} \in [0, 1]$$

$$e_{testing} = \frac{1}{q}\sum_{i=1}^{q} \mathbf{1}(y_i \neq f(\mathbf{x}_i; W)) \qquad e_{testing} \in [0, 1]$$

Both training and testing errors are between 0 and 1

(because we don't know if classes are equally likely in the dataset)

In nearly all situations for any classifier,

$$e_{testing} \geq e_{training}$$

The difference between the testing and training set error is called generalization error.

When $e_{testing} >> e_{training}$, we call it "overfitting"

Training Error and Testing Error

$$e_{testing} \geq e_{training}$$

Why?
When?

As long as both datasets have the same distribution of values, this must be true “on average” over train/test cycles.

Any given train/test cycle COULD yield a random result in contrast to this rule.

If the algorithm has low enough bias to fit the noise of the training data, then it will overfit that particular noise pattern.

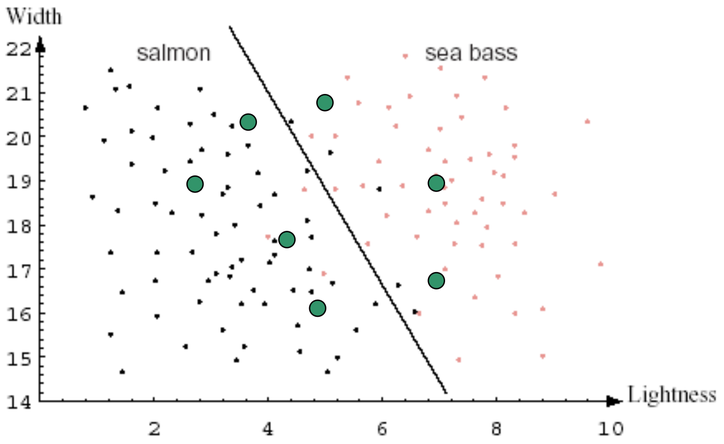
What if
training set complexity >>
test set complexity?

What’s your intuition about what happens to the relationship between testing and training errors in the limit as both set sizes approach infinity?

Testing error

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\} \qquad S_{testing} = \{(\mathbf{x}_i, y_i), i = 1..q\}$$

$$e_{testing} = e_{training} + generalization(f)$$



$$e_{testing} = \frac{1}{q}\sum_{i=1}^{q} \mathbf{1}(y_i \neq f(\mathbf{x}_i))$$

$$e_{testing} = 0.5?$$

1. $e_{testing} = 0.5 + 0.0$
2. $e_{testing} = 0.0 + 0.5$

Question: What are the two extreme cases leading to the same testing error=0.5 (Assume we have the same number of positives and negatives.)

Testing error

$$S_{training} = \{(\mathbf{x}_i, y_i), i = 1..n\} \qquad S_{testing} = \{(\mathbf{x}_i, y_i), i = 1..q\}$$

$$e_{testing} = e_{training} + generalization(f)$$

Case 1:

1. $e_{testing} = 0.5 + 0.0$

- We make a completely random guess even after training (didn't learn anything and attained an error of 0.5).
- A random guess is however highly generalizable, which doesn't incur any generalization error.

Case 2:

2. $e_{testing} = 0.0 + 0.5$

- We make perfect classifications on the training data (memorizing the labels for every training sample).
- Merely memorizing the training samples with their corresponding classification labels is however highly non-generalizable, incurring the largest generalization error (no identical training sample will appear in testing).

Both are extreme cases that lead to trivial ML models.

Testing error

$$e_{testing} = e_{training} + generalization(f)$$

Case 1:

1. $e_{testing} = 0.5 + 0.0$

Case 2:

2. $e_{testing} = 0.0 + 0.5$

Both are extreme cases that lead to trivial models that we should avoid.

Ideally: $e_{testing} = 0.0 + 0.0$

A classification model that is perfect after training and makes no error in testing.

In practice: $e_{testing} = 0.05 + 0.1$

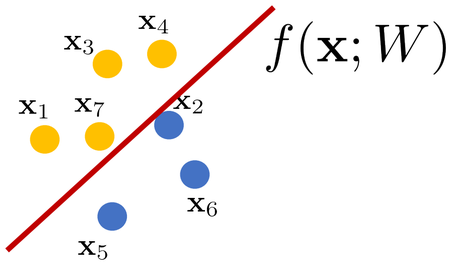
A classification model that does well after training and generalizes reasonably well in testing.

The key elements

Input: $\mathbf{x} = (x_1, x_2, ...)$

Label: $y \in 0, 1$

Model parameter: W



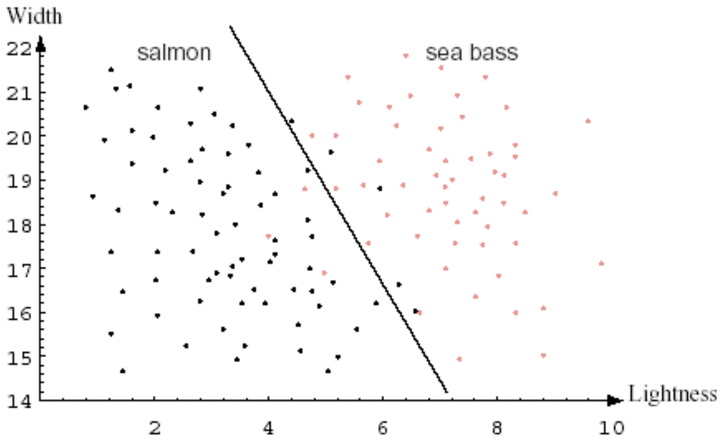
Decision boundary

Let **x** be the input vector (observation) and y be its label:

Often, we are given a set of training data

$$S = \{(\mathbf{x}_i, y_i), i = 1..n\} \quad \mathbf{x} = (x_1, ..., x_m), x_i \in \mathcal{R}, \quad \mathbf{x} \in \mathcal{R}^m$$

A classifier f(**x**):



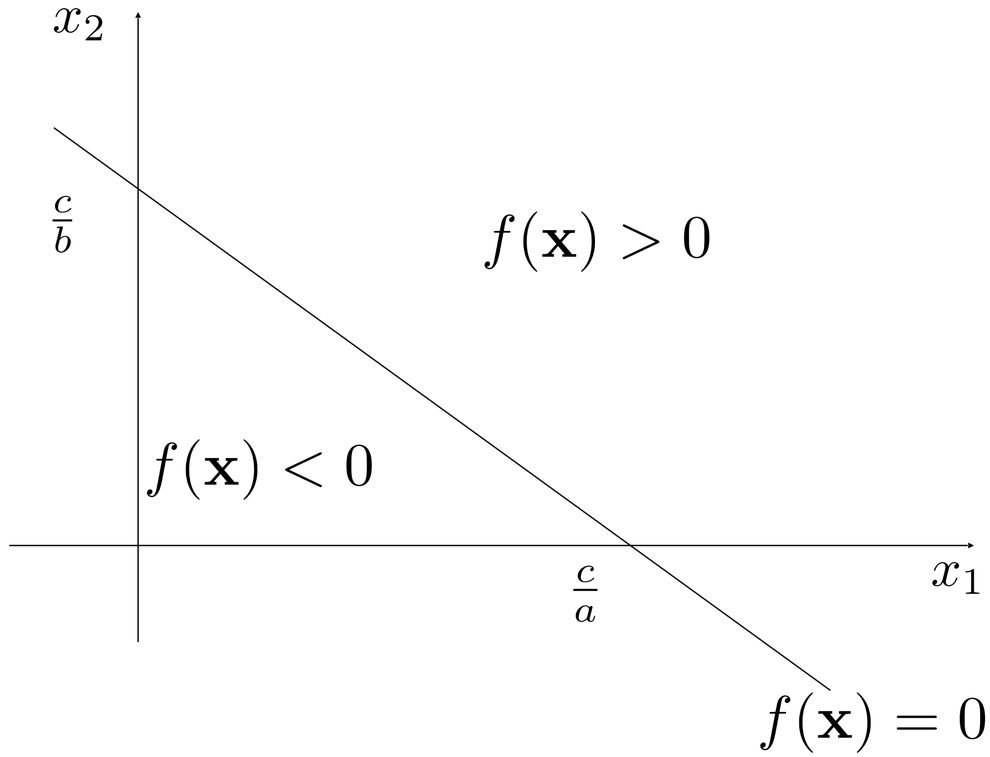
Decision boundary:
$\{\mathbf{x}_i, f(\mathbf{x}_i) = 0\}$

Decision boundary

Let **x** be the input vector (observation) and y be its label:

Decision boundary:
$\{\mathbf{x}_i, f(\mathbf{x}_i) = 0\}$

$$f(\mathbf{x}) = a \times x_1 + b \times x_2 - c$$



Decision boundary

"Decision boundary" is one of the most important concepts in machine learning.

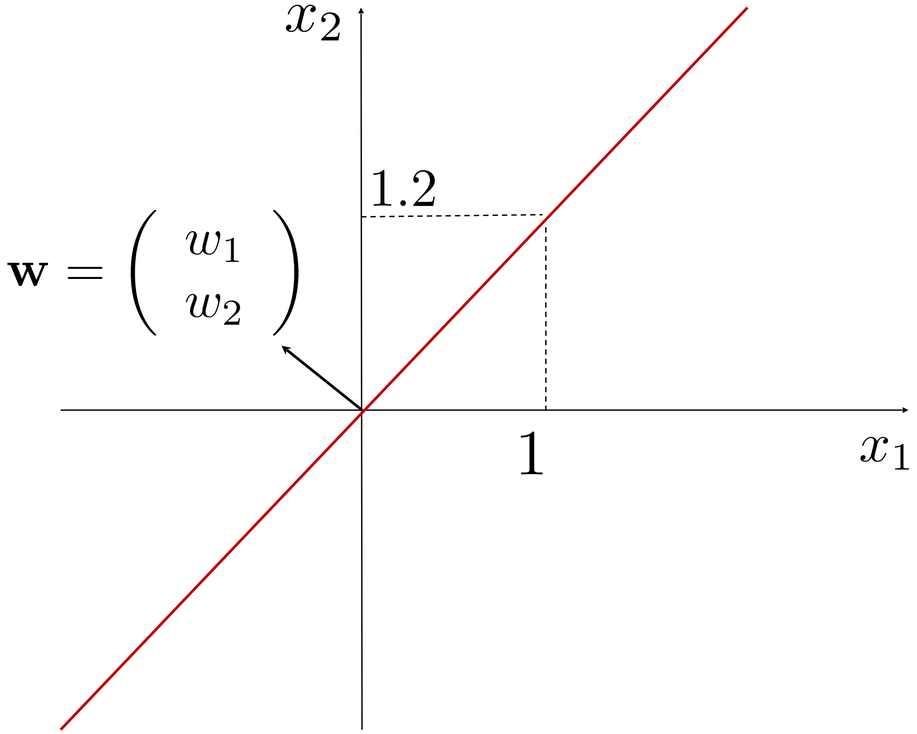
The decision boundary of a binary classifier refers to the set of data samples that are "on the fence" between making the decision being positive or negative: that is being 50%-50% for classification.

Decision boundary is a characteristic of your learned model so it varies due to the choice of your classification model and how it has been trained.

Typically, decision boundary: $\{\mathbf{x}_i, f(\mathbf{x}_i) = 0\}$

Line and vector

an example:



$\mathbf{w}$ is the normal direction of the line

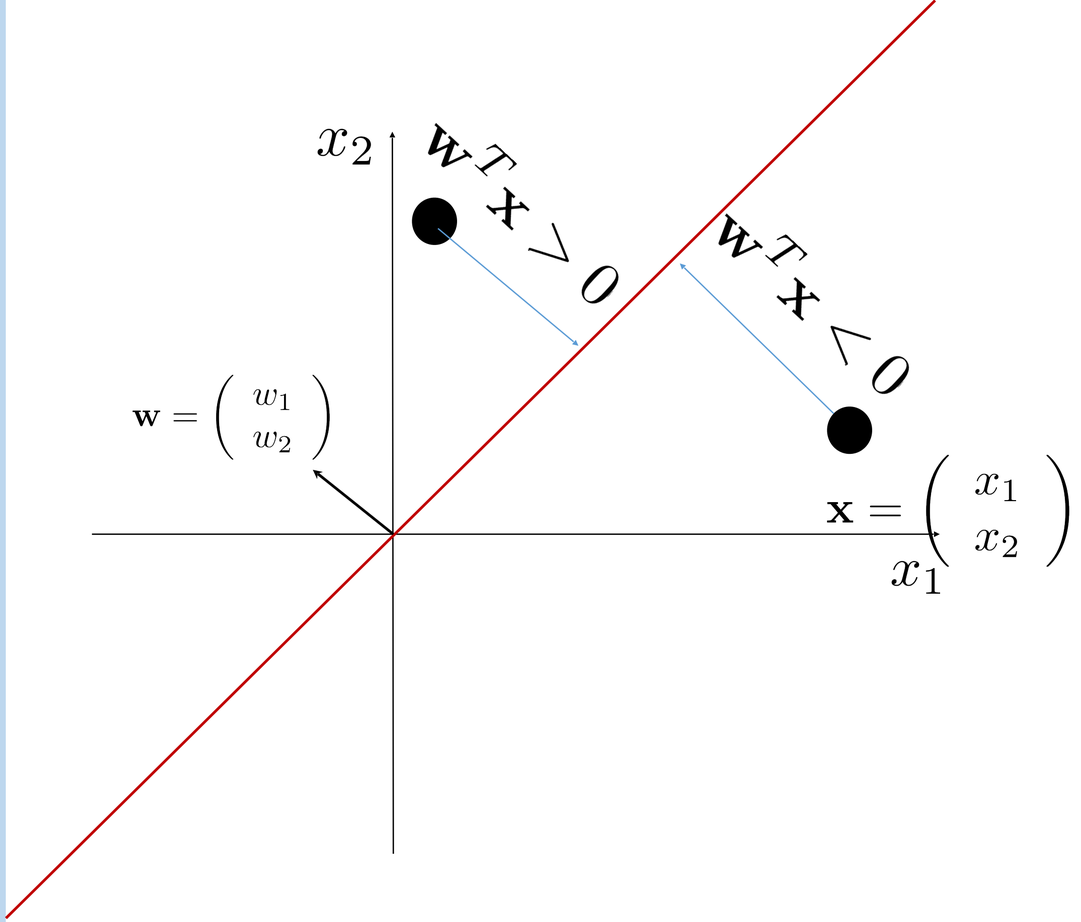
Often: $||\mathbf{w}||_2 = 1$: a unit vector

$$x_2 = 1.2 \times x_1$$

$$\mathbf{w} = \begin{pmatrix} -\frac{1.2}{\sqrt{2.44}} \\ \frac{1}{\sqrt{2.44}} \end{pmatrix}$$

Distance to the decision boundary

(arguably the most important concept in machine learning)



The distance (signed) of any point $\mathbf{x}$ to the line is:

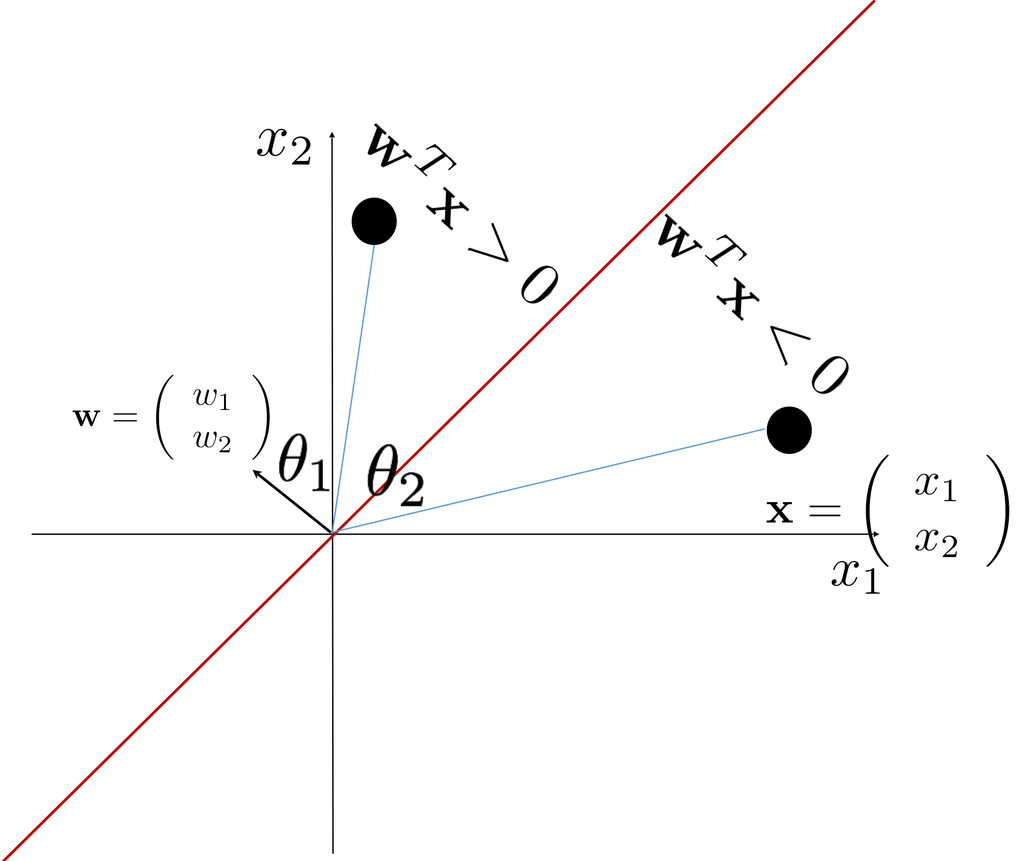
$$\mathbf{w}^T\mathbf{x} \equiv < \mathbf{w}, \mathbf{x} >$$

$\mathbf{w}^T\mathbf{x} > 0$: above the line

$\mathbf{w}^T\mathbf{x} < 0$: below the line

Distance to the decision boundary

(arguably the most important concept in machine learning)



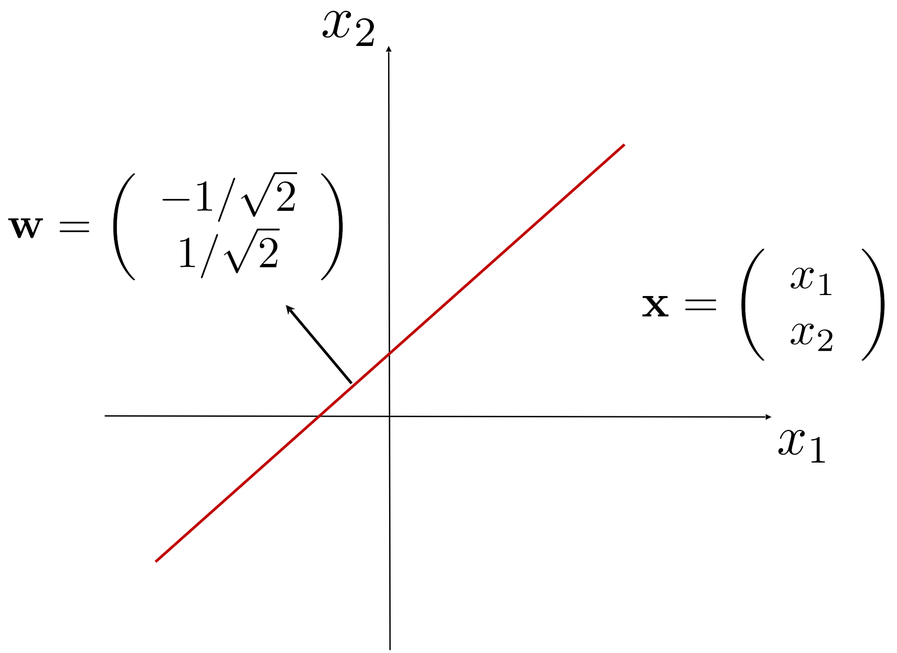
Why below the line is < 0
while above the line is > 0?

$$\mathbf{w}^T\mathbf{x} \equiv < \mathbf{w}, \mathbf{x} >$$

$$\mathbf{w}^T\mathbf{x} = \|\mathbf{w}\| \cdot \|\mathbf{x}\| \cdot \cos\theta$$

It depends on the angle
formed by w and x

Decision boundary?



Can the line (in red) be the decision boundary of the classifier below

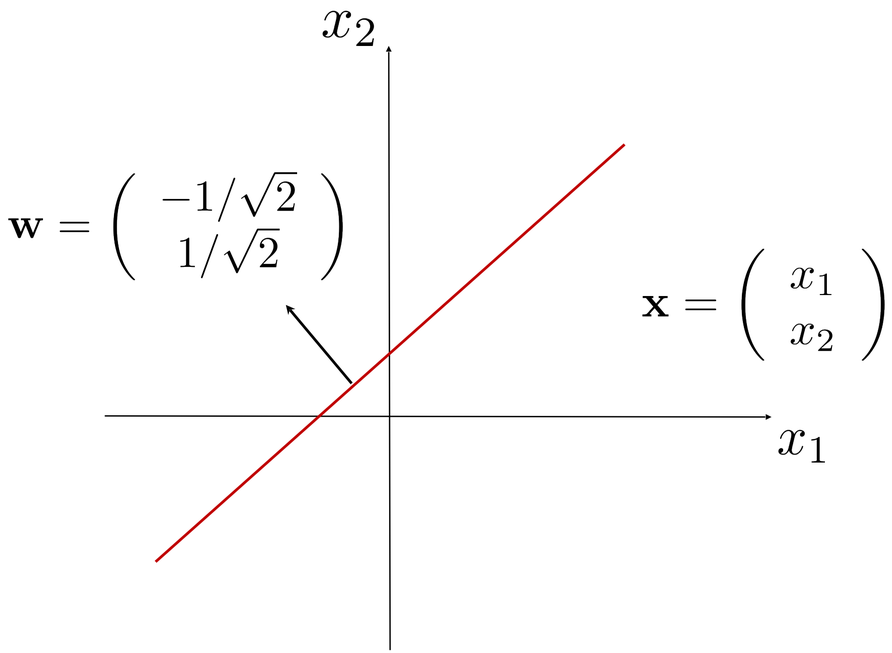
$$\mathbf{w} = \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \qquad y = \begin{cases} +1 & if\ \mathbf{w}^T\mathbf{x} > 0 \\ -1 & if\ \mathbf{w}^T\mathbf{x} < 0 \end{cases} ?$$

A. Yes

B. No

C. It depends

Decision boundary?



Can the line (in red) be the decision boundary of a classifier

$\mathbf{w} = \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$ $\quad y = \begin{cases} +1 & if\ \mathbf{w}^T\mathbf{x} > 0 \\ -1 & if\ \mathbf{w}^T\mathbf{x} < 0 \end{cases}$?

A. Yes

B. No

C. It depends

Decision boundary

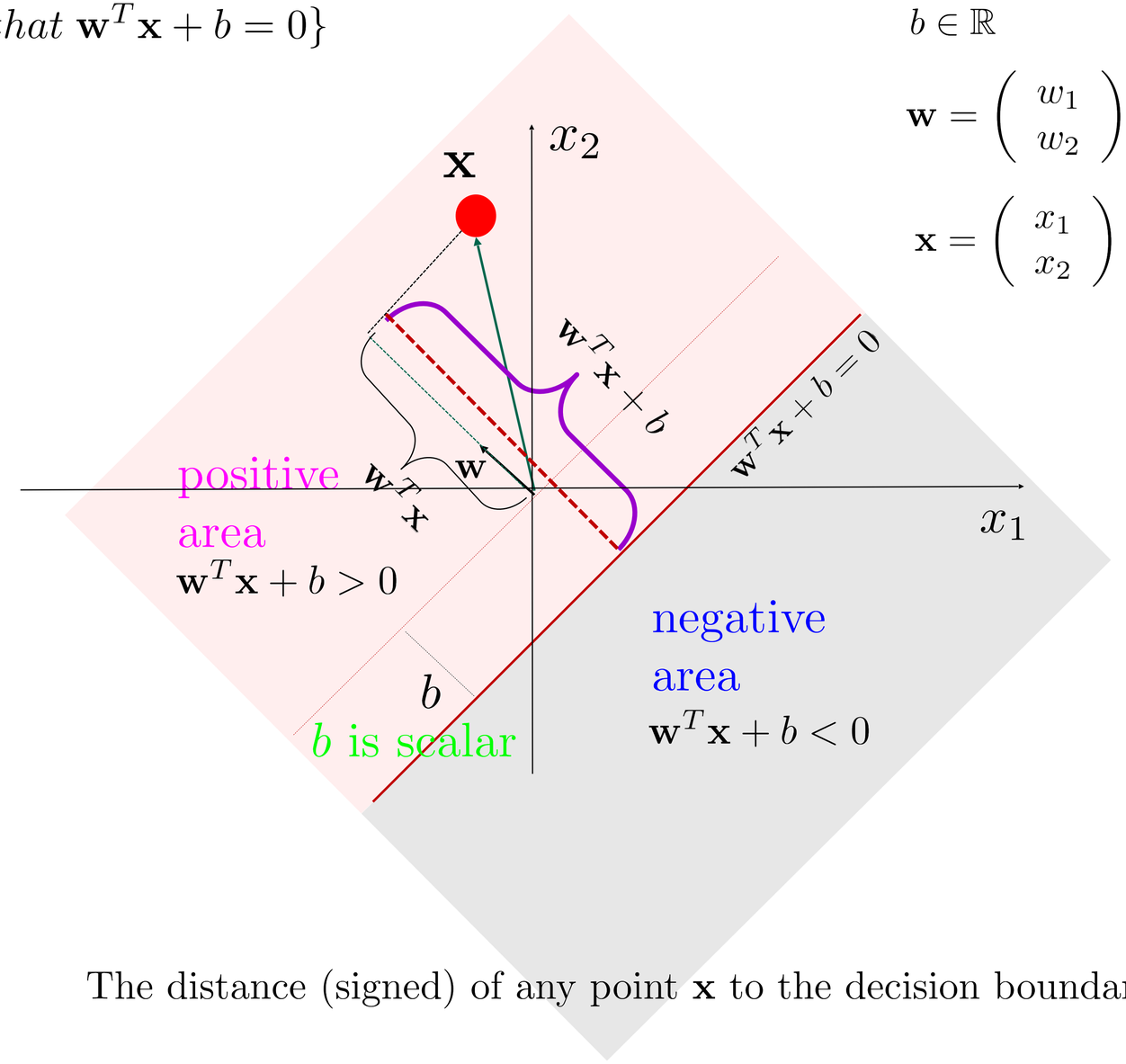
Decision boudary:

$\{\mathbf{x}; \forall \mathbf{x}\ such\ that\ \mathbf{w}^T\mathbf{x} + b = 0\}$

$b \in \mathbb{R}$

$$\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



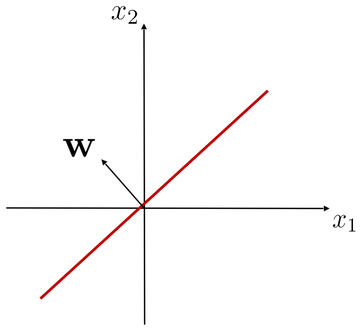
The distance (signed) of any point $\mathbf{x}$ to the decision boundary is: $\mathbf{w}^T\mathbf{x} + b$

Decision boundary

When b=0:

$\{\mathbf{x}; \forall \mathbf{x}\ such\ that\ \mathbf{w}^T\mathbf{x} = 0\}$

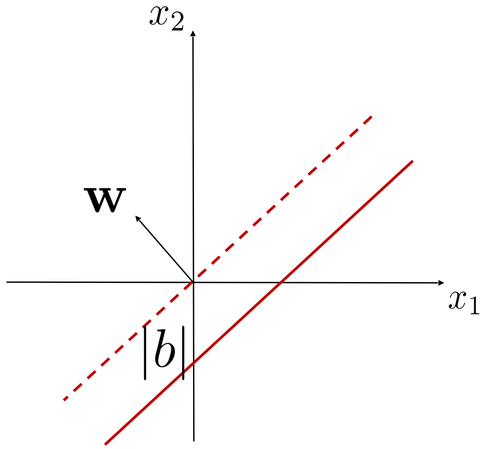
The decision boundary always goes through the origin.



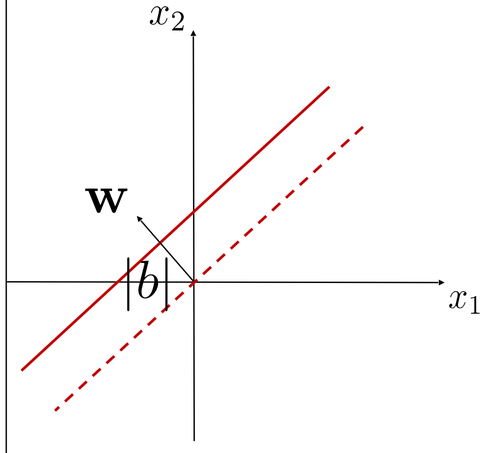
Decision boudary:

$\{\mathbf{x}; \forall \mathbf{x}\ such\ that\ \mathbf{w}^T\mathbf{x} + b = 0\}$

When b>0 the decision boundary is moved along the opposite direction of w.



When b<0 the decision boundary is moved along the same direction of w.



Some typical examples

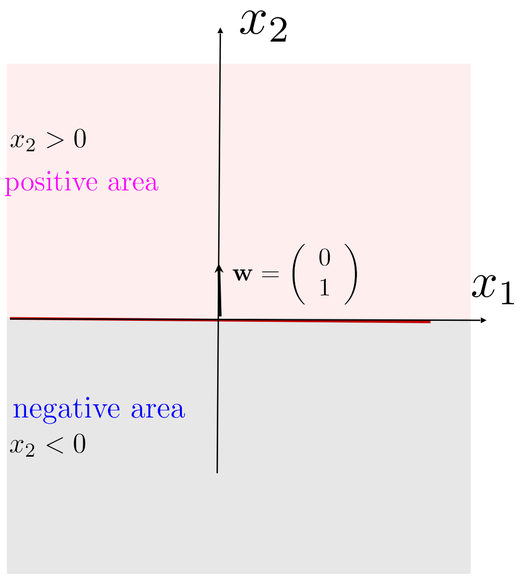
Assuming $\mathbf{w}$ being normalized: $||\mathbf{w}||_2 = \sqrt{w_1^2 + w_2^2} = 1$.

$$\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \qquad b \in \mathbb{R} \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$0 \times x_1 + 1 \times x_2 = 0$

$\Downarrow$

$x_2 = 0$



$$\mathbf{w} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$b = 0$

Decision boundary:

$0 \times x_1 + 1 \times x_2 = 0$

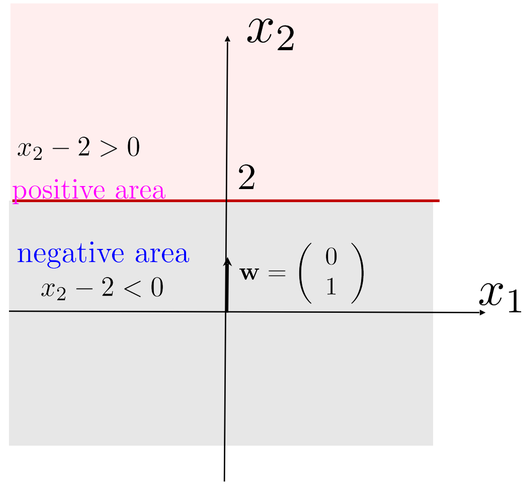
$\Downarrow$

$x_2 = 0$

$0 \times x_1 + 1 \times x_2 - 2 = 0$

$\Downarrow$

$x_2 - 2 = 0$



$$\mathbf{w} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$b = -2$

Decision boundary:

$0 \times x_1 + 1 \times x_2 - 2 = 0$

$\Downarrow$

$x_2 - 2 = 0$

Decision boundary

The decision boundary of a binary classifier refers to the set of data samples that are “on the fence” between making the decision being positive or negative: that is being 50%-50% for classification.

For a linear model, the decision boundary is a line/hyper-plane, depending upon the dimension of the data.

The model parameter **w** is along the normal direction that is orthogonal to the decision boundary.

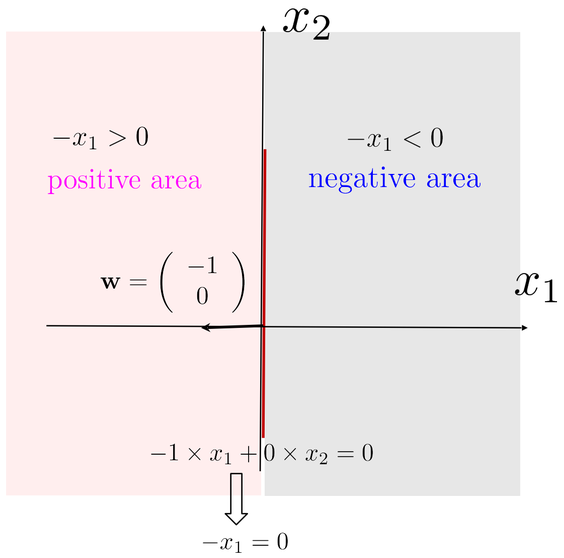
There is often a bias terms, b (scalar), refers to as the translation (shift) of the decision boundary.

Note that **w** itself is NOT the decision boundary.

Some typical examples

Assuming $\mathbf{w}$ being normalized: $||\mathbf{w}||_2 = \sqrt{w_1^2 + w_2^2} = 1$.

$$\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \qquad b \in \mathbb{R} \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



$$\mathbf{w} = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$$

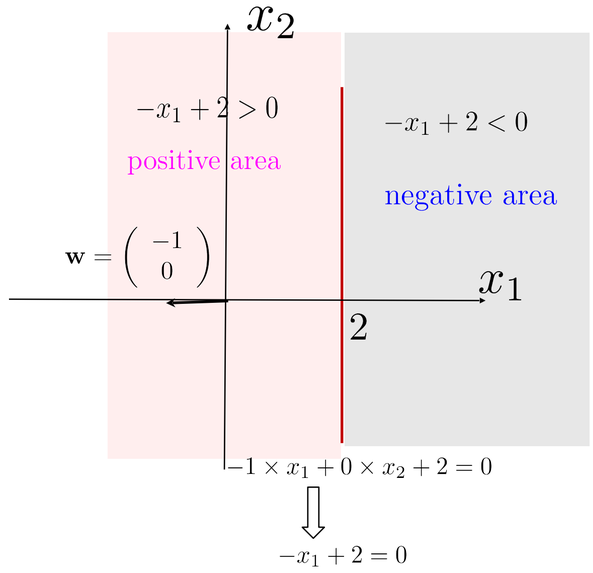
$b = 0$

Decision boundary:

$$-1 \times x_1 + 0 \times x_2 = 0$$

$$\Downarrow$$

$$-x_1 = 0$$



$$\mathbf{w} = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$$

$b = 2$

Decision boundary:

$$-1 \times x_1 + 0 \times x_2 + 2 = 0$$

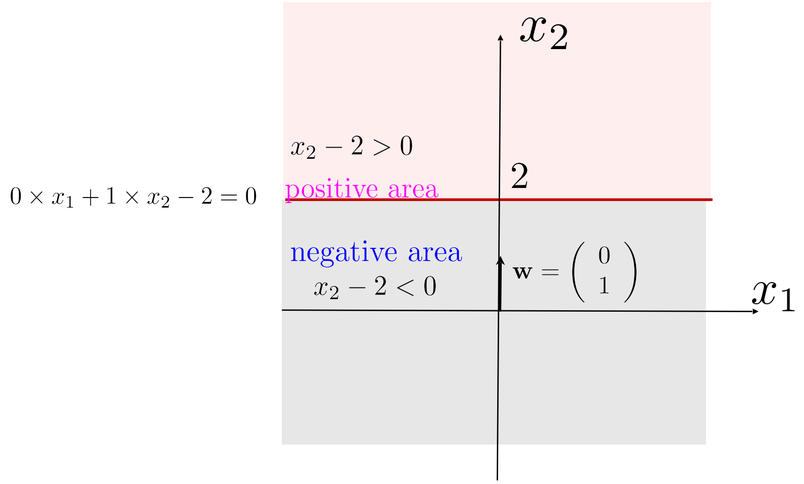
$$\Downarrow$$

$$-x_1 + 2 = 0$$

Pay attention to the bias term b

When b is negative, decision boundary moves in the direction of w

When b is positive, boundary moves opposite w

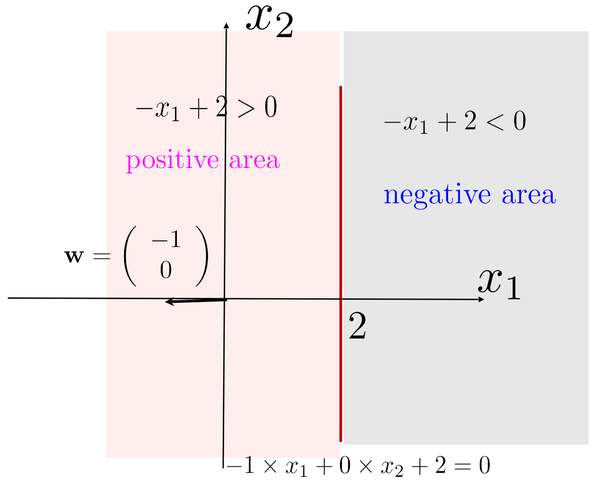


$\mathbf{w} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

Decision boundary:

$0 \times x_1 + 1 \times x_2 - 2 = 0$

$b = -2$



$\mathbf{w} = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$

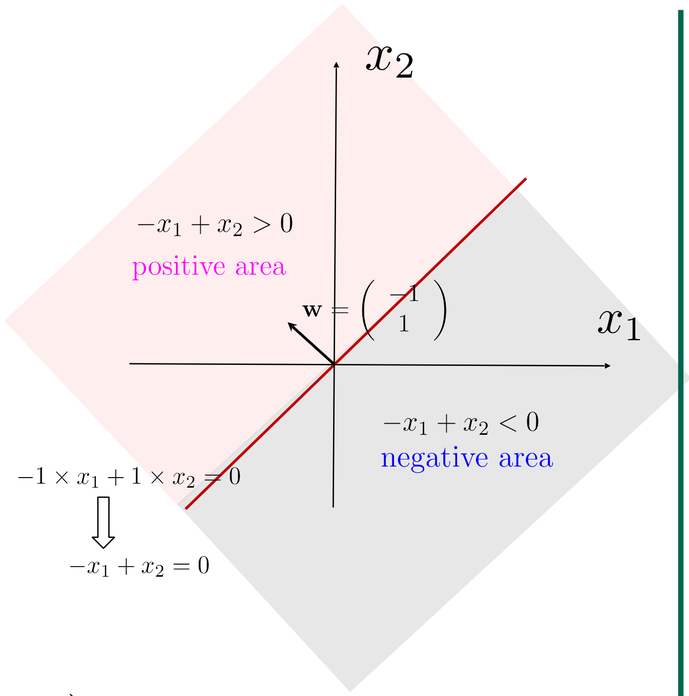
Decision boundary:

$-1 \times x_1 + 0 \times x_2 + 2 = 0$

$b = 2$

Some typical examples

$$\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \qquad b \in \mathbb{R} \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



$\mathbf{w} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$

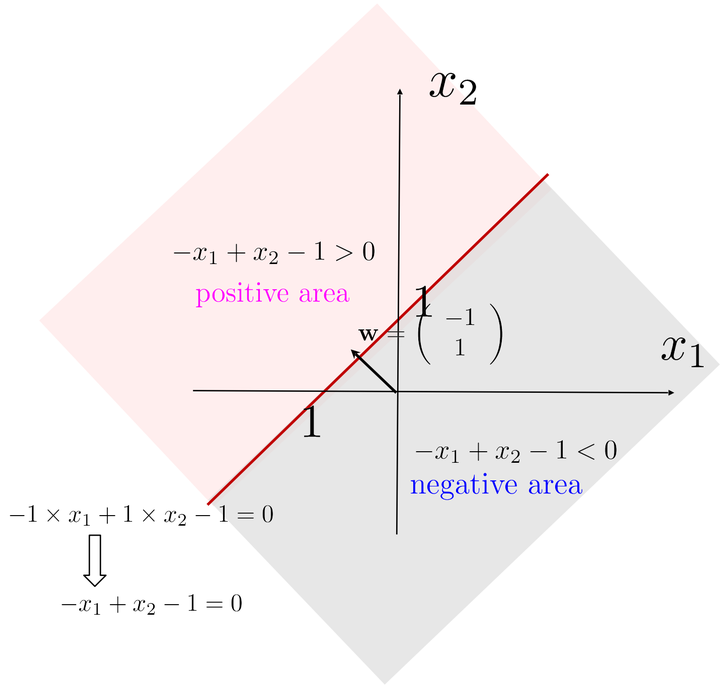
$b = 0$

Decision boundary:

$-1 \times x_1 + 1 \times x_2 = 0$

$\Downarrow$

$-x_1 + x_2 = 0$



$\mathbf{w} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$

$b = -1$

Decision boundary:

$-1 \times x_1 + 1 \times x_2 - 1 = 0$

$\Downarrow$

$-x_1 + x_2 - 1 = 0$

Take home message

Any data sample (point) lying on the decision boundary receives a classification decision that is equally positive and negative.

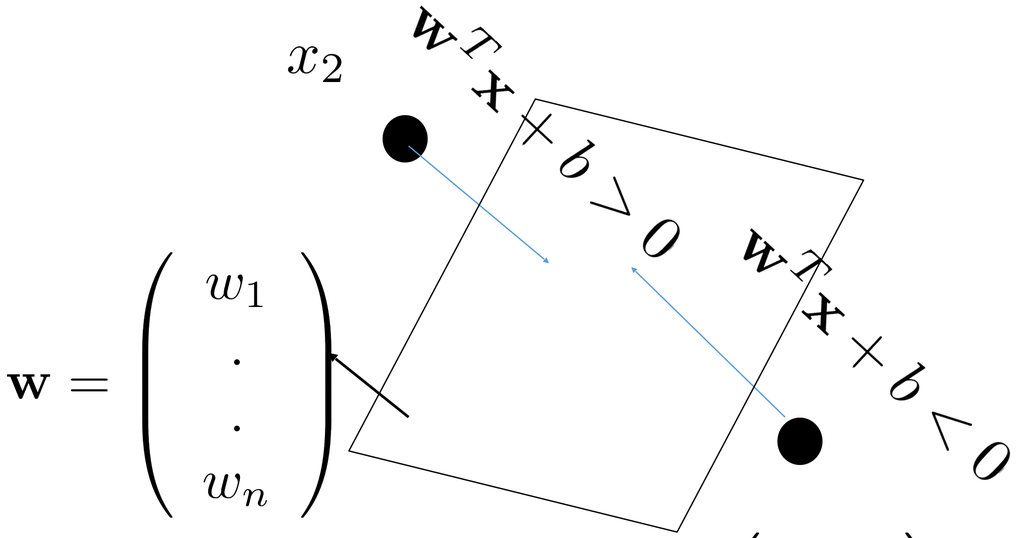
The decision boundary of a linear classifier is a hyper-plane.

The model parameter **w** is along the normal direction of the decision boundary, pointing to the positive samples.

The bias terms, b (scalar), refers to as the translation (shift) of the decision boundary.

Distance to the decision boundary

(arguably the most important concept in machine learning)



$$\mathbf{w} = \begin{pmatrix} w_1 \\ . \\ . \\ w_n \end{pmatrix}$$

$$\mathbf{w} \in \mathbb{R}^n$$

$$b \in \mathbb{R}$$

$$\mathbf{x} = \begin{pmatrix} x_1 \\ . \\ . \\ x_n \end{pmatrix}$$

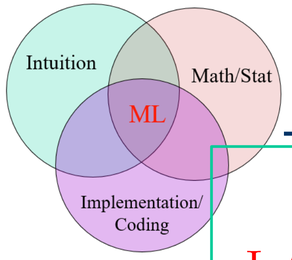
$$\mathbf{x} \in \mathbb{R}^n$$

The distance (signed) of any point $\mathbf{x}$ to the hyper-plane is:

$$\mathbf{w}^T\mathbf{x} + b \equiv < \mathbf{w}, \mathbf{x} > + b \equiv \mathbf{w} \cdot \mathbf{x} + b$$

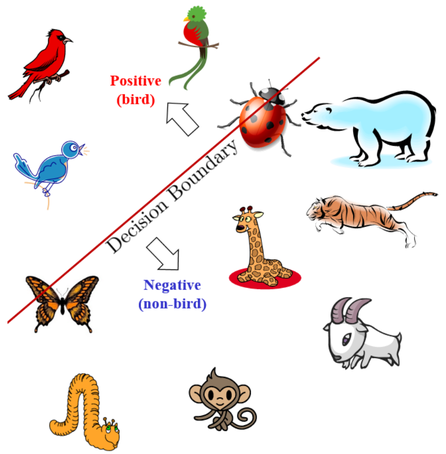
$\mathbf{w}^T\mathbf{x} + b > 0$: above the hyper-plane

$\mathbf{w}^T\mathbf{x} + b < 0$: below the hyper-plane



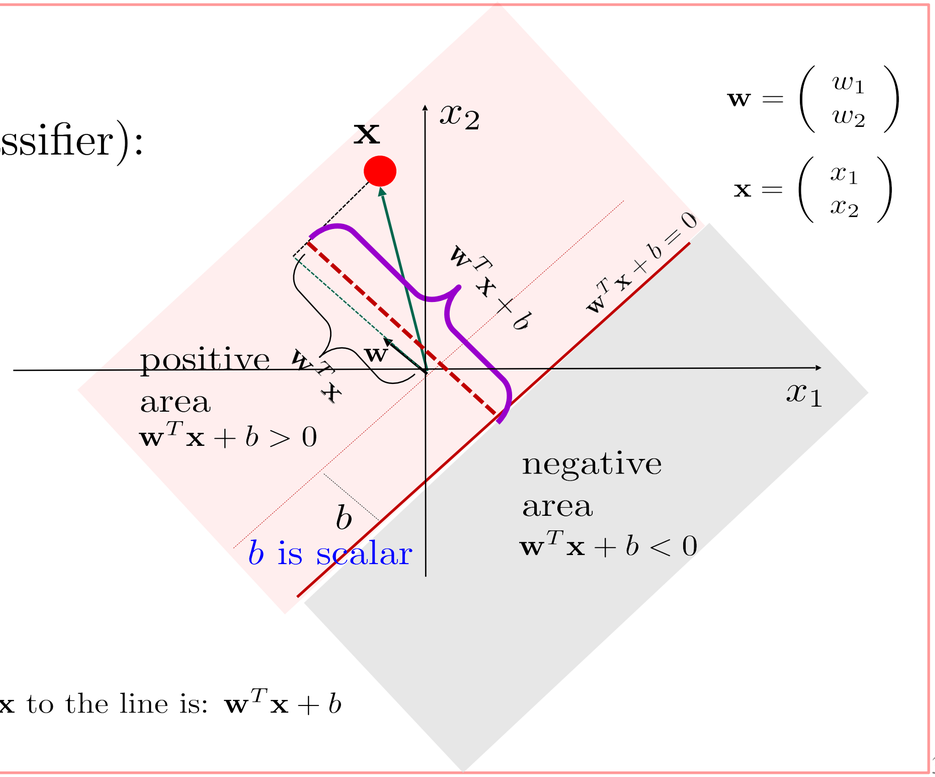
Recap: Decision Boundary

Intuition: Decision boundary of a discriminative classifier is a set that consists of possible samples that are on the border (typically $50\% - 50\%$) of the separation between the positive and negative areas (for two-class classification).



Math:

Decision boudary (for a linear classifier):
$\{\mathbf{x}; \forall \mathbf{x}\ \textit{such that}\ \mathbf{w}^T\mathbf{x} + b = 0\}$



The distance (signed) of any point $\mathbf{x}$ to the line is: $\mathbf{w}^T\mathbf{x} + b$