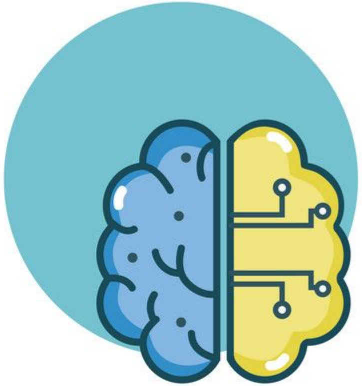
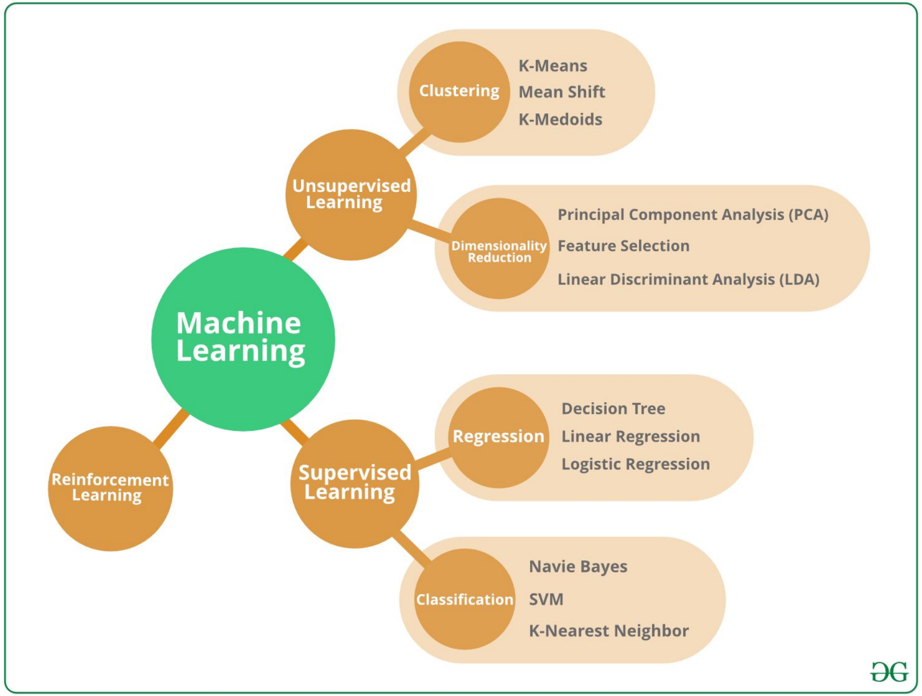
INTRODUCTION TO MACHINE LEARNING

Linear Models

Elisa Ricci



Machine Learning Models



Machine learning models

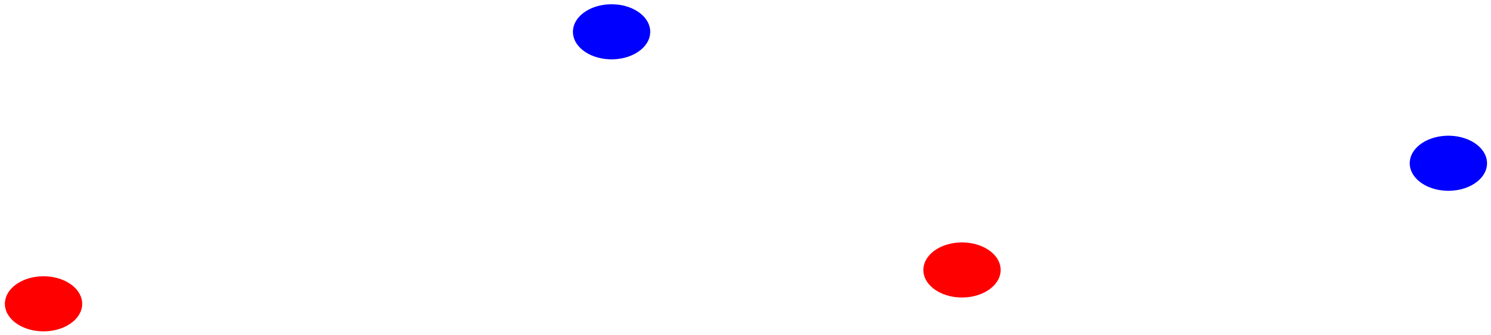
Some machine learning approaches make strong assumptions about the data

- If the assumptions are true it can often lead to better performance
- If the assumptions aren't true, the approach can fail miserably

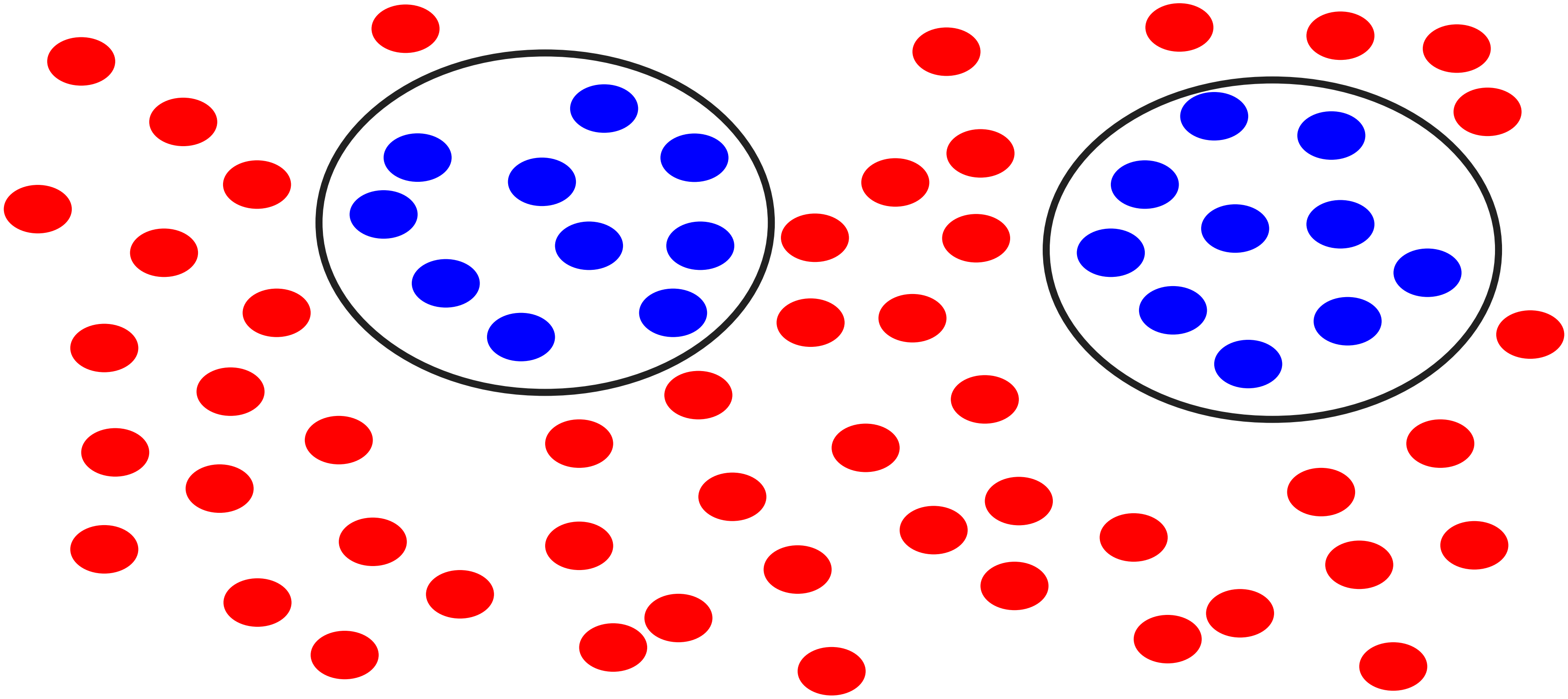
Other approaches don't make many assumptions about the data

- This can allow us to learn from more varied data
- But, they are more prone to overfitting and generally require more training data

What is the data generating distribution?



Actual model

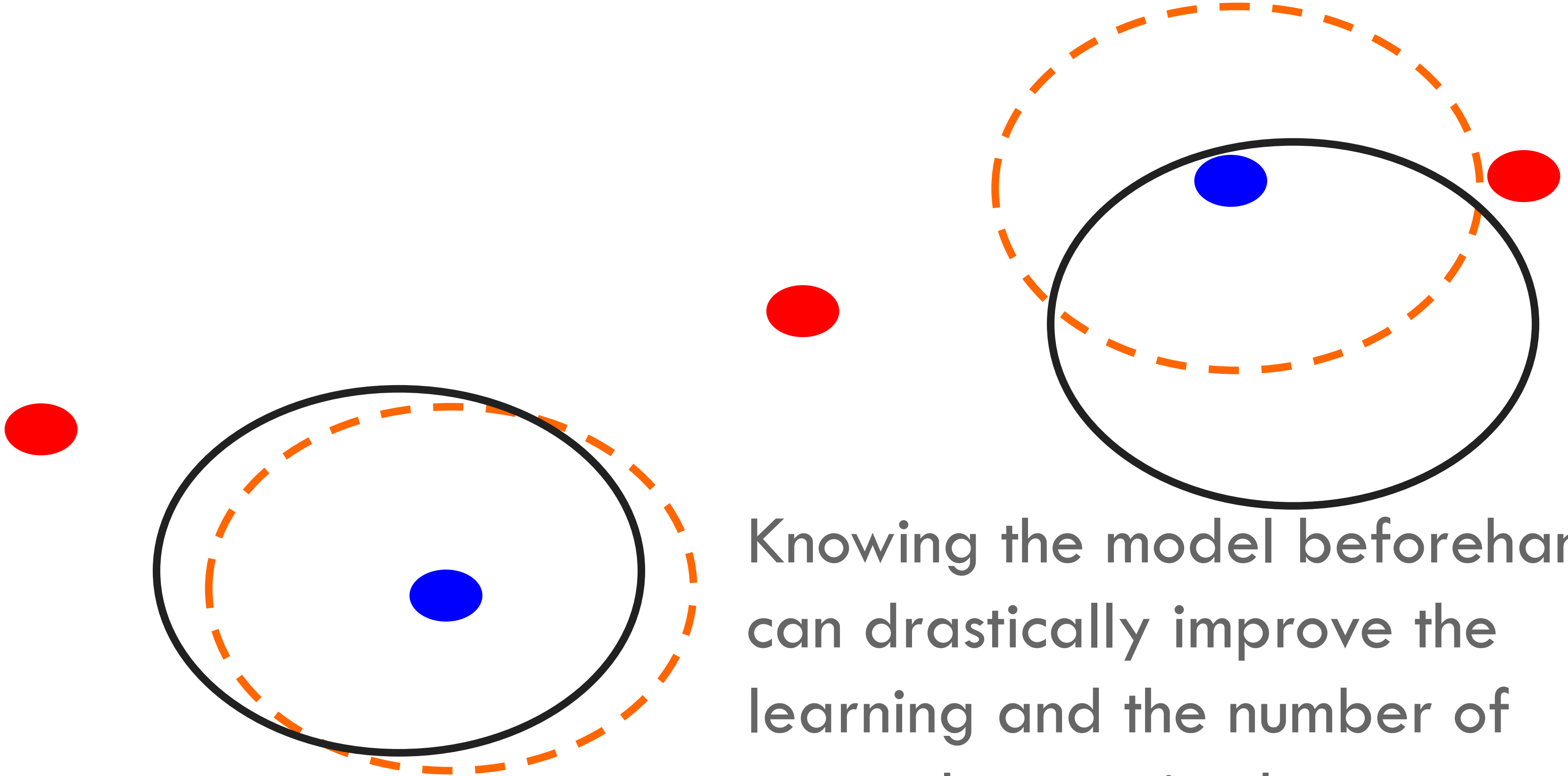


Model assumptions

If you don’t have strong assumptions about the model, it can take you a longer to learn

Assume now that our model of the blue class is two circles

What is the data generating distribution?



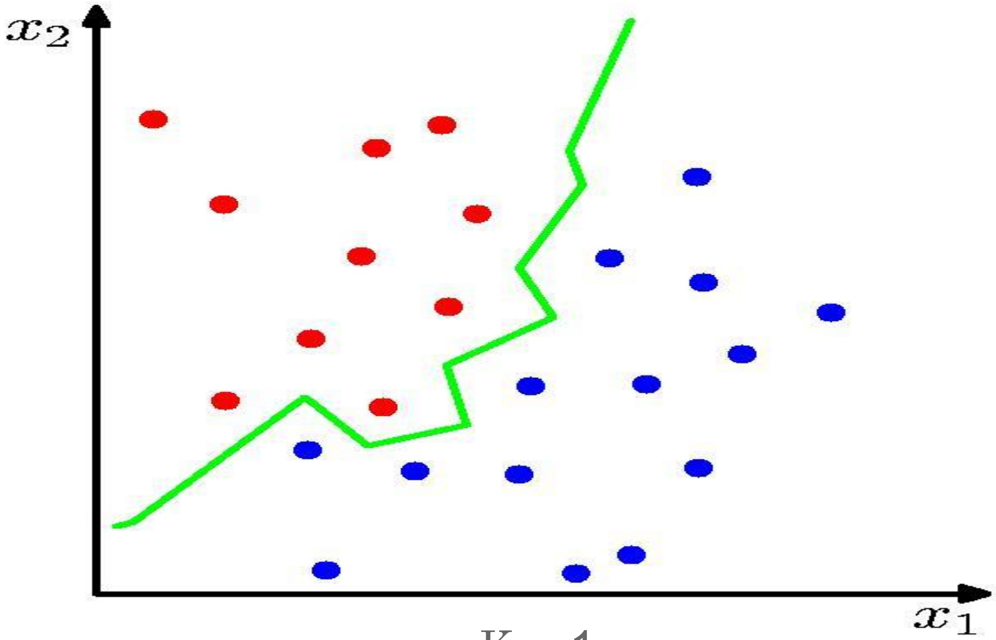
Knowing the model beforehand can drastically improve the learning and the number of examples required

Machine learning models

What are the ***model assumptions*** (if any) that *k*-NN make about the data? k-nn non fa alcuna assunzione sulla distribuzione dei dati

Are there data sets that could never be learned correctly by it?

k-Nearest Neighbor (k-NN)



K = 1

No model assumptions. Assumes that proximity relates to class

kNN can learn any arbitrary separation between the classes

Bias

The "bias" of a model is how strong the model assumptions are.

- low-bias classifiers make minimal assumptions about the data (k-NN and DT are generally considered low bias)

  decision trees

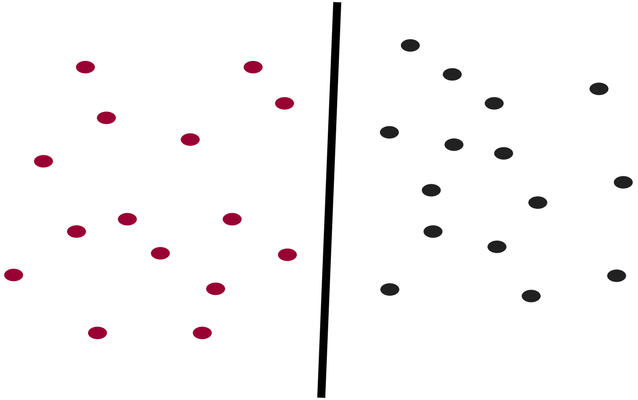
- high-bias classifiers make strong assumptions about the data

Linear models

A strong high-bias assumption is **linear separability**:

- in 2 dimensions, can separate classes by a line
- in higher dimensions, need hyperplanes

A **linear model** is a model that assumes the data is linearly separable



quindi no multi class classification
massimo 2 classi

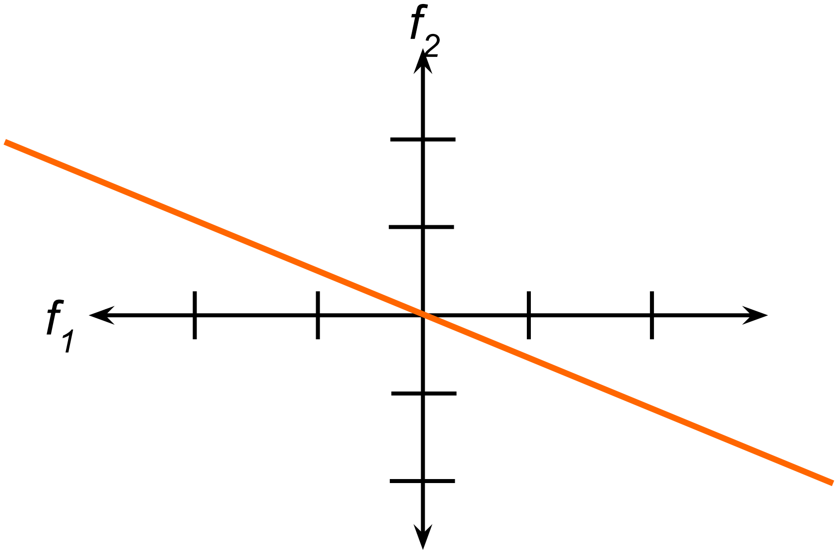
Defining a line

Any pair of values (w_1,w_2) defines a line through the origin:

$$0 = w_1 f_1 + w_2 f_2$$

$$0 = 1 f_1 + 2 f_2$$

-2	1
-1	0.5
0	0
1	-0.5
2	-1



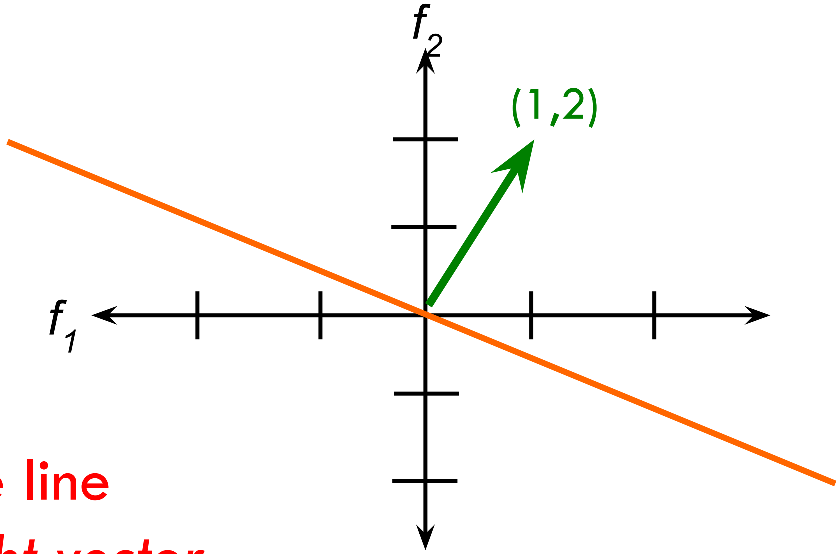
Defining a line

Any pair of values (w_1,w_2) defines a line through the origin:

$$0 = w_1 f_1 + w_2 f_2$$

$$0 = 1 f_1 + 2 f_2$$

w=(1,2)



We can also view it as the line perpendicular to the *weight vector*

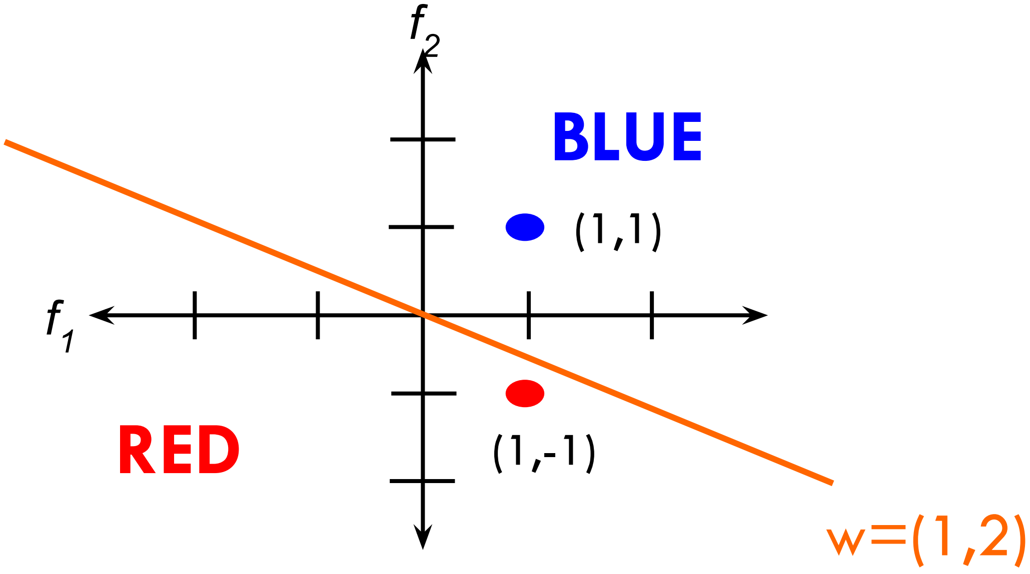
Classifying with a line

Mathematically, how can we classify points based on a line?

$$0 = 1f_1 + 2f_2$$

(1,1): $1*1+2*1=3$

(1,-1): $1*1+2*-1=-1$



The sign indicates which side of the line

Defining a line

weights

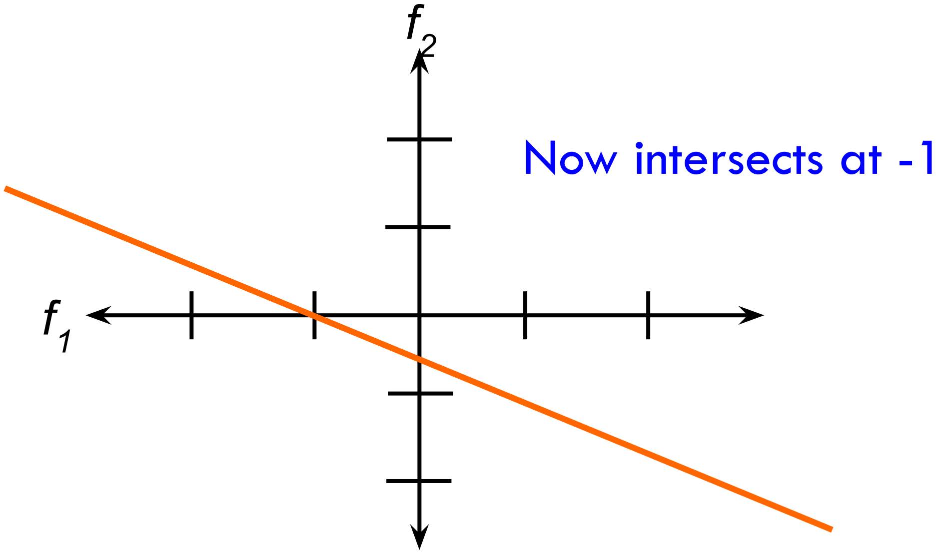
Any pair of values (w_1,w_2) defines a line through the origin:

bias

$$a = w_1 f_1 + w_2 f_2$$

$$-1 = 1 f_1 + 2 f_2$$

-2	0.5
-1	0
0	-0.5
1	-1
2	-1.5



Linear models

A linear model in *n*-dimensional space (i.e. *n* features) is defined by *n*+1 weights. In two dimensions, we have a line:

$$0 = w_1 f_1 + w_2 f_2 + b$$

(where b = -a)

In three dimensions, a plane:

$$0 = w_1 f_1 + w_2 f_2 + w_3 f_3 + b$$

In *n*-dimensions, a **hyperplane**

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

Classifying with a linear model

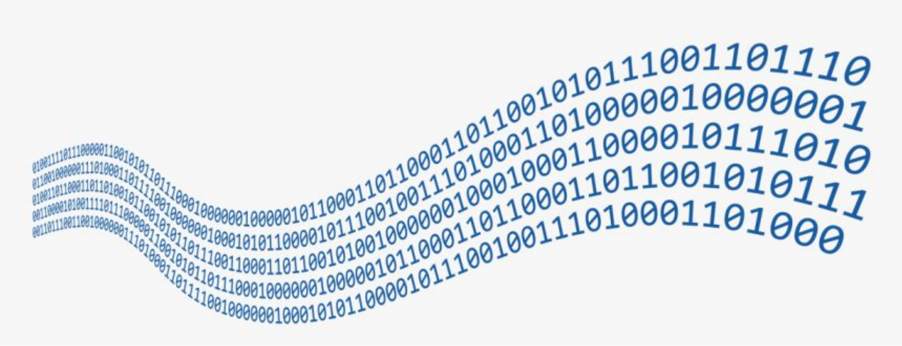
We can classify with a linear model by checking the sign:

inference phase

$f_1, f_2, \ldots, f_n$ → classifier

$b + \sum_{i=1}^{n} w_i f_i > 0$ Positive example

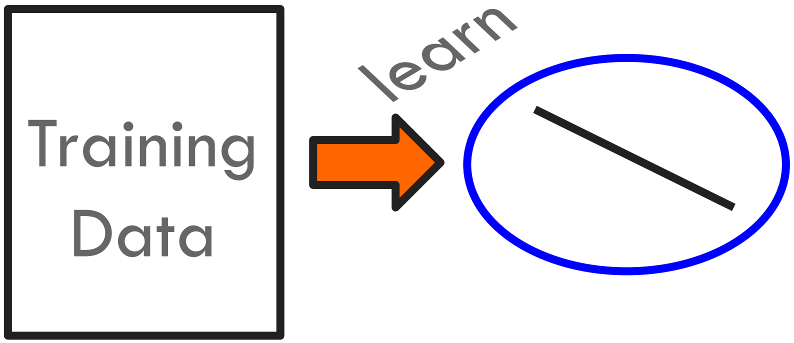
$b + \sum_{i=1}^{n} w_i f_i < 0$ Negative example



Online Learning

How do we learn a linear model?

Given a linear model (i.e. a set of weights w_i and b) we can classify examples



(data with labels)

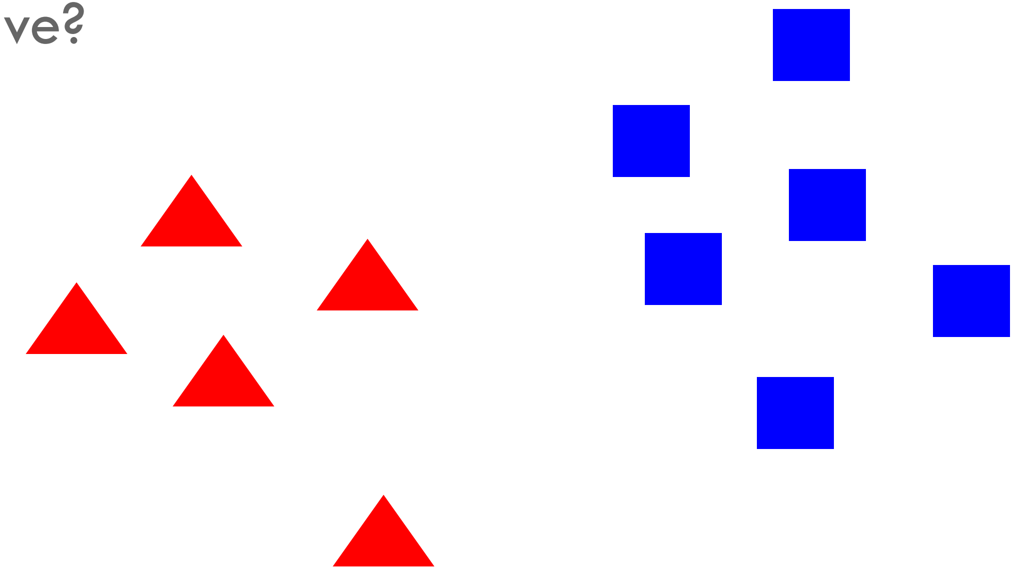
How do we learn a linear model?

Learning a linear model

Positive or negative?

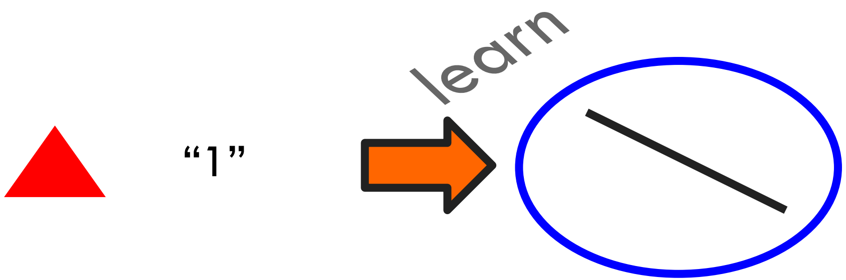
Learning a linear model

Positive or negative?



Online learning algorithm

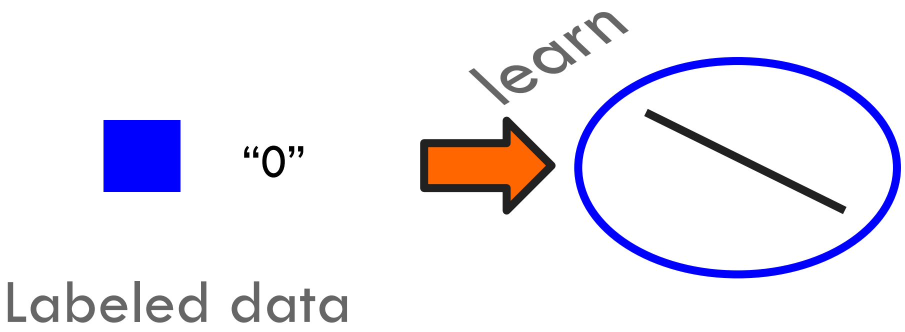
We only see one example at the time!



Labeled data

Online learning algorithm

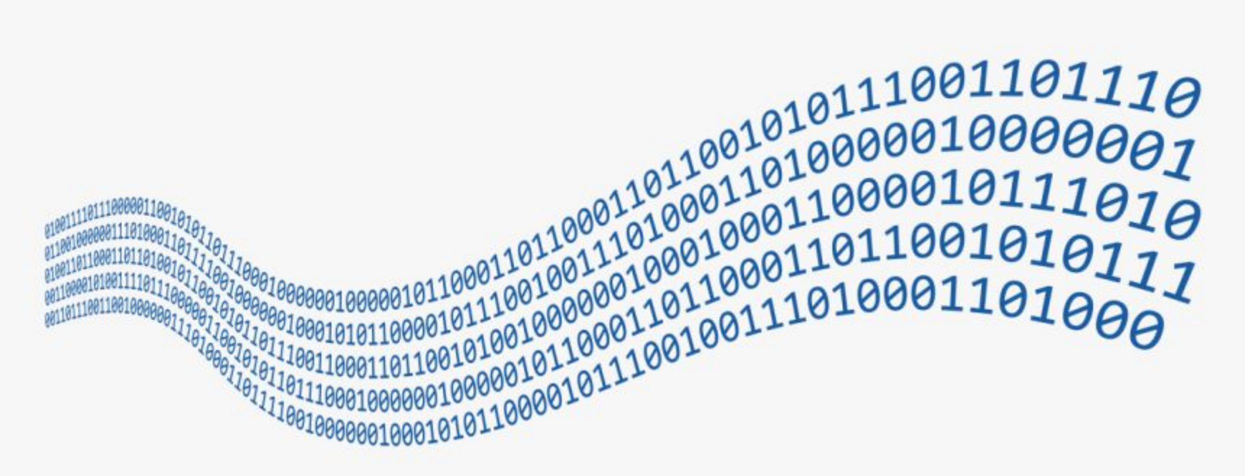
We only see one example at the time!



Learning a linear model

When we need online learning?

Data Streams!



Learning a linear model

When we need online learning?

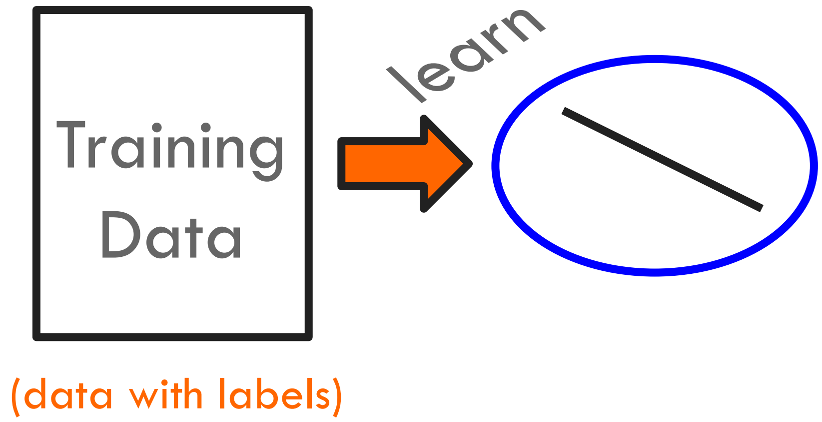
Privacy-preserving applications

Lesson learned: Online VS Batch

- **Batch:** Given training data $\{(x_i, y_i) : 1 \le i \le n\}$, typically i.i.d.

  indipendenti identicamente distribuiti

- **Online:** data points arrive one by one
  - The algorithm receives an unlabeled example x_i
  - The algorithm predicts a classification of this example.
  - The algorithm is then told the correct answer y_i, and update its model

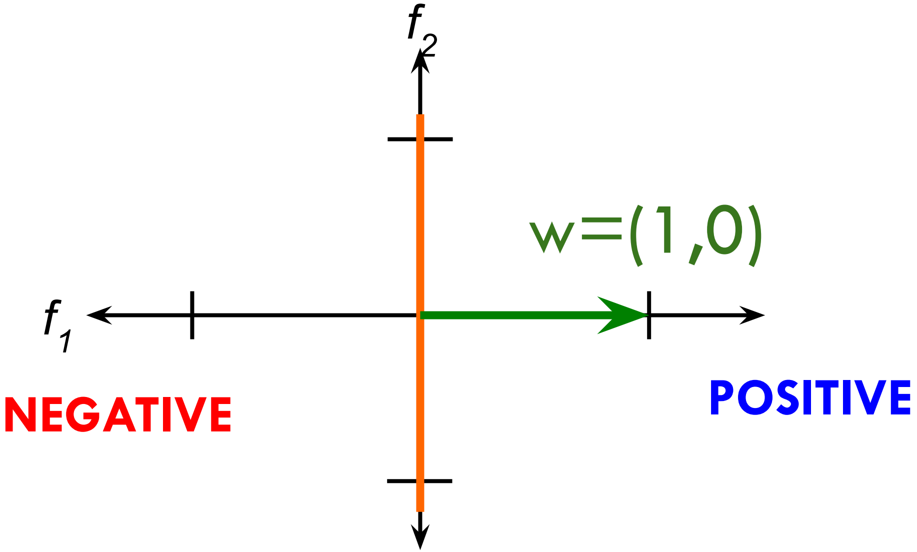


Learning a Linear Model

Learning a linear classifier

According to the rule we have seen before:

$$0 = w_1 f_1 + w_2 f_2$$

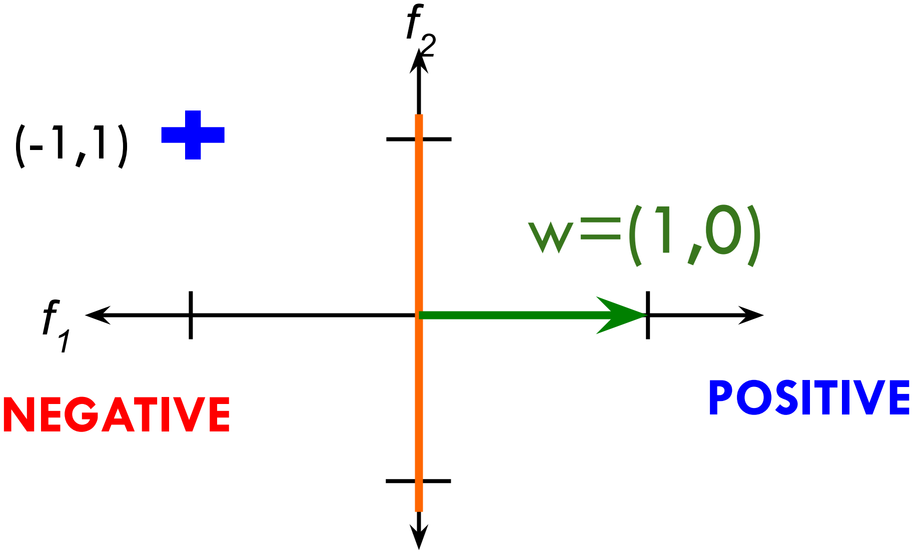


Learning a linear classifier

Now a new sample arrive. It is a positive sample:

$$1 * f_1 + 0 * f_2 =$$

$$1 * -1 + 0 * 1 = -1$$



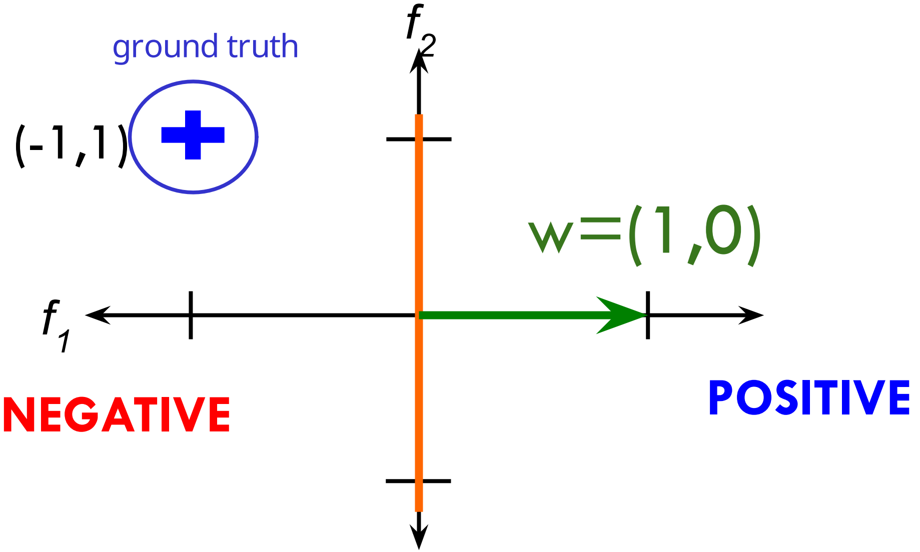
Negative, wrong!

Learning a linear classifier

Now a new sample arrive. It is a positive sample:

$$1 * f_1 + 0 * f_2 =$$

$$1 * -1 + 0 * 1 = -1$$ prediction



Negative, wrong!

Model must be updated!

A closer look at why we got it wrong

$$\overset{w_1}{1} * f_1 + \overset{w_2}{0} * f_2 =$$

$$1 * -1 + 0 * 1 = -1$$

This value should be positive!value

(-1, 1) +

contributed in the wrong direction

could have contributed (positive feature) but it did not since the weight is 0

A closer look at why we got it wrong

$$\overset{w_1}{1} * f_1 + \overset{w_2}{0} * f_2 =$$

$$1 * -1 + 0 * 1 = -1$$

(-1, 1) **+**

This value should be positive!value

decrease
e.g. from 1 to 0

increase
from 0 to 1

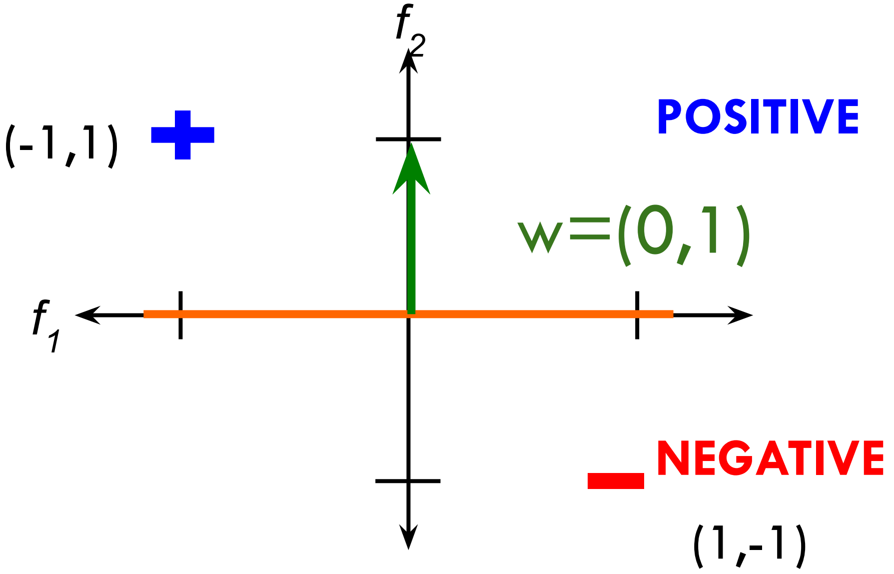
Learning a linear classifier

$$0 = w_1 f_1 + w_2 f_2$$

$$0 * f_1 + 1 * f_2 =$$

$$0 * 1 + 1 * -1 = -1$$

Is it correct? YES

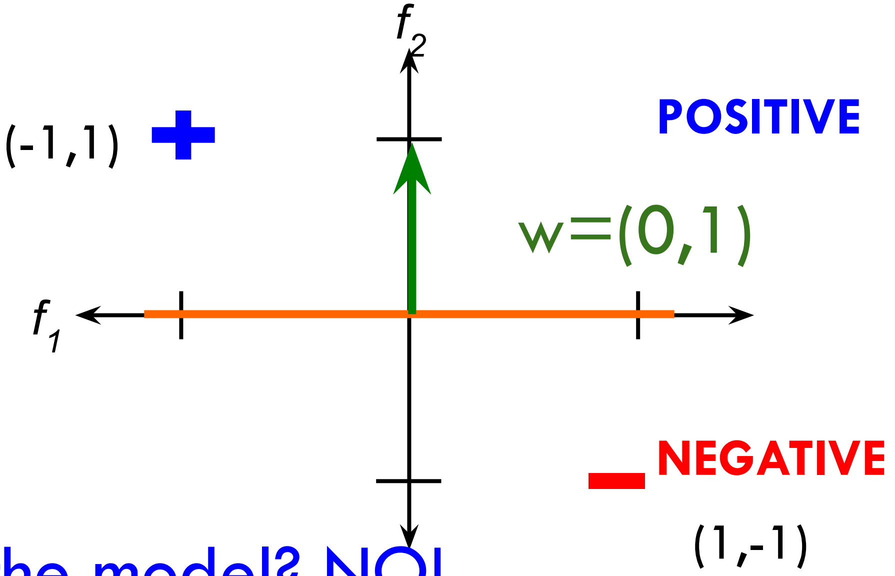


Learning a linear classifier

$$0 = w_1 f_1 + w_2 f_2$$

$$0 * f_1 + 1 * f_2 =$$

$$0 * 1 + 1 * -1 = -1$$



Do we need to update the model? NO!

Perceptron learning algorithm

repeat until convergence (or for some # of iterations):

for each training example (f_1, f_2, …, f_n, label):

check if it is correct based on the current model

if not correct, update all the weights:

for each w_i:

$w_i = w_i + f_i$*label

$b = b$ + label

questo è un esempio

```
label is -1/1
```

label = ground truth

il for non è presente nel caso dell' online learning

Perceptron learning algorithm

repeat until convergence (or for some # of iterations):

for each training example (f_1, f_2, ..., f_n, label):

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

if *prediction is different from label* prediction * label <= 0

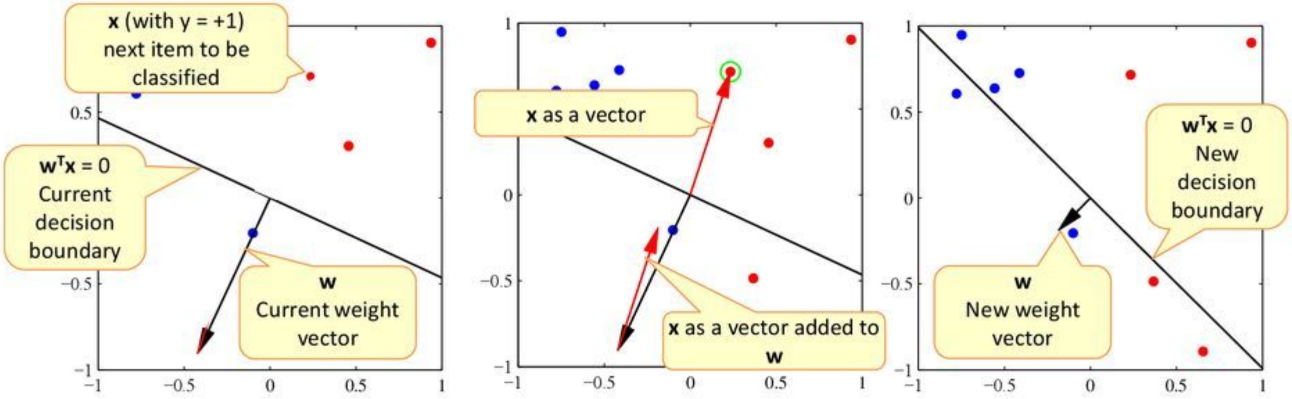
moltiplicazione da come risultato un numero negativo
se i segni non sono concordi
il perceptron crea una linea che separa i sample
in positivi e negativi

for each w_i:

$w_i = w_i + f_i$*label

$b = b$ + label

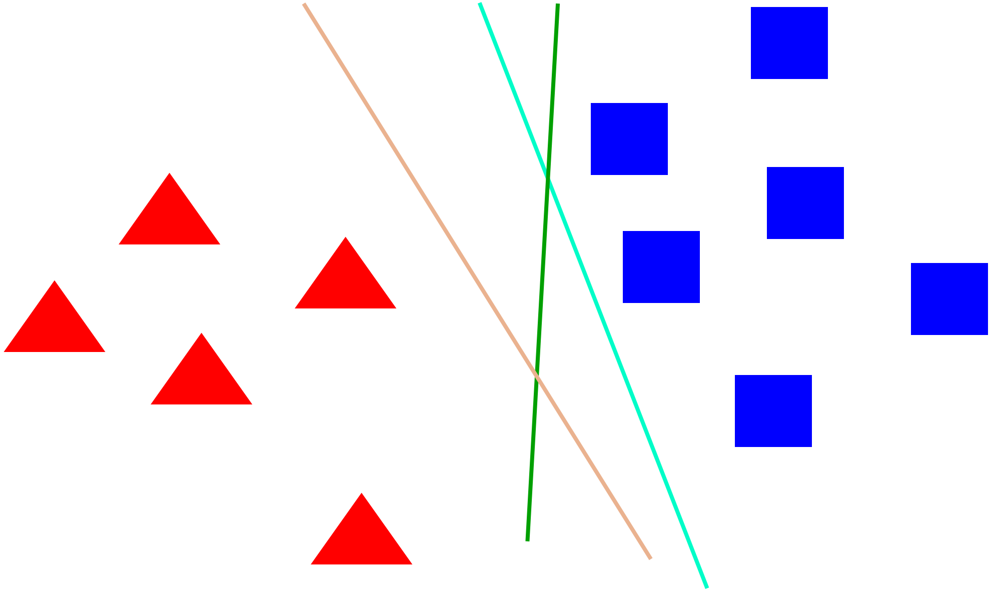
Perceptron In Action



(Figures from Bishop 2006)

Which line will the perceptron find?

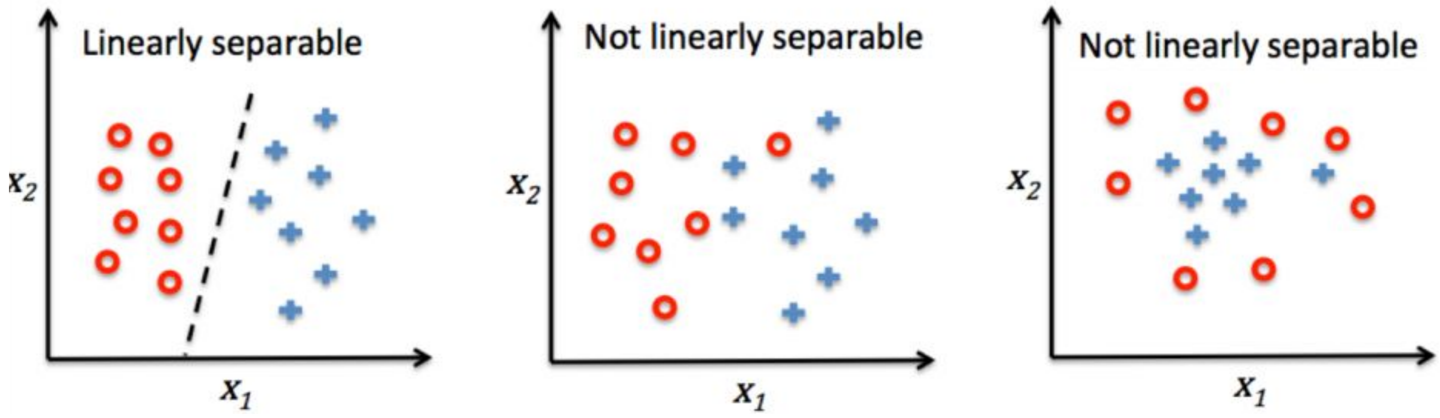
poor notion of geralization
alcune linee generalizzano meglio di altre



Only guaranteed to find ***some*** line that separates the data!

Linear Separable sets

The training instances are linearly separable if there exists a hyperplane that will separate the two classes.



Number of Iterations

repeat until convergence (or for some # of iterations):

for each training example (f_1, f_2, …, f_n, label):

check if it is correct based on the current model

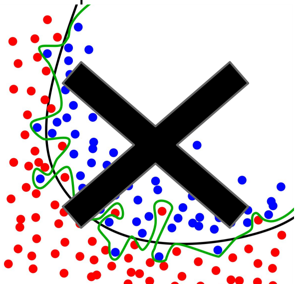
if not correct, update all the weights:

for each w_i:

$w_i = w_i + f_i$*label

$b = b$ + label

Limit overfitting!!!!



SAMPLE ORDER?

soluzione non basata sull'errore ma su
- ordine di arrivo dei sample
- itarazioni

repeat until convergence (or for some # of iterations):

random sample one example (f_1, f_2, ..., f_n, label):

check if it is correct based on the current model

if not correct, update all the weights:

for each w_i:

$w_i = w_i + f_i$*label

$b = b +$ label

QUESTIONS?