

 kdenisk on March 24, 2017 at 12:39

# Geometry of machine learning. Separating hyperplanes or what is the geometric meaning of a linear combination?

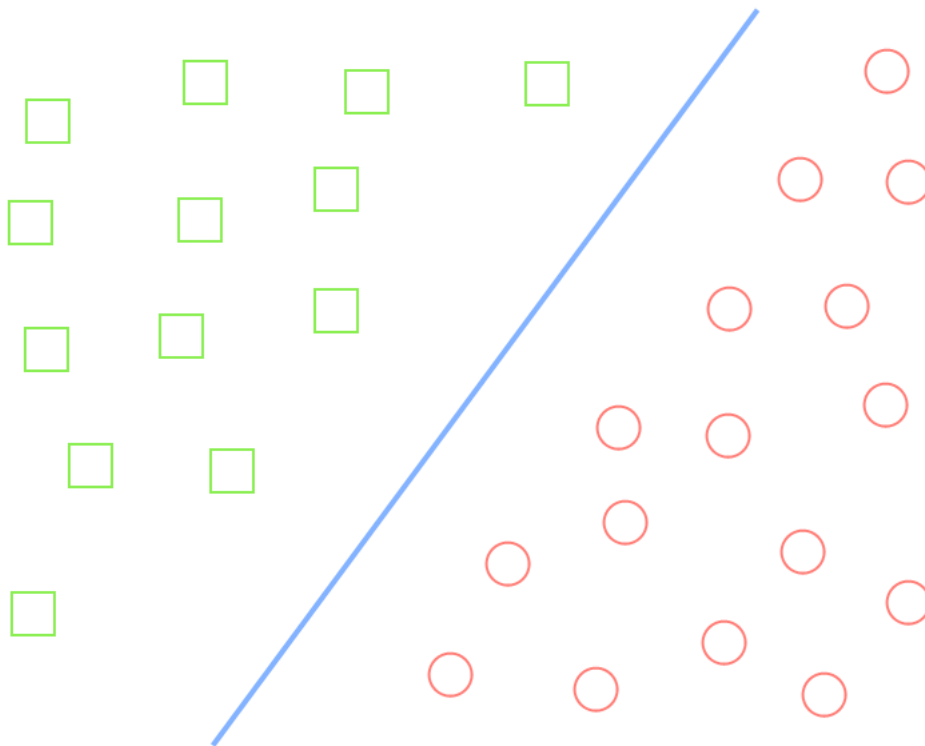
Machine learning , Algorithms

In many algorithms of machine learning, including in neural networks, we constantly have to deal with a weighted sum or, otherwise, a linear combination of input vector components. And what is the meaning of the obtained scalar value?

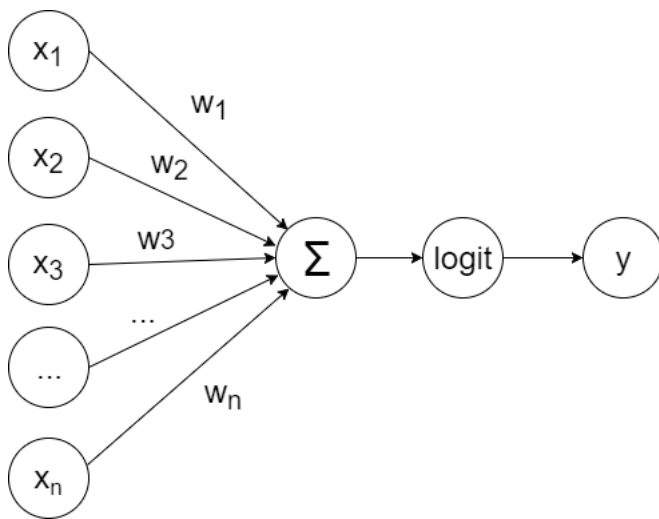
In this article, we will try to answer this question with examples, formulas, as well as many illustrations and code in Python, so that you can easily reproduce everything and put your own experiments.

## Model Example

To ensure that the theory does not break away from real cases, let's take the binary classification problem as an example. There is a data set:  $m$  samples, each sample is an  $n$ -dimensional point. For each sample, we know which class it belongs to (green or red). It is also known that the data linearly separable, i.e. There exists an  $n$ -dimensional hyperplane such that the green points lie on one side of it, and the red points lie on the other



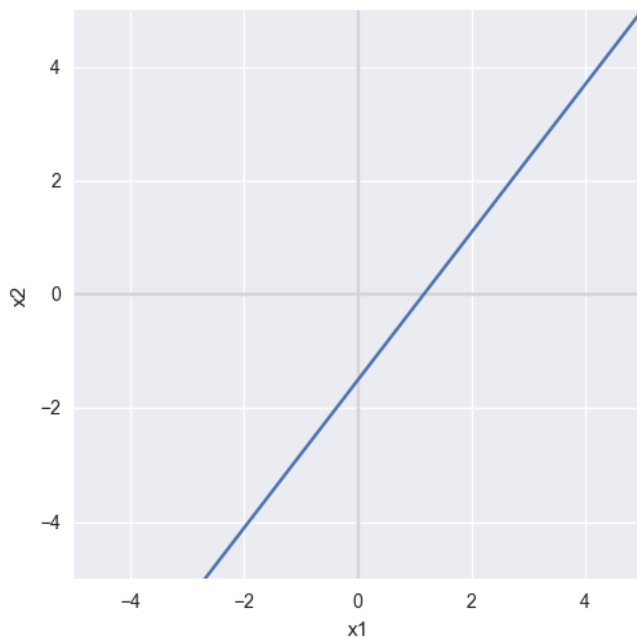
The solution to the problem of finding such a hyperplane can be approached in different ways, for example, by using logistic regression, the method of reference vectors with a linear kernel (linear SVM), or by taking the simplest neural network:



At the end of the article, we write the perceptron training mechanism from scratch and solve the problem of binary classifications with their own  $\mathbf{w}$  using the knowledge gained.

## From a straight line to a hyperplane

Consider the detailed math for the straight line. For the general case of a hyperplane in an  $n$ -dimensional space everything will be exactly the same with the correction for the number of components in the vectors.



A straight line in the plane is given by three numbers  $-(\mathbf{w}_1, \mathbf{w}_2, b)$ :

$$\mathbf{w}_1 x_1 + \mathbf{w}_2 x_2 + b = 0$$

or:

$$\sum_{i=1}^2 \mathbf{w}_i x_i + b = 0$$

or:

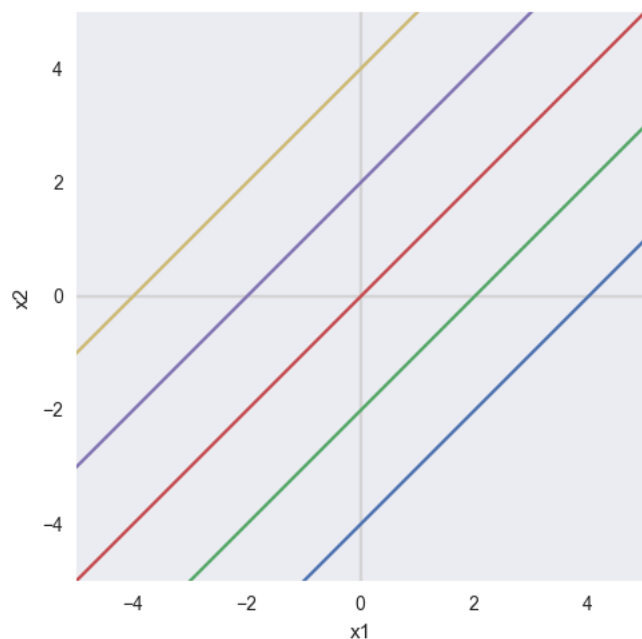
$$\mathbf{w}^T \mathbf{x} + b = 0$$

The first two coefficients  $\mathbf{w}_1, \mathbf{w}_2$  define the whole family of straight lines passing through the point  $(0, 0)$ . The relationship between  $\mathbf{w}_1$  and  $\mathbf{w}_2$  determines the angle of inclination of the straight line to the axes.

If  $\mathbf{w}_1 = \mathbf{w}_2$ , we get a line going at an angle of 45 degrees  $(\frac{\pi}{4})$  to the axes  $\mathbf{x}_1$  and  $\mathbf{x}_2$  and dividing the first / third quadrants in half.

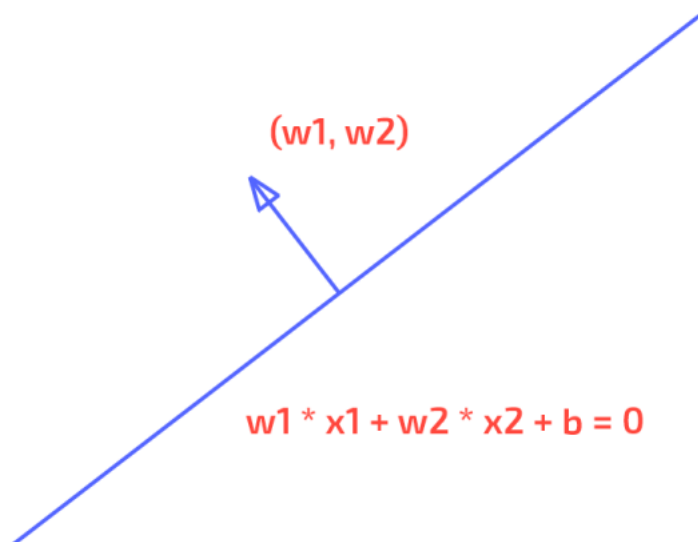
The nonzero coefficient  $b$  allows the line not to pass through zero. In this case, the inclination to the axes  $\mathbf{x}_1$  and  $\mathbf{x}_2$  does not change. Those  $b$  define

family of parallel lines: The



geometric meaning of a vector  $(w_1, w_2)$  is the normal to the straight line  $w_1 x_1 + w_2 x_2 + b = 0$ :

(If you do not consider the offset  $b$ , then  $w^T x$  is nothing more than a scalar product of two vectors. Equality to zero is equivalent to their orthogonality. Consequently,  $x$  - family of vectors orthogonal to  $(w_1, w_2)$ .)



PS It is clear that there are infinitely many such normals, as well as triples  $(w_1, w_2, b)$  defining a straight line. If all three numbers are multiplied by zero coefficient  $k$  - the line will remain the same.

In the general case of an  $n$ -dimensional space,  $(w_1, \dots, w_n, b)$  defines an  $n$ -dimensional hyperplane.

$$w_1 x_1 + w_2 x_2 + \dots + w_n x_n + b = 0$$

or:

$$\sum_{i=1}^n w_i x_i + b = 0$$

or:

$$w^T x + b = 0$$

## The geometric meaning of a linear combination

If the point  $\mathbf{x}(x_0, \dots, x_n)$  lies on the hyperplane, then

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

And what happens to this sum, if the point does not lie on the plane?

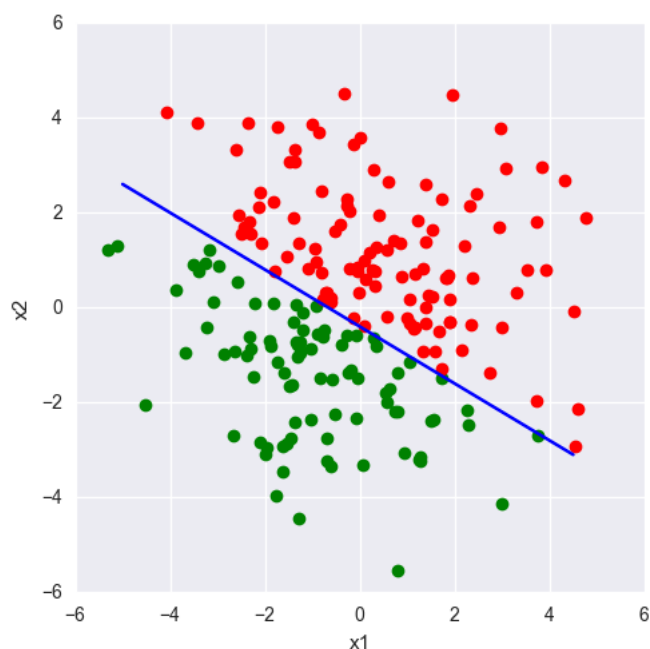
The hyperplane divides hyperspace into two hypersubspaces. So the points in one of these subspaces (conditionally speaking "higher" than the hyperplane) and the points in the other of these subspaces (conditionally speaking "below" the hyperplane) will give a different sign in this sum:

$\mathbf{w}^T \mathbf{x} + b > 0$  - the point lies "above" the hyperplane

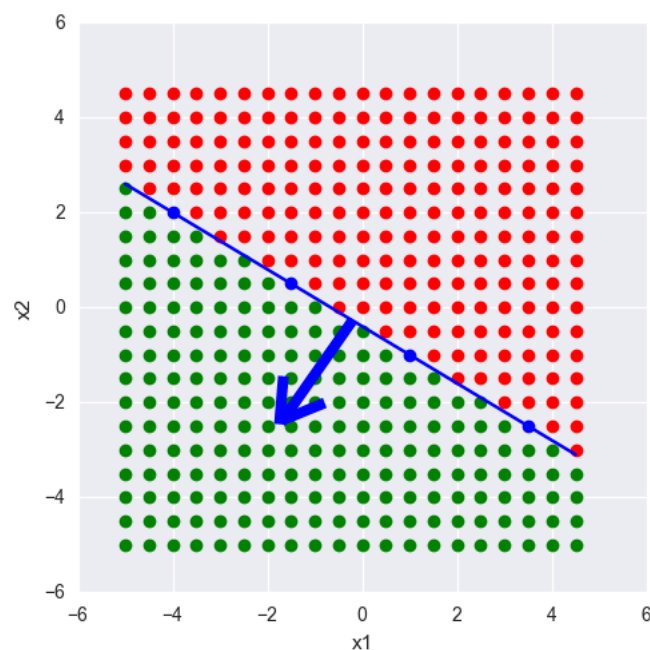
$\mathbf{w}^T \mathbf{x} + b < 0$  - the point lies "below" the hyperplane

This is a very important observation, so I suggest that it be cross-checked with simple code in Python:

► [Code example in Python](#)



It is necessary to understand that "above" and "below" here are conditional concepts. This is specifically reflected in the example - the green dot: visually lower. From the geometric point of view, the direction "higher" for a given particular line is determined by the normal vector. Where to look normal, and there top:



Thus The sign of a linear combination allows one to assign a point to the upper or lower subspace.

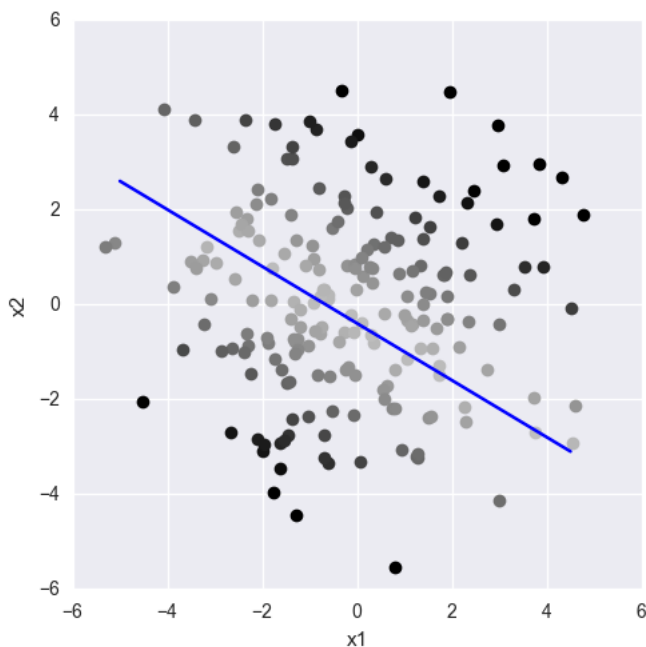
And the meaning? The value (modulo) determines the distance from the plane:

$$\text{dist}(x) = \frac{|w^T x + b|}{\|w\|}$$

Those. The farther from the plane the point is, the greater the value of the linear combination for it. If we fix the value of a linear combination, we get points lying on a line parallel to the original one.

Again, the observation is important, so we recheck:

► [Code example in Python](#)



Everything converges.

## conclusions

- A linear combination allows us to divide the n-dimensional space by a hyperplane.
- The points on opposite sides of the hyperplane will have different signs of the linear combination  $w^T x + b$ .
- The point is farther from the hyperplane, the absolute value of the linear combination will be larger.

From the standpoint of binary classification, the last statement can be reformulated as follows. The more distant the point from the hyperplane, the more confident we are that our sample (sample) determined by this point falls into one or another class.

## Close and far: how is it?

Close and far - the concepts are purely subjective. And when classifying an answer we need to clearly - either the detail is suitable for the construction of a missile to fly to Mars, or it is a marriage. Either the person clicks on the advertisement, or not. It is possible to answer with a certainty of confidence - to give the probability of a positive (true) outcome.

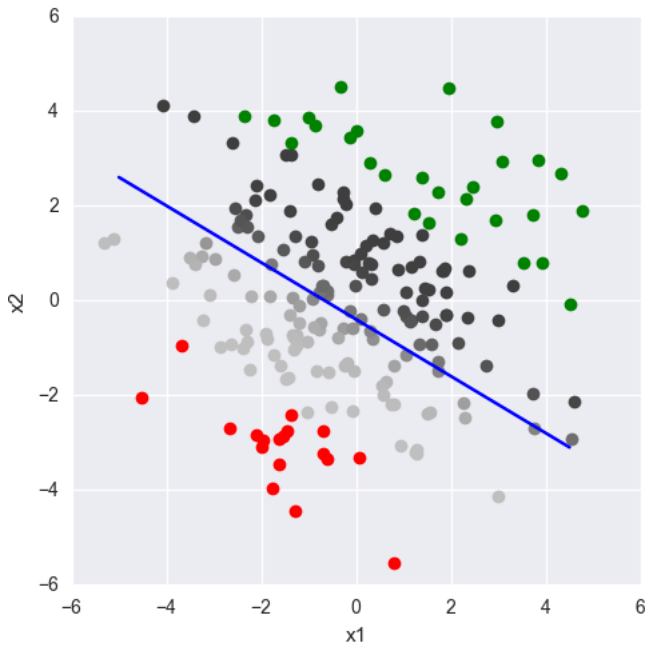
To do this, you can apply the activation function to the linear combination (in the terminology of neural networks).

If you apply the logistic function (see the chart below):

$$\text{logistic}(x) = \frac{1}{1 + e^{-x}}$$

we get the probability and such a picture on the output:

► [Code example in Python](#)



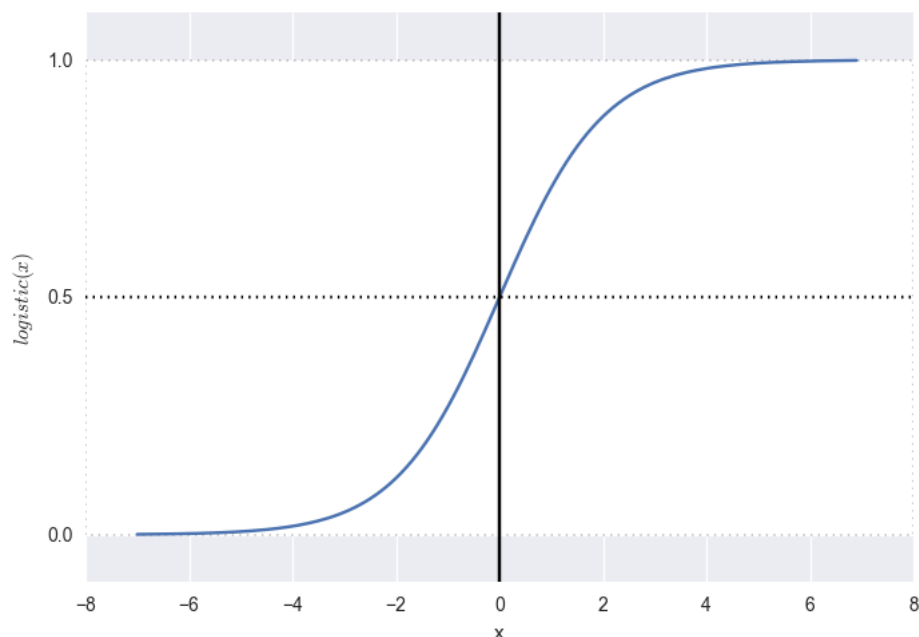
Red - just not (false, just a marriage, just do not click). Green - exactly yes (true, exactly suitable, just click). All that in a certain range of proximity the hyperplane (the boundary of the solutions) gets some probability. On the most direct probability is exactly 0.5.

PS "Exactly" here is defined as less than 0.001 or greater than 0.999. The logistic function itself tends to zero by minus infinity and to unity by plus infinity, but never takes these values.

**NB** Note that this example only demonstrates how you can squash the distance with the sign in the probability interval  $(0, 1)$ . In practical problem calibration of probabilities is used to find the optimal mapping. For example, in the Platt scaling algorithm, the logistic function is parameterized:

$$f(x) = \frac{1}{1 + e^{Ax+B}}$$

and then the coefficients  $A$  and  $B$  are selected by machine learning. For more information, see: binary classifier calibration, probability calibration



## In what space are we? (a useful speculative exercise)

It would seem clear - we are in the data space  $\mathbf{X}$  (data space), in which the samples lie  $\mathbf{x}$ . And we are looking for the optimal separation by the plane determined by the vector  $\mathbf{w}$ .

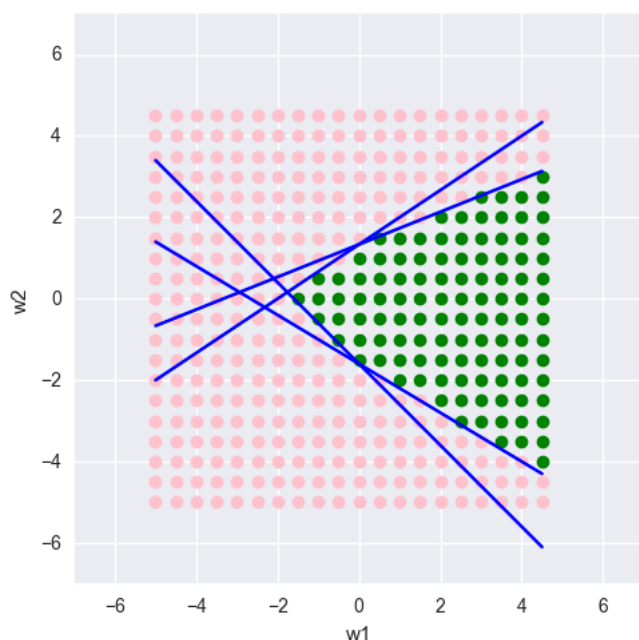
$\mathbf{w}^T \mathbf{x} + \mathbf{b} > 0$  for green dots

$\mathbf{w}^T \mathbf{x} + \mathbf{b} < 0$  for red points

But in our problem of binary classification the samples are fixed, and weights change. Accordingly, we can replay everything by moving into the space of the weights  $\mathbf{W}$  (weight space):

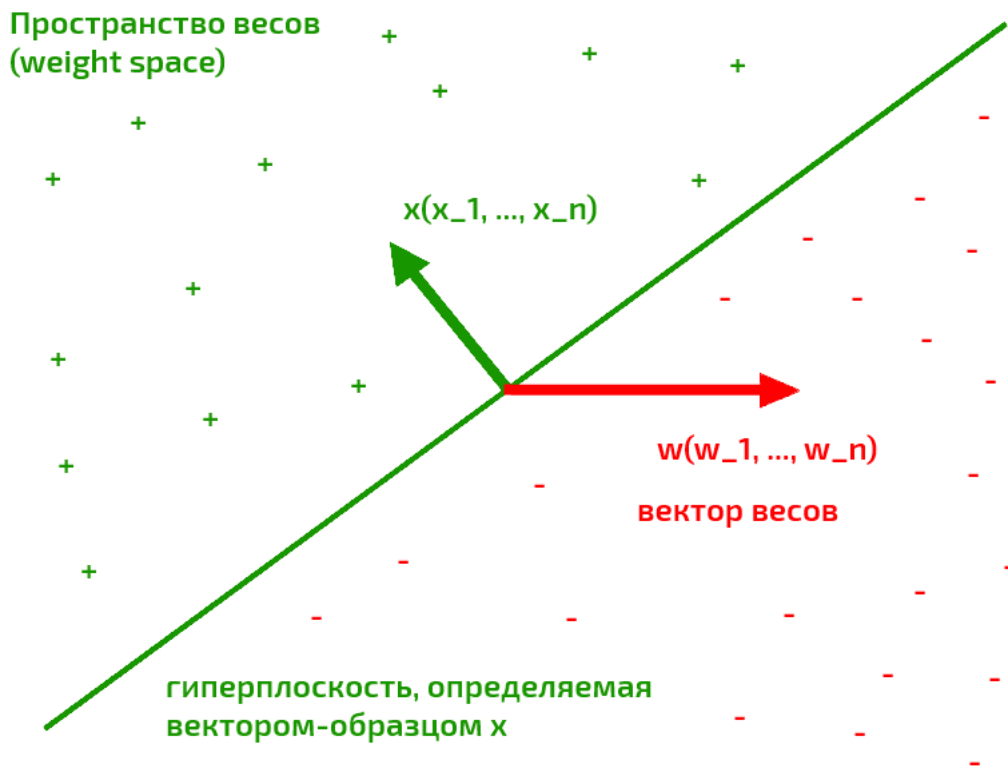
$$\mathbf{x}^T \mathbf{w} + \mathbf{b}$$

Samples from the training set  $\mathbf{x}_1 \dots \mathbf{x}_m$  in this case the  $m$  hyperplanes and our task is to find such a point  $\mathbf{w}$ , which would lie on the right side of the plane. If the original dataset is linearly separable, then such a point exists.



► [Code example in Python](#)

When teaching the model it is more convenient to reason in the space of weights, since the weights are updated, and the sample vectors from the training set set the normals to the hyperplanes. For example:



Suppose that a sample  $x$  corresponds to the green class corresponding to the inequality:

$$x^T w + b > 0$$

Because on illustration vector  $w$  looks against the normal  $x$ , then the value of the linear combination will be negative - hence we have a classification error.

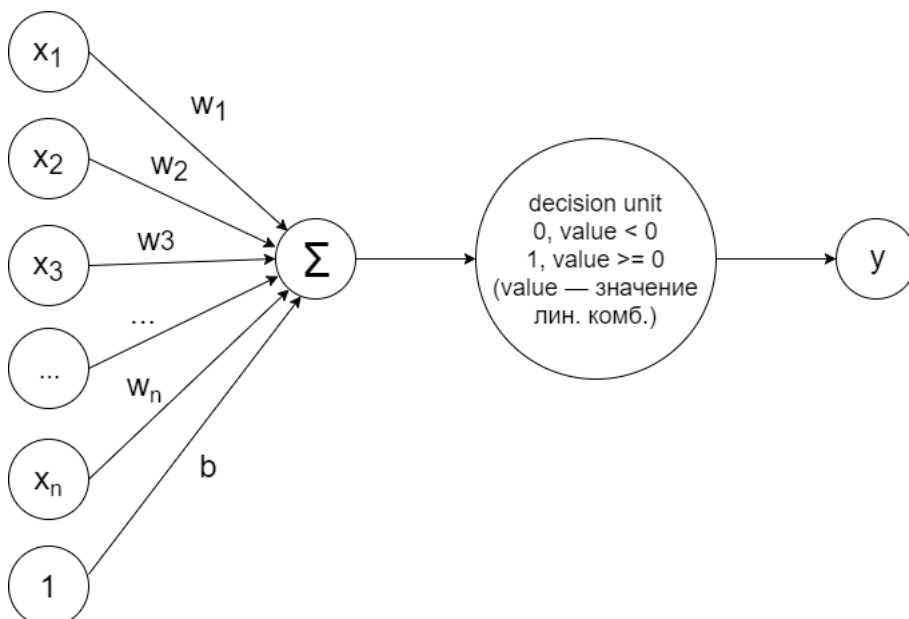
Accordingly, it is necessary to update the vector  $w$  in the direction indicated by the normal:

$$w_{new} = w_{old} + \lambda x, \text{ where } \lambda > 0$$

with a certain "speed"  $\lambda$ . Thus, in the next step, the prediction will be either true or less incorrect, since term  $\lambda x$ , co-directional with the normal, "entrust" the vector of weights in the green area.

## Practice. We teach perceptron

To solve the problem of binary classification in the case of linear separability of samples, one can teach a simple perceptron arranged according to the scheme:





This construction implements exactly the principle that was described above. A linear combination is calculated:

$$\sum_{i=1}^n w_i x_i + b$$

By the value of which the decision unit (decision unit) decides to assign a sample to one of the two classes according to the following principle:

$$w^T x + b \geq 0 \text{ class } +1 \text{ (green dots)}$$

$$w^T x + b < 0 \text{ class } -1 \text{ (red dots)}$$

Initially weights are initialized randomly, and at each step of the training the following algorithm is done for each sample:

The predicted label is calculated. If it does not coincide with the real class, the weights are updated according to the following principle:

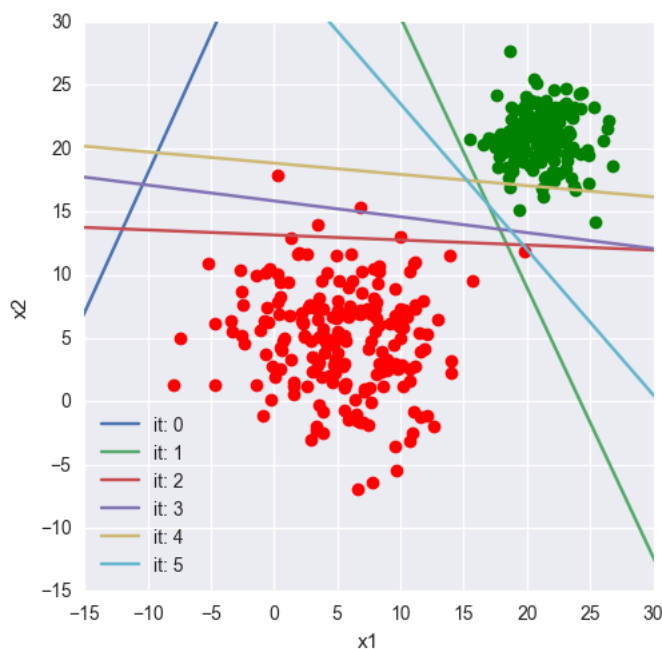
$$w_{new} = w_{old} + y_n x_n$$

$$b_{new} = b_{old} + y_n$$

Where  $y_n$  - real class of the sample  $x_n$ . Why does this work described above in the speculative exercise with the transition to the space of the weights? Briefly:

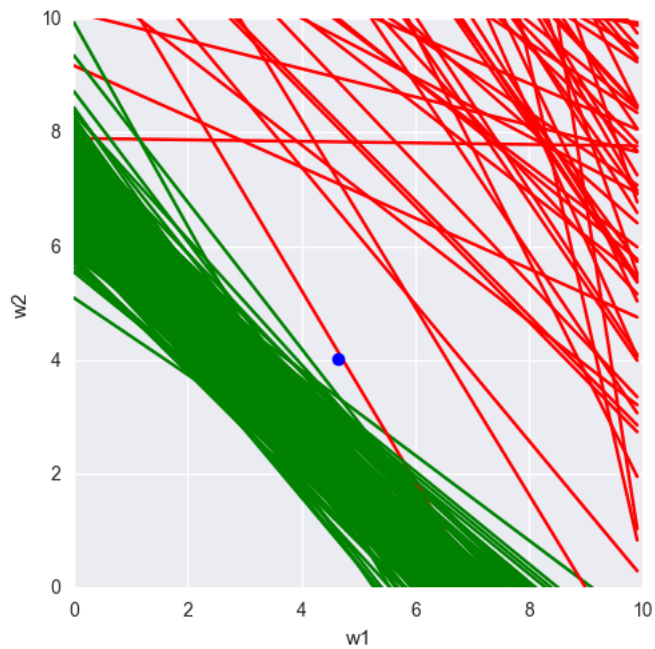
- We finish the vector-weight in the direction of the correct class: normal  $x_n$  in the case of class +1; against the normal  $x_n$  in the case of class -1 (normal itself always looks toward the +1 class.)
- We update the offset by a similar principle.

Here's what happens:



► [Python code](#)

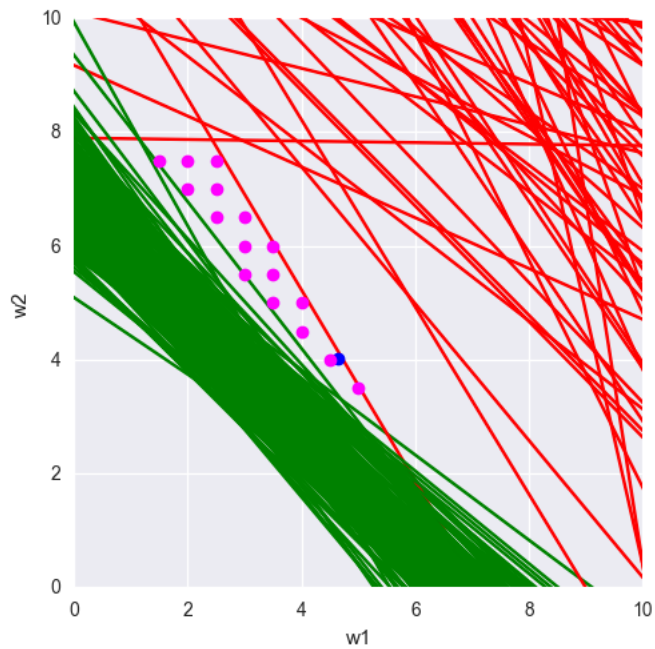
We now look at the space of the weights  $W$  (weight space):



► [Python code](#)

Red and green lines are the original samples, the blue dot is the total weight.

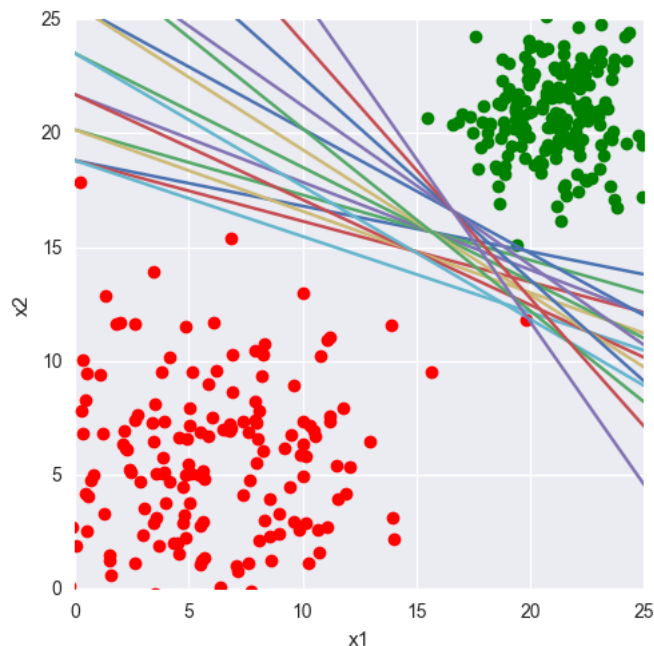
And what other weights give a correct classification? We look:



► [Python code](#)

The red and green lines are the original samples, the blue dot is the final weight, the purple dots are the other possible weights.

And we turn everything inside out again, going back to the data space  $\mathbf{X}$  (data space):



Those weights that were marked with purple dots in the space of the weights above were here, in the data space, lines of other possible solution boundaries.

**Exercise (simple):** The last illustration has four characteristic bundles of lines. Find them among the purple dots in the space of the weights.

## From the author

Thanks to all the Khabrovians for critical comments on the first version of the article, including [@yorko](#), which, together with the [Open Data Science](#) community, is doing a super-open, open-source computer training course - I recommend it to everyone.

It became clear that the material lacked the final touch and the article was sent for revision. The second (the same current) version is supplemented with an example of perceptron training.

## Outcomes

I hope this article will allow you to better understand and feel the geometric meaning of linear combinations. Below are links to materials used in the preparation of the article and interesting in terms of deepening the topic. (All materials in English.)

- [Geoffrey Hinton. An overview of the main types of neural network architecture](#) More information about the perceptron trained with the transition into the space weights of neural networks guru Geoffrey Hinton.
- [Supervised Learning / Support Vector Machines To solve the problems of binary classification by the method of reference vectors and how from the point of view of this algorithm to choose the optimal dividing plane.](#)
- [Hyperplane based classification: Perceptron and \(Intro to\) Support Vector Machines](#) Again about perceptrons, in passing about the method of reference vectors. The issue of training and correcting weights is considered in detail.
- [Polyhedra and Linear Programming. Polyhedra, Polytopes, and Cones](#) Linear algebra of linear combinations (theory) with many illustrative examples.

**Tags:** [neural networks](#), [machine learning](#), [logistic regression](#), [support vector method](#)



**44.0**  
Karma

**0.0**  
Rating

**44**  
Subscribers

Denis Kulagin @kdenisk

Computer Linguistics

Site

In contact with

Share this publication

f

twitter

vk

telegram

RELATED PUBLICATIONS

June 14, 2013 at 13:04

Support vectors method for finding polymorphisms in the genome

↑ +27

8,7k

87

9

eye

bookmark

comment

October 12, 2011 at 19:51

Classification of documents by the method of reference vectors

↑ +38

13.7k

132

20

eye

bookmark

comment

September 29, 2010 at 20:20

Classification of data by the method of reference vectors

↑ +74

66.5k

201

27

eye

bookmark

comment

CAREERS

Мой к

Developer (machine learning)

SEC "Volcano" • Moscow

from 100 000 to 150 000

Python programmer (backend)

VITAL • Togliatti

from 70 000 to 150 000

Software Engineer - Computer Vision

AIFactory.app • St. PetersburgRemote • work is possible

from 100 000 to 180 000

Machine Learning Engineer (Deep Learning, NLP)

whisk Remote • work is possible

from 4 000

Lead Developer (Face Recognition)

Facechain • St. PetersburgRemote • work is possible

from 150 000 to 250 000

All jobs

Advertising

Comments 5

ServPonomarev 24.03.17 at 13:21

#

bookmark

↑







About the sigmoid in the section 'close-far' does not agree.

Take our regression, take the training and test samples. We find the maximum value of false positives and the minimum value of false-negative. Those points that lie above and below these boundaries - can be considered points of high confidence in the results. Those points that are located between these two boundaries - this is the working area of the classifier. If we plot the completeness-accuracy graphs for this region, then we will see a classic cross. The

09/09/2018      Geometry of machine learning. Separating hyperplanes or what is the geometric meaning of a linear combination? / Habr





intersection of the accuracy and completeness charts can be taken as 50%, the lower limit as 0%, the upper limit as 100% (yes, it usually turns out to be asymmetric). Then the answer of the classifier will be a human-intuitive estimate in the form of interest.

If you want to use a sigmoid, you can also, but the limits of the trigger ("confidence") are then determined by the number of sigma. 2 sigma for business is already above the roof.





 **kdenisk** 24.03.17 at 2:33 pm     

I agree. True, you are talking about the interpretation of the results of the classifier, and the example showed how the distances from  $R$  are nonlinearly squashed into the probability interval  $(0, 1)$ .

The limits of confidence in this case are fixed and chosen simply for illustration purposes.

 **Dark\_Daiver** 03/24/17 at 18:30   








Probably the most illustrated explanation of the linear combination that I saw in my life

 **yorko** 24.03.17 at 19:43   

And the purpose of the article was to explain only the meaning of a linear combination?

And then there are so many words about logistic regression, SVM, neural networks, prostigospodi. And so quickly everything ended.

Ps. It can be clarified that the last thing you came to is a simple perceptron, a brick laid in the foundation of neural networks.

 **kdenisk**  03/27/17 12:01     

Thanks for the comment. Added an example of training perceptron for completeness of presentation.

PS Yes, in general, the article is about a linear combination and a hyperplane. I think this is quite an important topic to give her a separate attention.

Only [full users](#) can leave comments. [Come in](#) , please.

MOST READ

Day

A week

Month


How World of Warcraft was created: 20-year view from inside


↑ +41


24.5k

137

46








New process technology for the production of microcircuits is increasingly postponed - why?


↑ +56


25.3k

36

88








What is the importance of  $196\,884 = 196\,883 + 1$ ? How to explain it on the fingers?


↑ +28


8.8k

74

26








Practical use of neural networks


↑ +10


5.4k

43

30









Drama holes in the "Union" turned into a phantasmagoria


↑ +78



30.8k


42







My Account	Sections	Information	The services	Apps
<a href="#">To come in</a> <a href="#">check in</a>	<a href="#">Publications</a> <a href="#">Hubs</a> <a href="#">Company Profile</a> <a href="#">Members List</a> <a href="#">Sandbox</a>	<a href="#">rules</a> <a href="#">Help</a> <a href="#">Documentation</a> <a href="#">Agreement</a> <a href="#">Confidentiality</a>	<a href="#">Advertising</a> <a href="#">Rates</a> <a href="#">Content</a> <a href="#">Seminars</a>	<div> <small>Загрузите в</small> <b>App Store</b></div> <div> <small>доступно в</small> <b>Google</b></div>

 © 2006 - 2018 « TM »

[About the site](#)

[Support service](#)

[mobile version](#)

