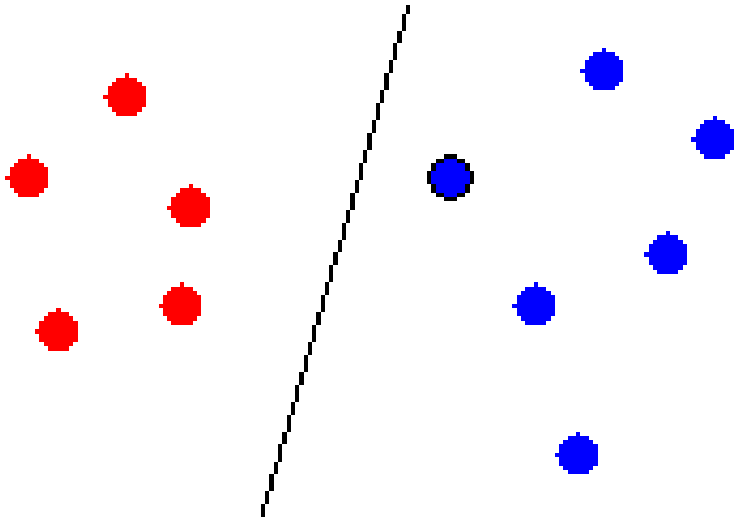


Linear classification

Konstantin Tretyakov (kt@ut.ee)

So far…

- Machine learning is important and interesting
- The general concept:

Fitting models to data

- Some models:

  ?

So far…

- Machine learning is important and interesting
- The general concept:

Fitting models to data

- Some models:
  - Decision trees: if (x < 1) then y = 2, …
  - Linear regression $y = 2x_1 - 3x_2 + 0.5$

Today

- Another model:
  - Linear classification

Linear classification

- **Model type: Binary* classification**

Input: real vectors	Output: class
(1.0, -2.0, 3.1)	"1"
(-2.1, 5.1, 3.0)	"-1"
(0.1, 3,4, -2.0)	"1"

- **Model form:**

$$y = \text{sign}(2x_1 - 3x_2 + x_3 + 0.5)$$

Linear classification

- **Binary* classification**

Linear classification

Binary* classification

Can be generalized to multi-class case:

One-versus-All

All-versus-All

Linear classification

- **Model type: Binary* classification**

Input: real vectors	Output: class
(1.0, -2.0, 3.1)	"1"
(-2.1, 5.1, 3.0)	"-1"
(0.1, 3,4, -2.0)	"1"

- **Model form:**

$$y = \text{sign}(2x_1 - 3x_2 + x_3 + 0.5)$$

Linear classification

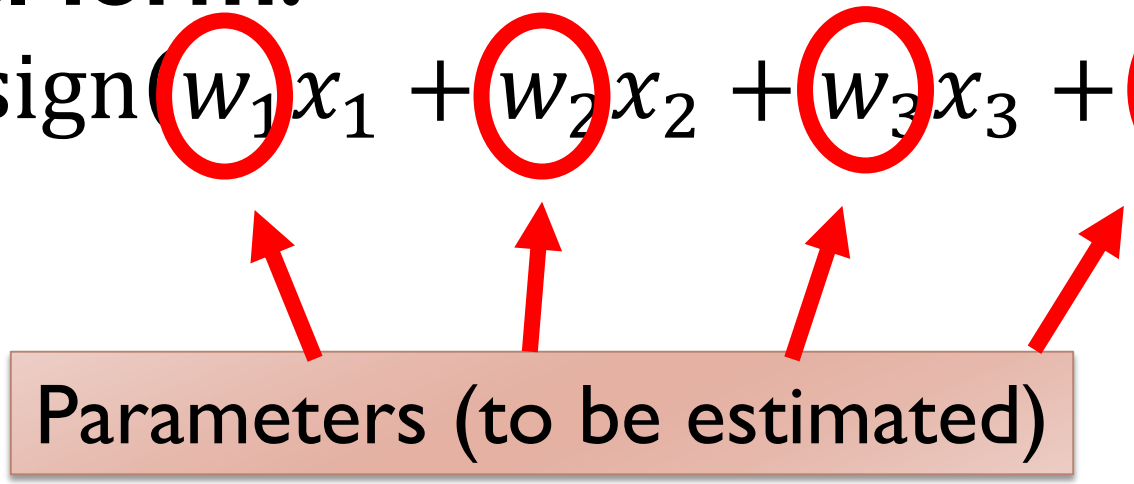
- **Model form:**

$$y = \text{sign}(w_1 x_1 + w_2 x_2 + w_3 x_3 + w_0)$$

Linear classification

- **Model form:**

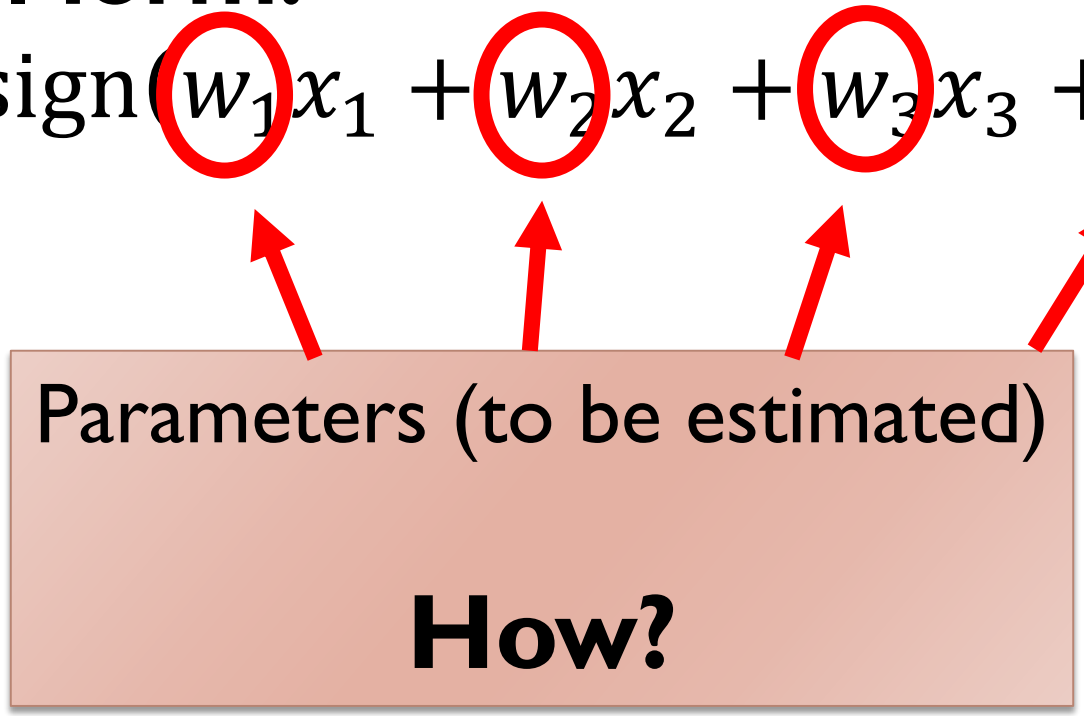
$$y = \text{sign}(w_1 x_1 + w_2 x_2 + w_3 x_3 + w_0)$$



Linear classification

- **Model form:**

$$y = \text{sign}(w_1 x_1 + w_2 x_2 + w_3 x_3 + w_0)$$



Linear classification

- **Parameter learning algorithms:**
  - Fisher discriminant
  - Least-squares (and variations)
  - Perceptron (and variations)
  - Logistic regression
  - Support vector machines
  - 1-norm SVM,
  - Naïve Bayes* …

Linear classification

- **Parameter learning algorithms:**
  - Fisher discriminant

Important

There are **many** parameter-fitting algorithms for the **same** model!

  - 1-norm SVM,
  - Naïve Bayes* …

Practice

- Logistic regression: (NB: $y \in \{0,1\}$)
  - `m = glm(y ~ X, family=binomial)`
  - `predict(m, newX)`

- SVM: (NB: $y \in \{-1,1\}$)
  - `library('e1071')`
  - `m = svm(X, y, kernel='linear')`
  - `predict(m, newX)`

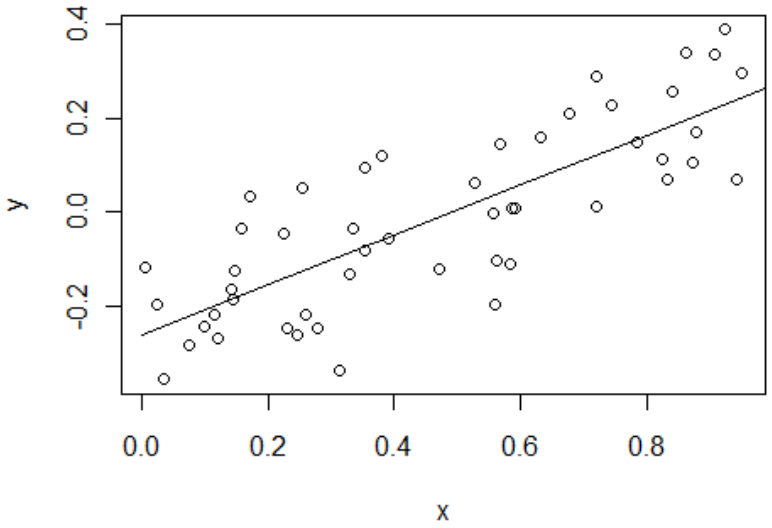
Practice

- Fisher discriminant:
  - `library('MASS')`
  - `m = lda(X, factor(y))`
  - `predict(m, newX)$x`

The Boring Theory

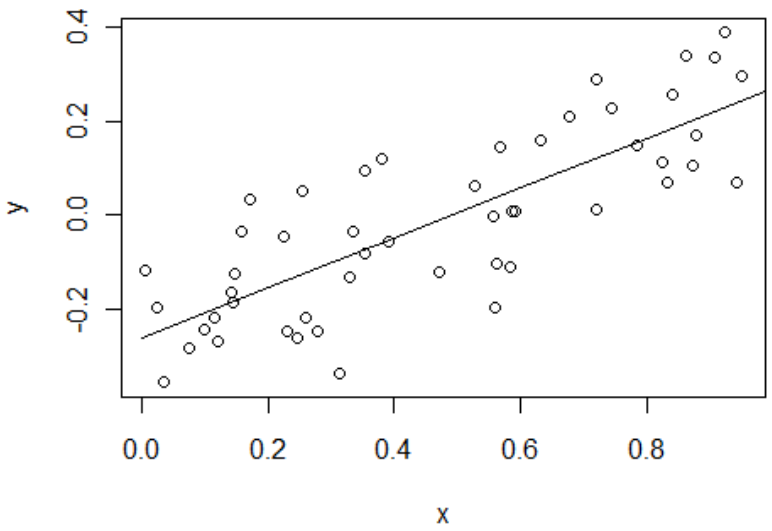
- Algebra & Geometry
- Fisher's Discriminant
- Least-squares approach
- Perceptron
- Other Methods

One-dimensional case

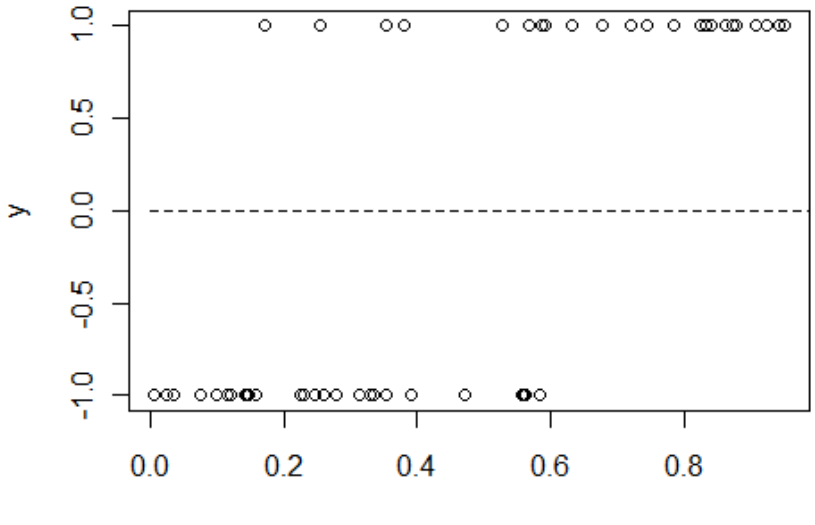


Linear regression

One-dimensional case

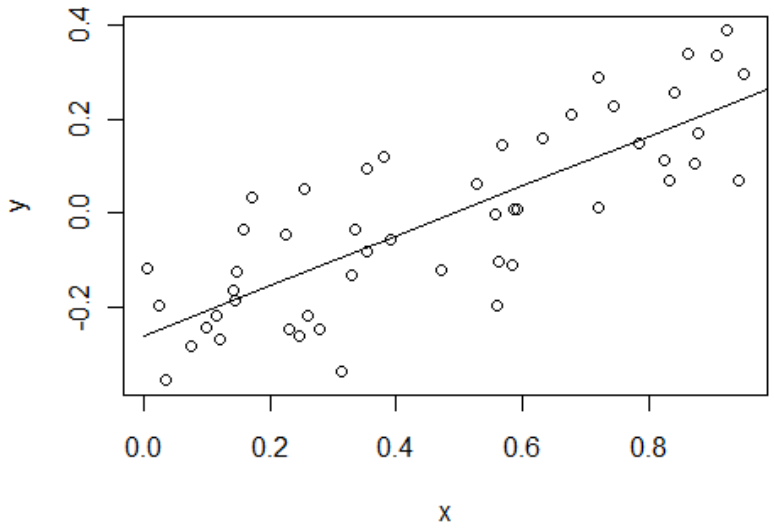


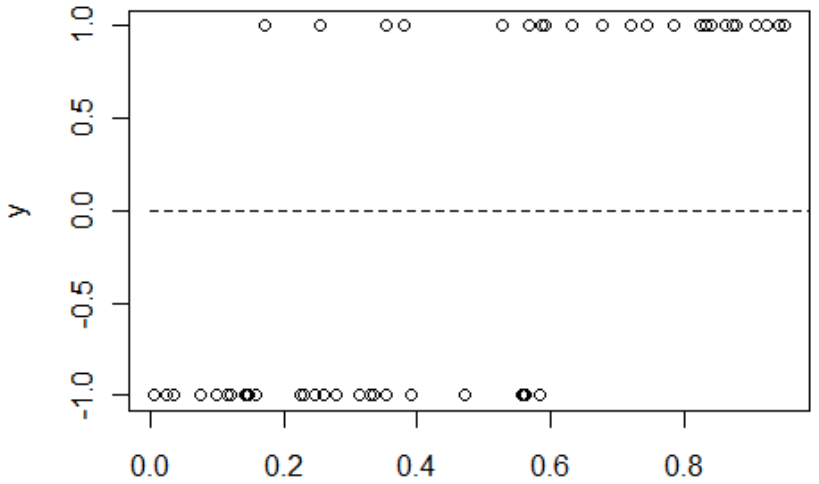
Linear regression



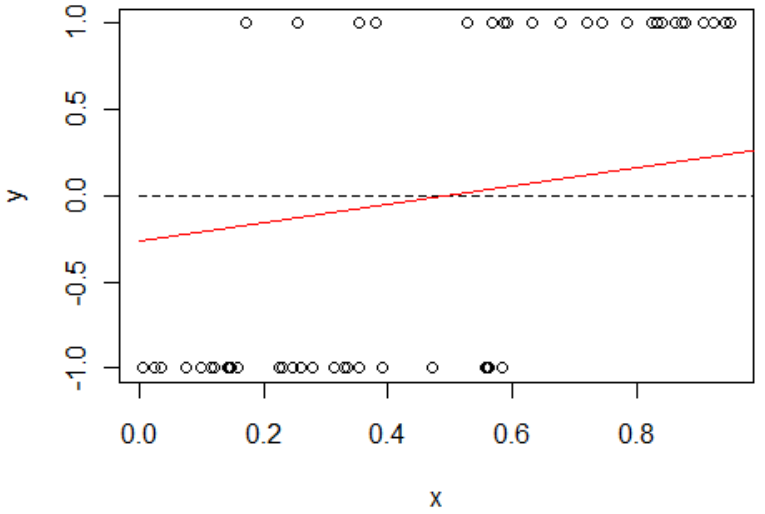
Linear classification

One-dimensional case

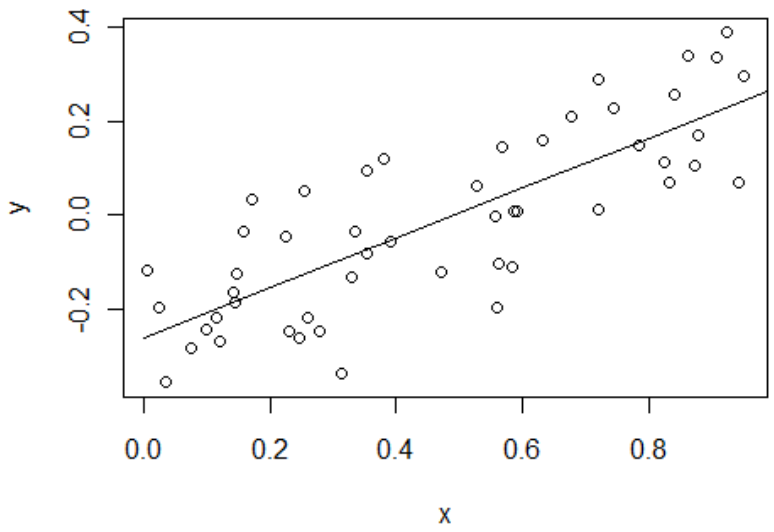


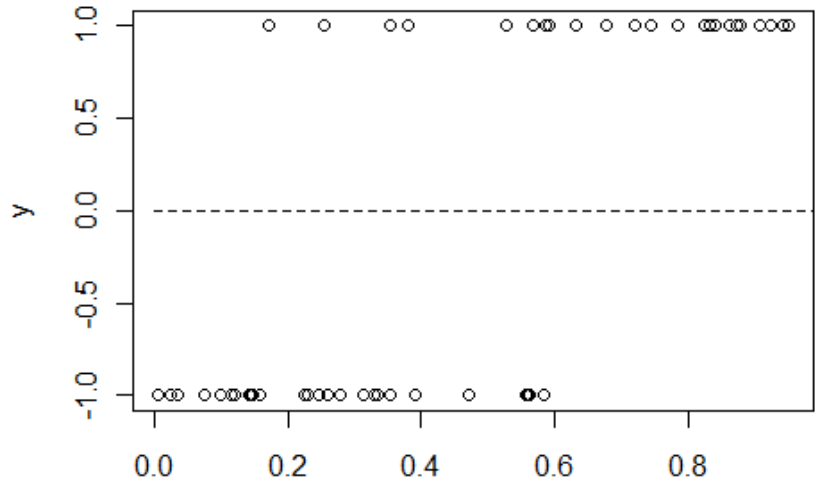


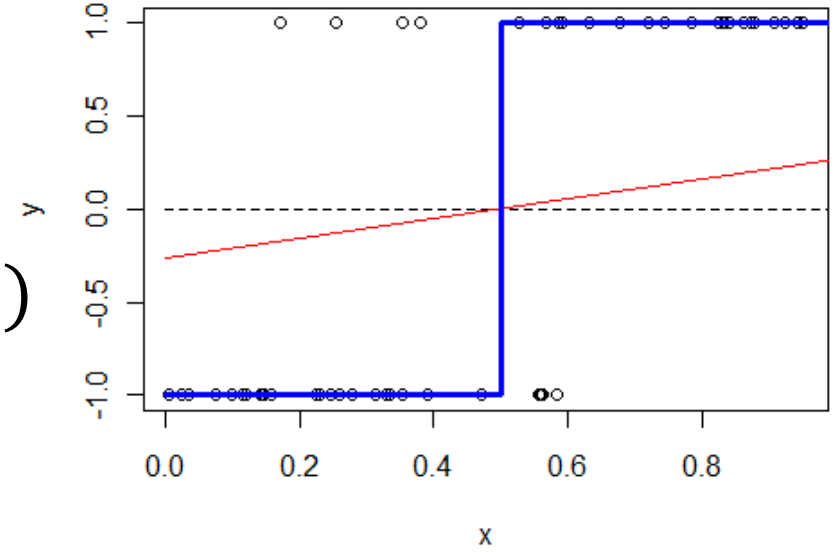
$$y = w_0 + w_1 x$$



One-dimensional case

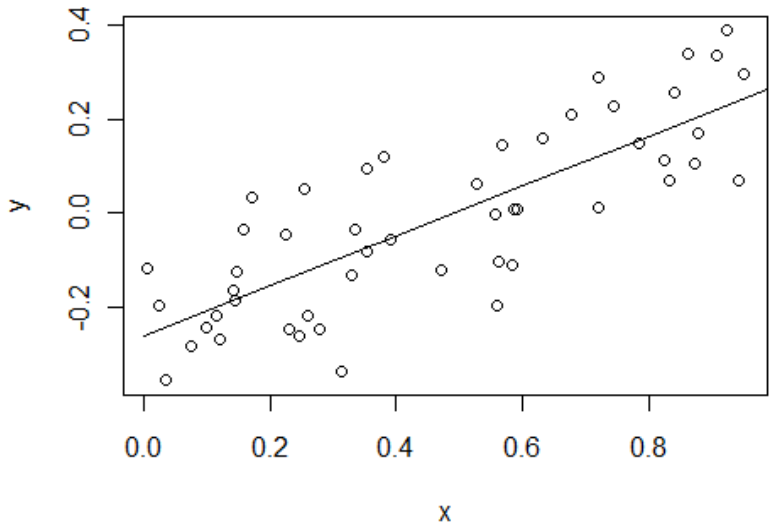


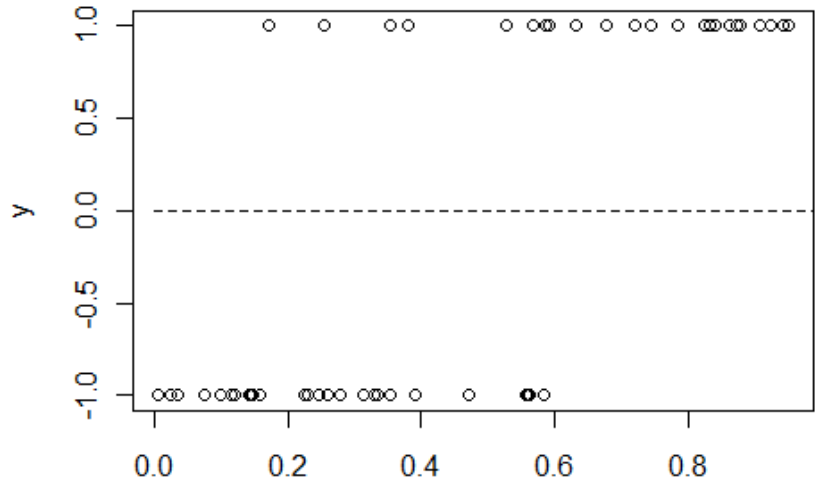




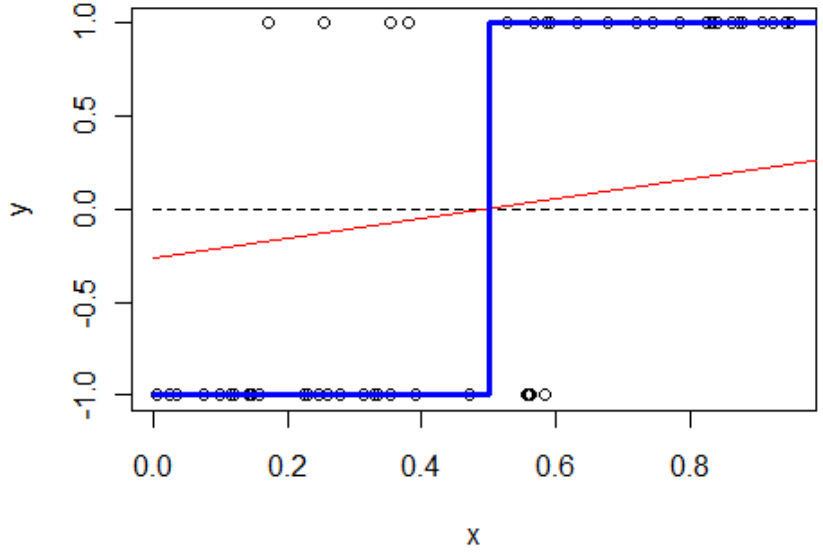
$$y = \text{sign}(w_0 + w_1 x)$$

One-dimensional case



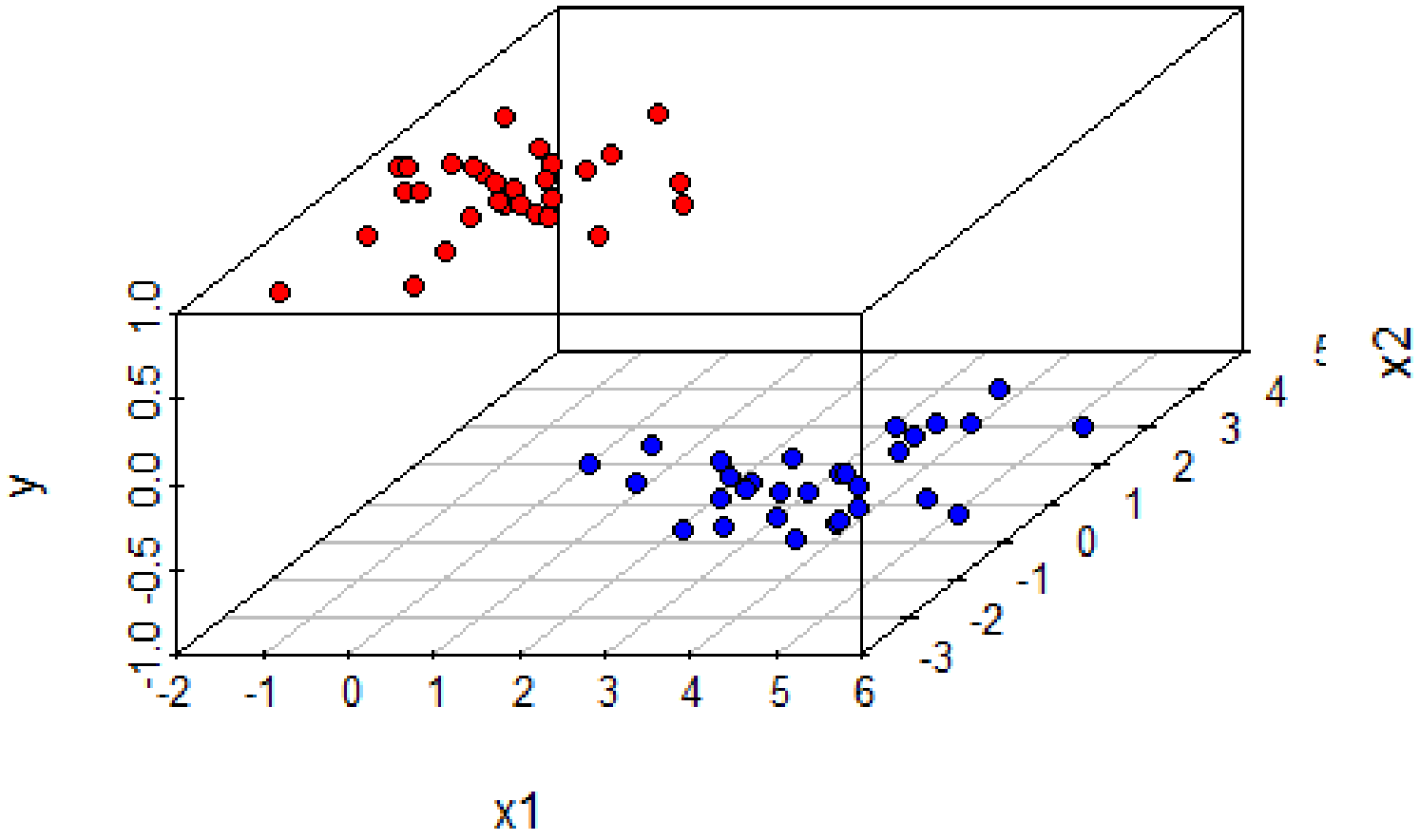


$$y = \begin{cases} 1 & \text{if } x \geq t \\ -1 & \text{otherwise} \end{cases}$$

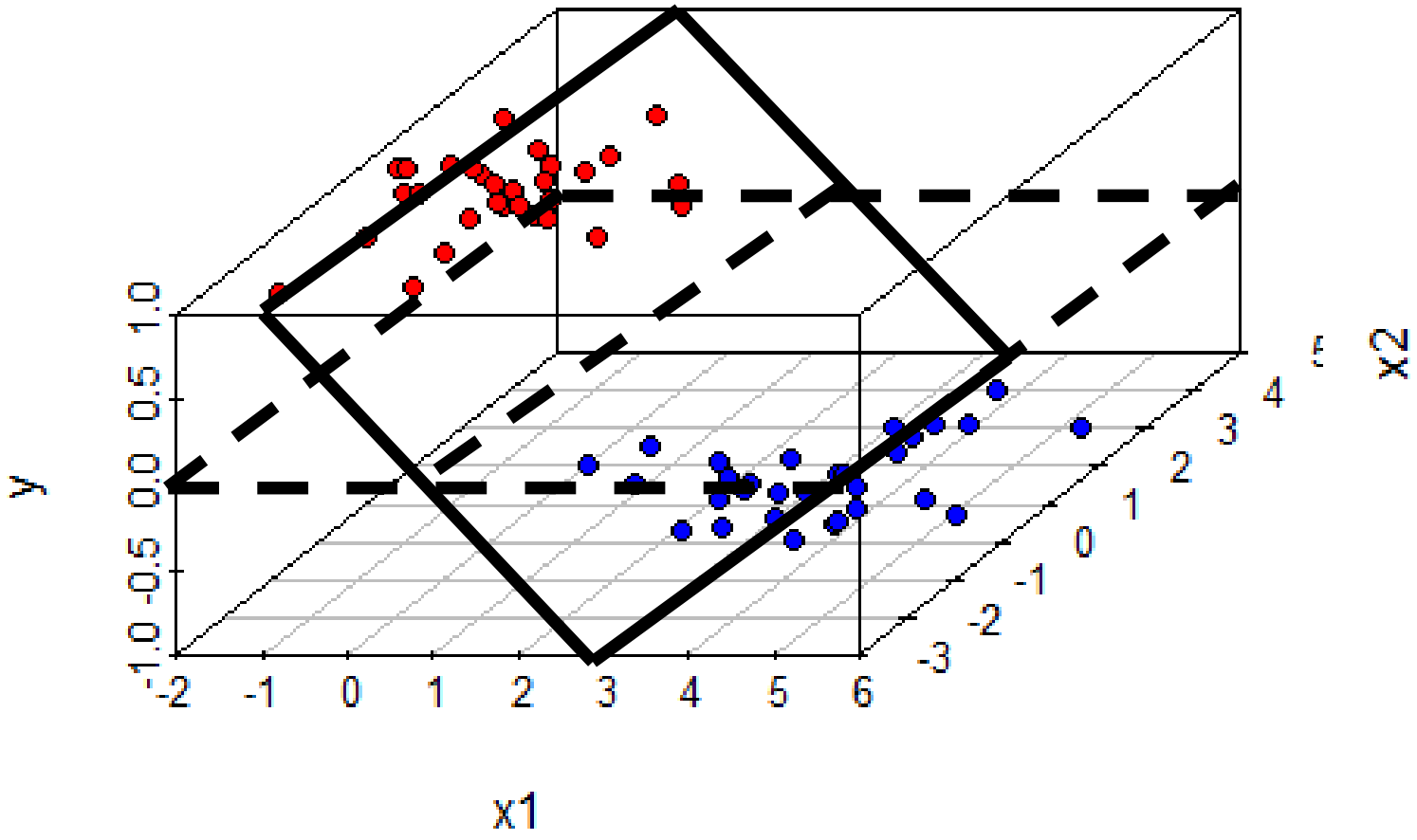


Linear classifier in a one-dimensional case is just a cutoff value.

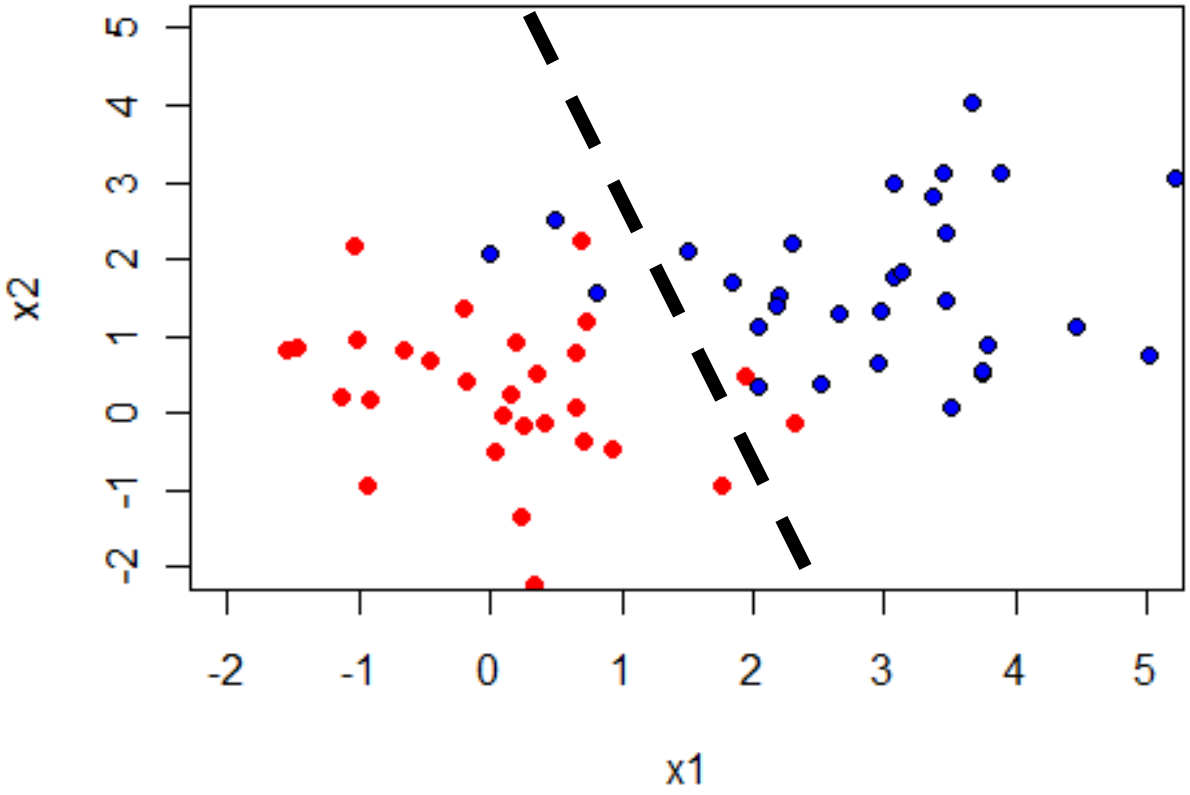
Two-dimensional case



Two-dimensional case

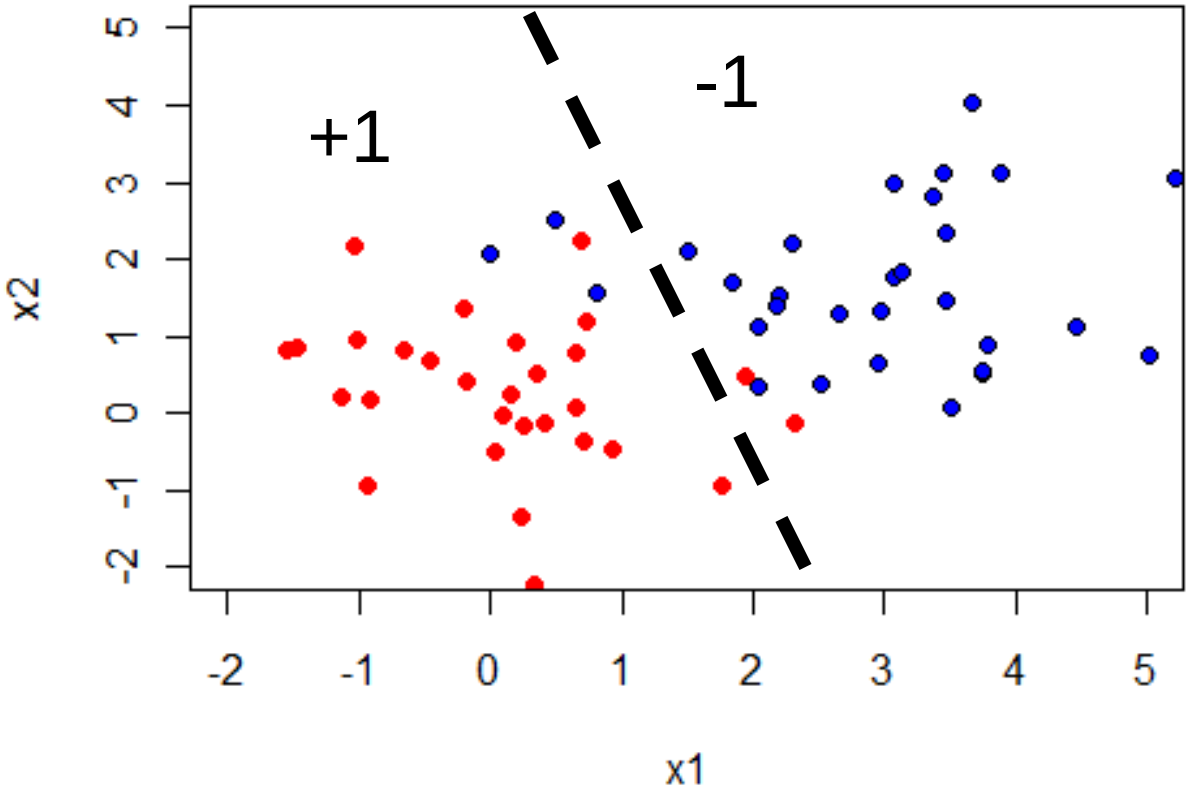


Two-dimensional case



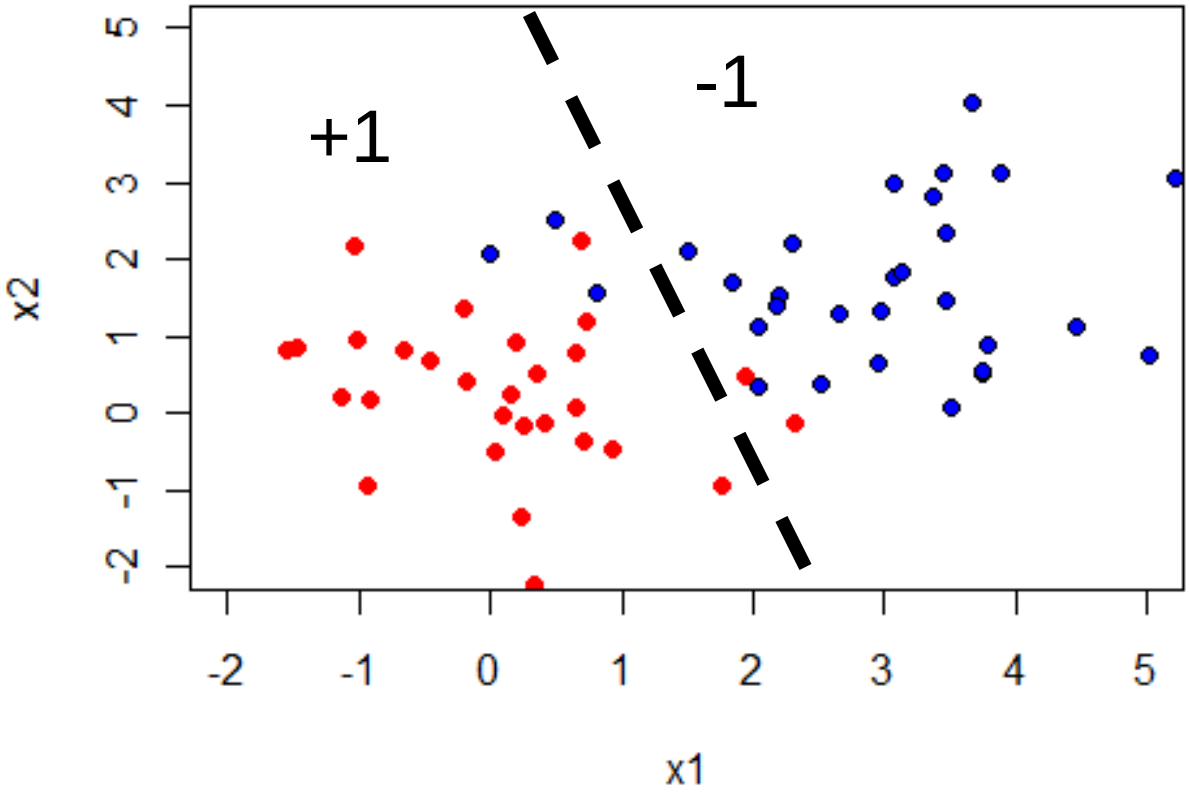
Two-dimensional case

$$f(x_1, x_2) = \text{sign}(w_1 x_1 + w_2 x_2 + w_0)$$



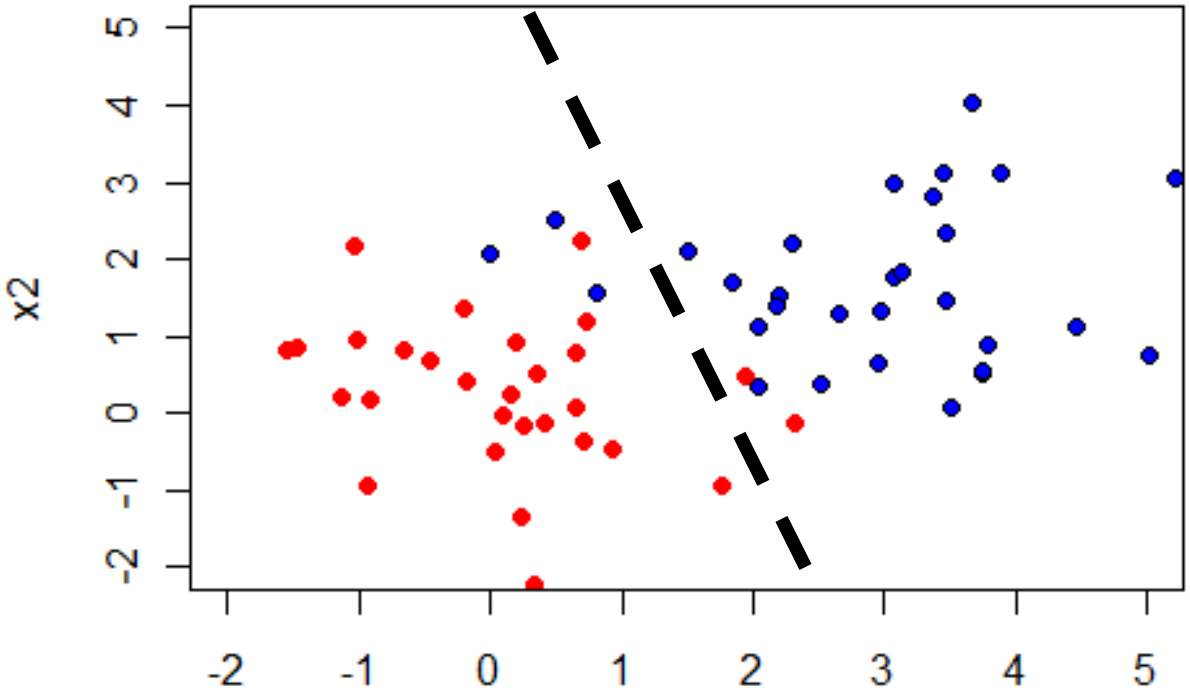
Two-dimensional case

$$f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^T \boldsymbol{x} + w_0)$$

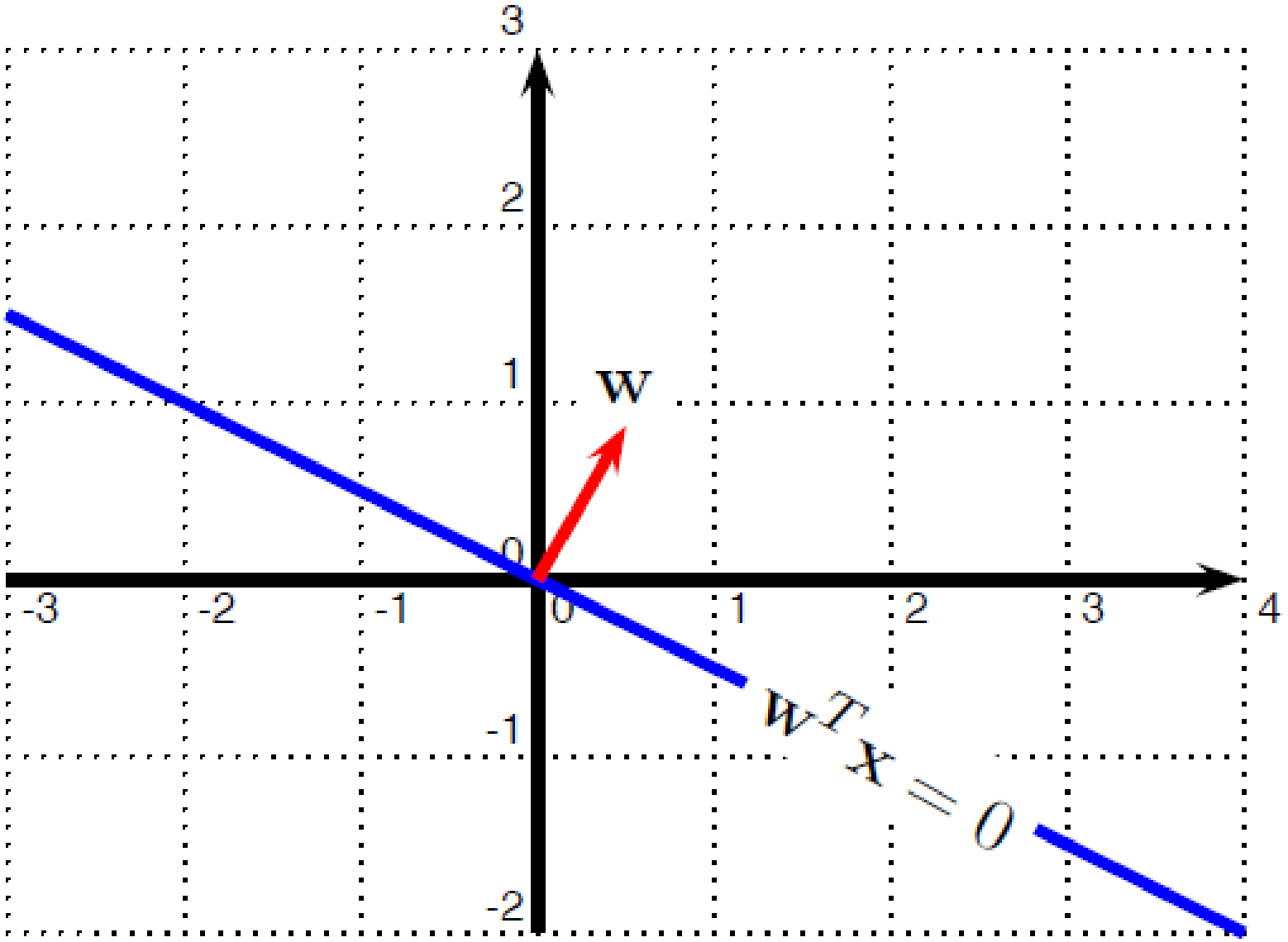


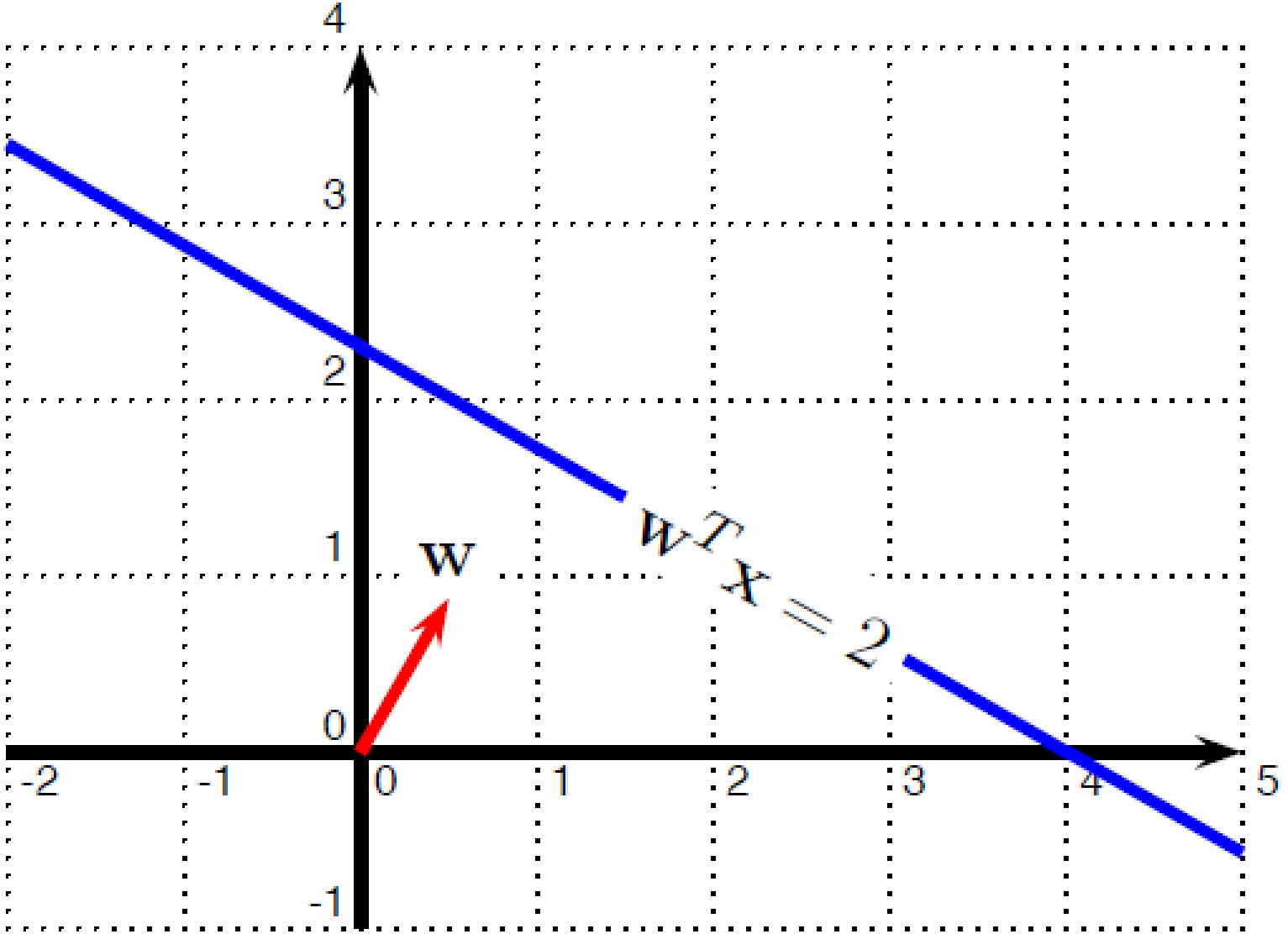
Two-dimensional case

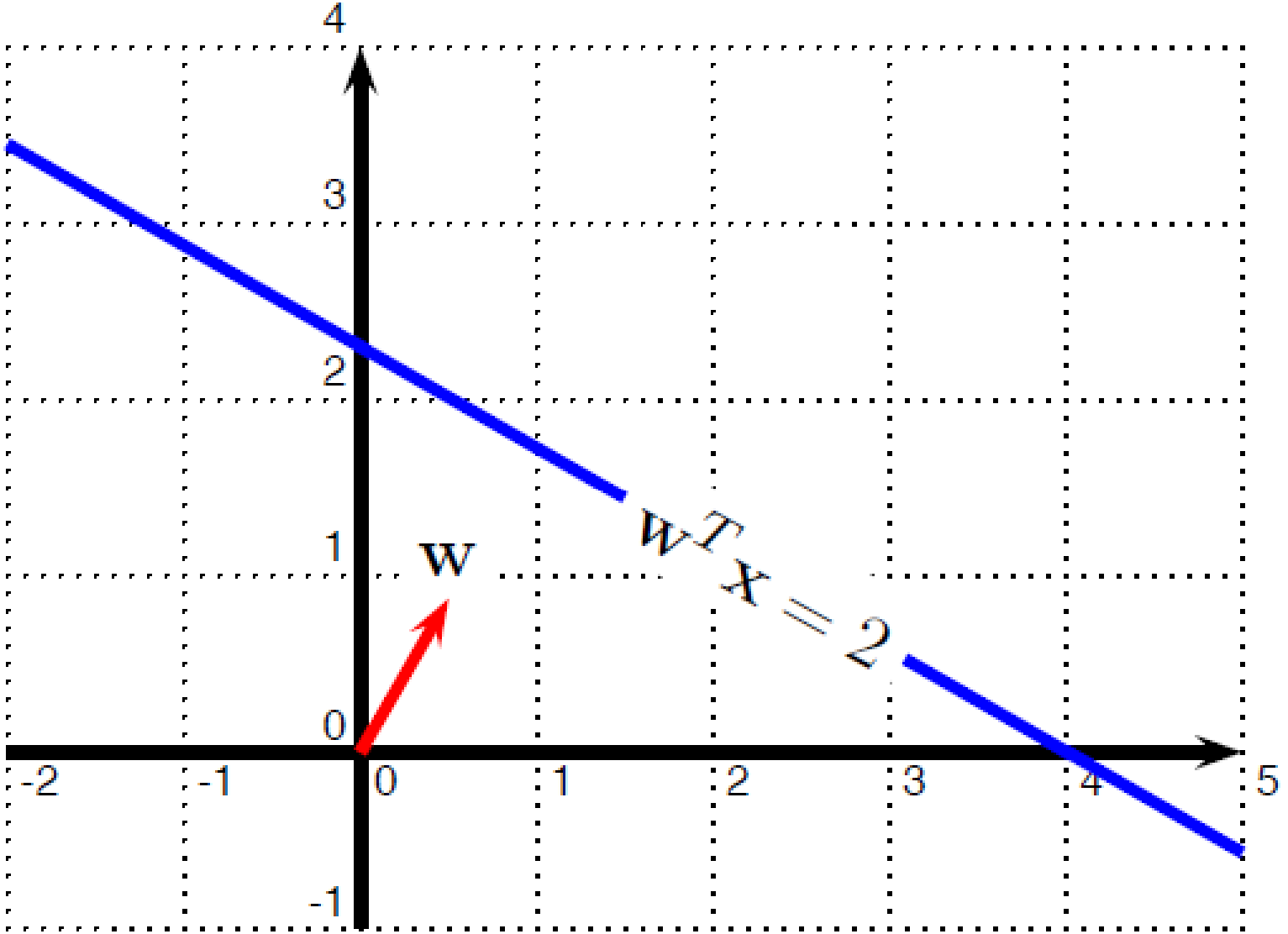
$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$$



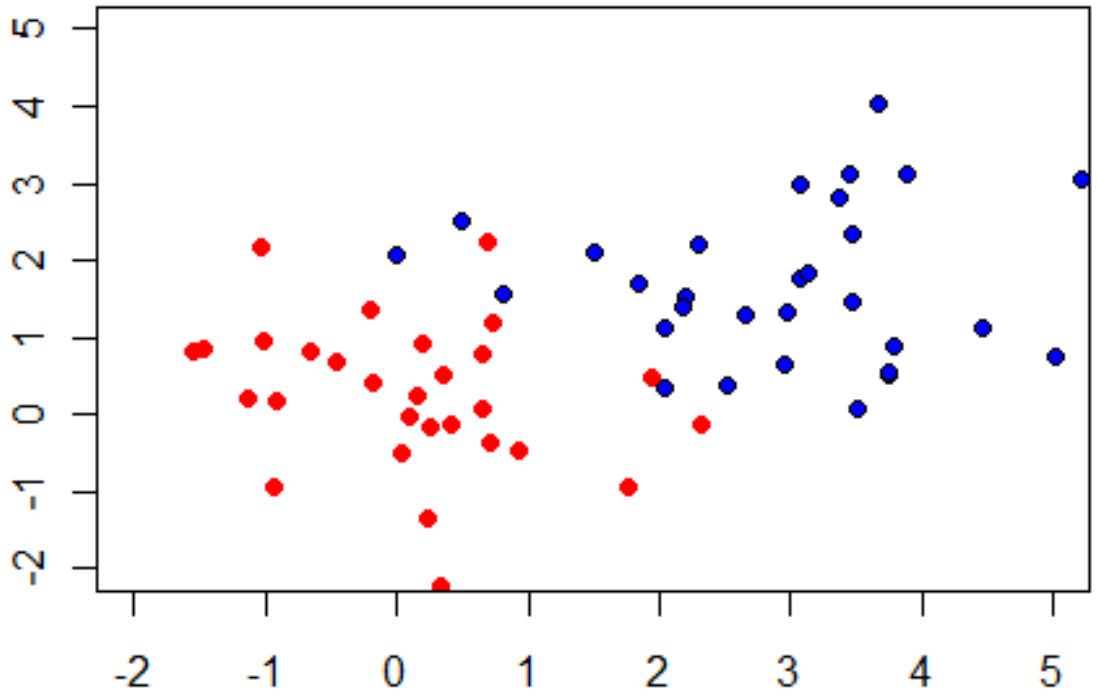
Separating line/plane/hyperplane


3
2
1
0
-1
-2
-3
-2
-1
0
1
2
3
4
w
$w^T x = 0$

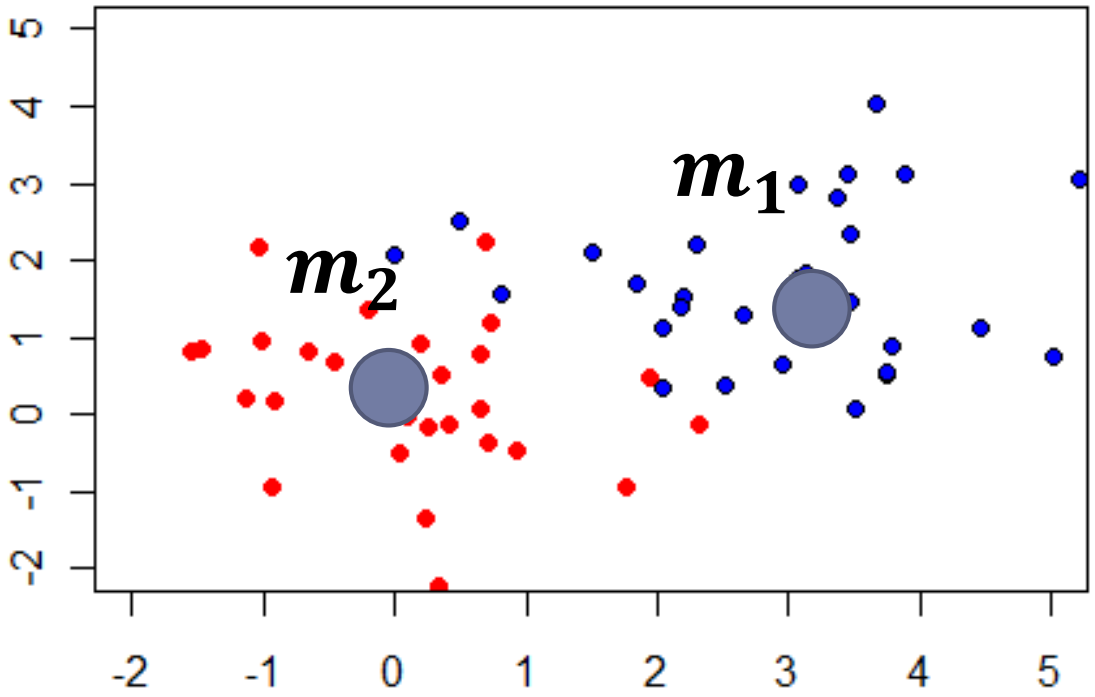

4
3
2
1
0
-1
-2
-1
0
1
2
3
4
5
w
$w^T x = 2$



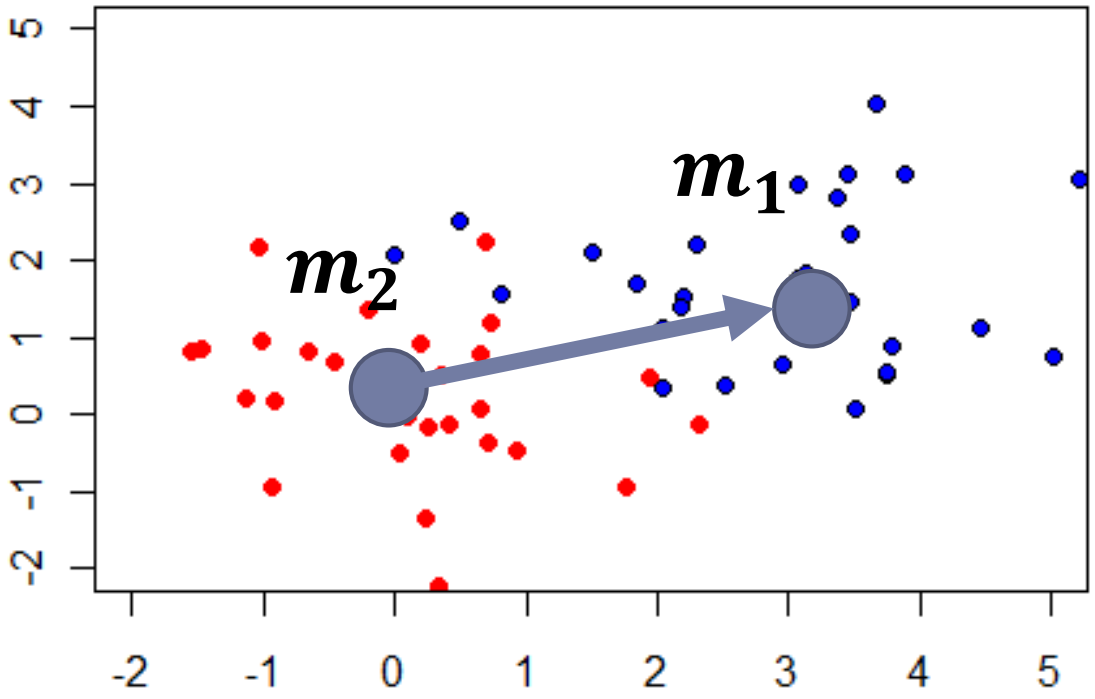
$$\boldsymbol{w^T(x-p)=0}$$


5
4
3
2
1
0
-1
-2
-2
-1
0
1
2
3
4
5

Fisher's discriminant

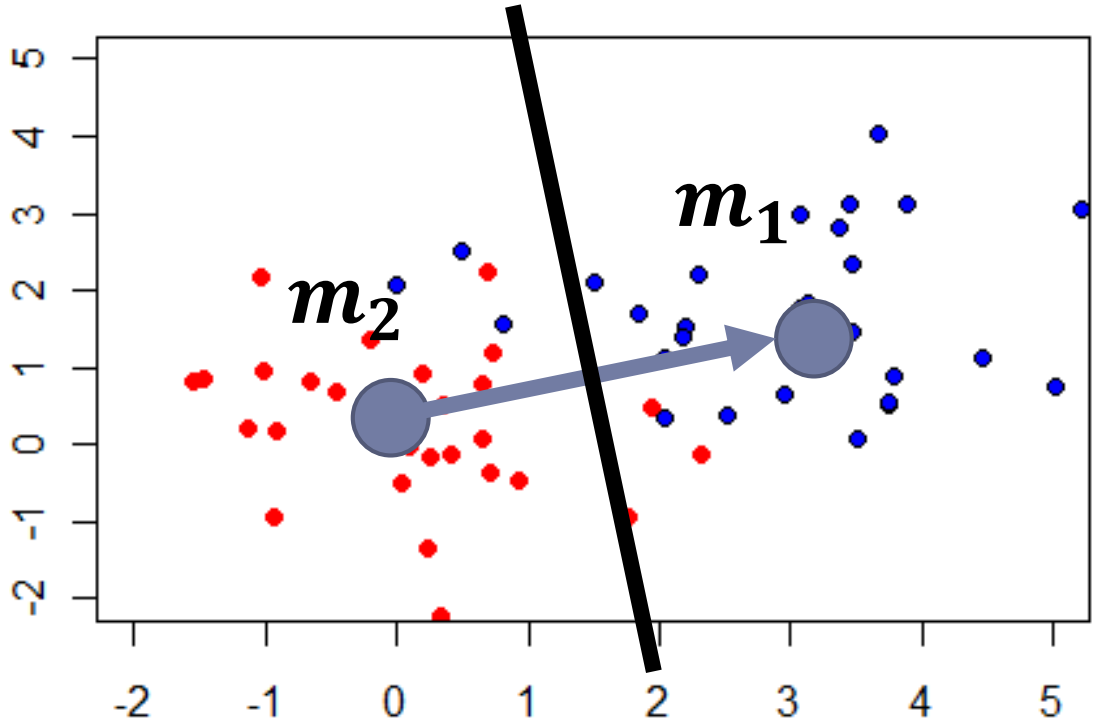


Fisher's discriminant



$$w = m_1 - m_2$$

Fisher's discriminant

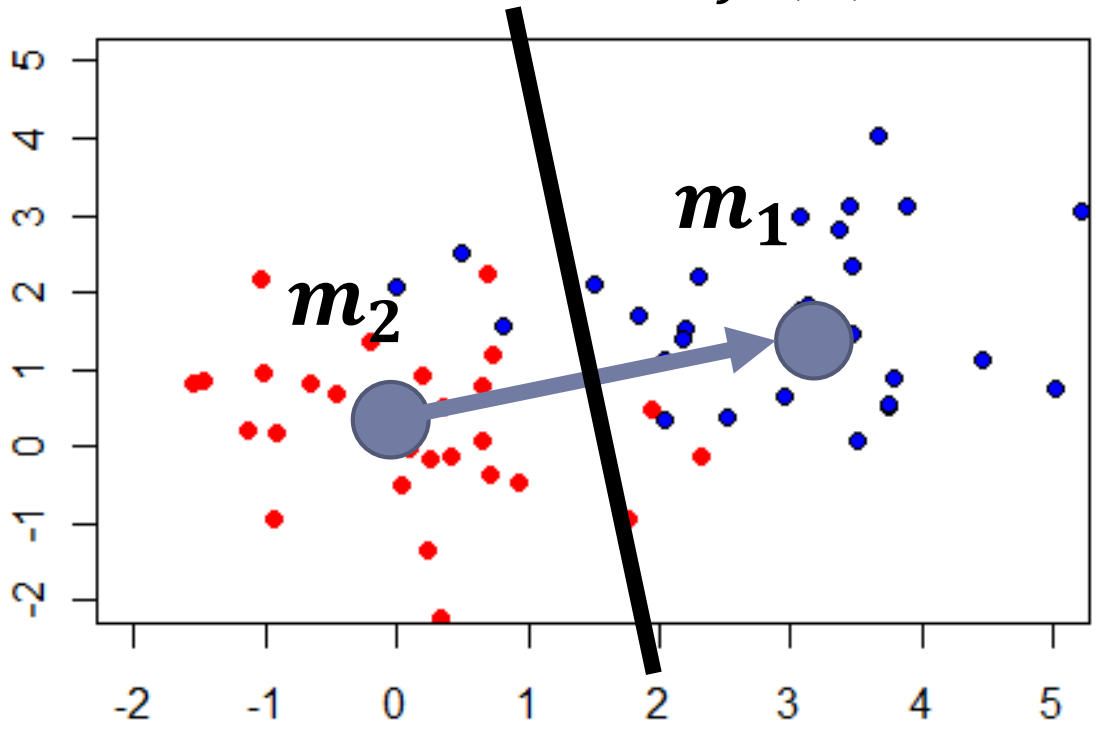


$$w = m_1 - m_2$$

$$p = (m_1 + m_2)/2$$

Fisher's discriminant

$$f(\boldsymbol{x}) = \boldsymbol{w}^T(\boldsymbol{x} - \boldsymbol{p})$$



$$\boldsymbol{w} = \boldsymbol{m_1} - \boldsymbol{m_2}$$

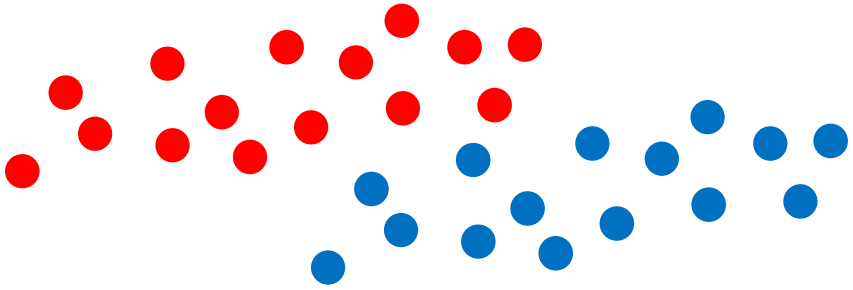
$$\boldsymbol{p} = (\boldsymbol{m_1} + \boldsymbol{m_2})/2$$

Fisher's discriminant

- Any problems with this approach?

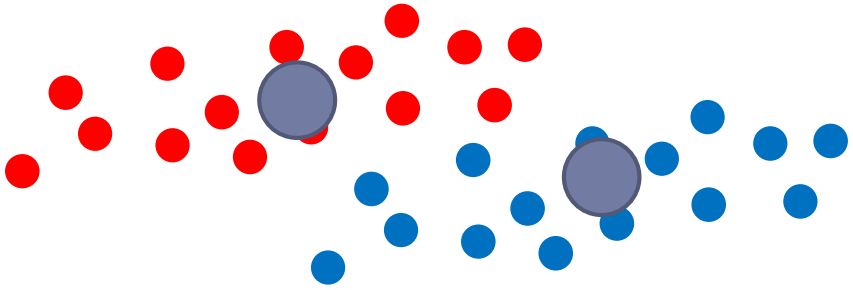
Fisher's discriminant

- Any problems with this approach?



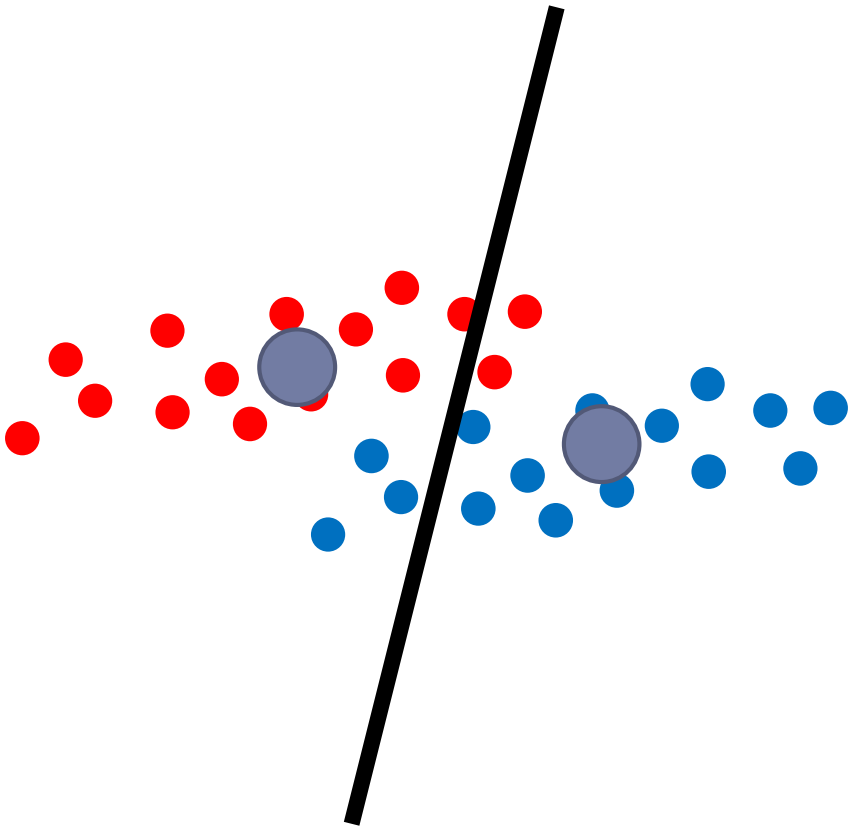
Fisher's discriminant

- Any problems with this approach?



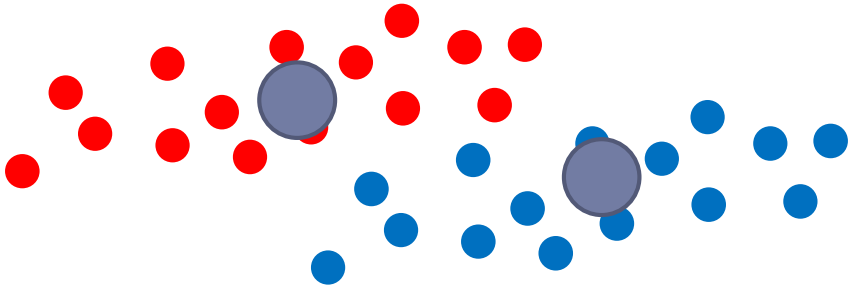
Fisher's discriminant

- Any problems with this approach?



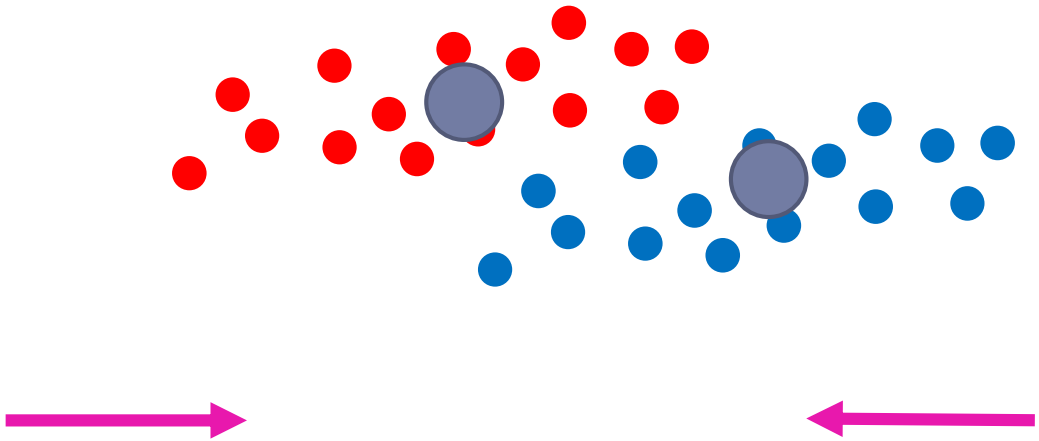
Fisher's discriminant

- Any problems with this approach?



Fisher's discriminant

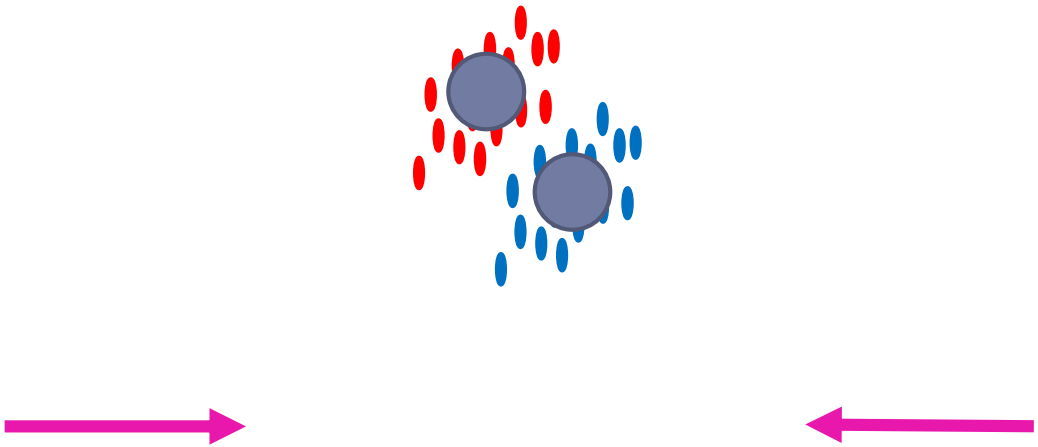
- Any problems with this approach?



Therefore, we augment the idea by first **transforming the space** to ensure **maximally "circular" shape** of the two point clouds.

Fisher's discriminant

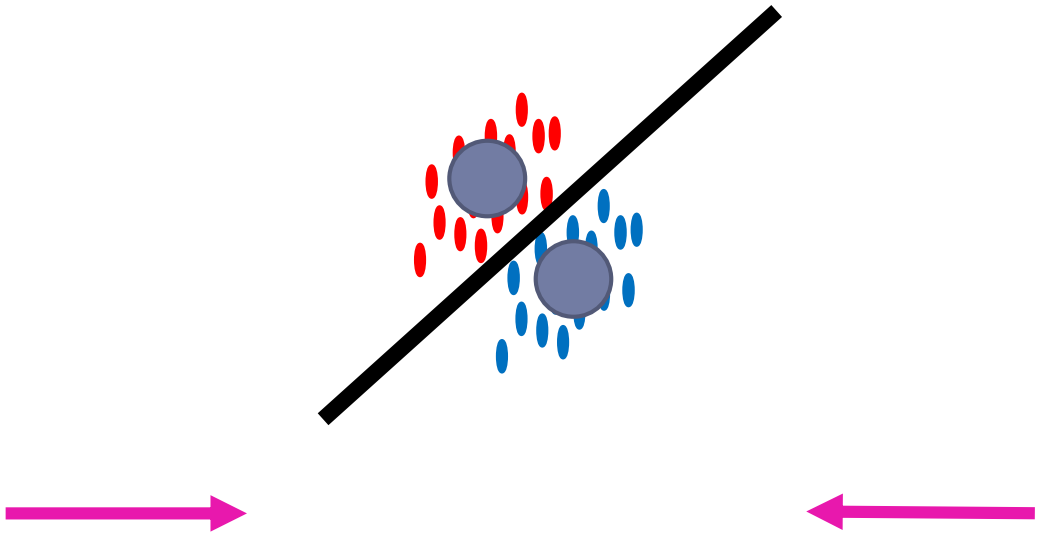
- Any problems with this approach?



Therefore, we augment the idea by first **transforming the space** to ensure **maximally "circular" shape** of the two point clouds.

Fisher's discriminant

- Any problems with this approach?



Therefore, we augment the idea by first **transforming the space** to ensure **maximally "circular" shape** of the two point clouds.

Fisher's discriminant

After doing some math, the estimation rule turns out to be the following:
(we shall discuss the derivation on the practice session)

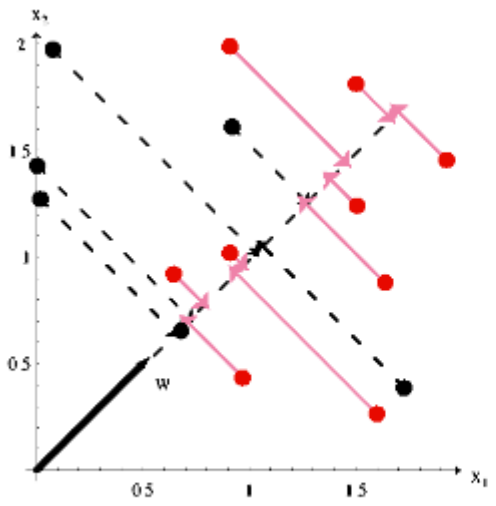
$$\mathbf{w} = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2).$$

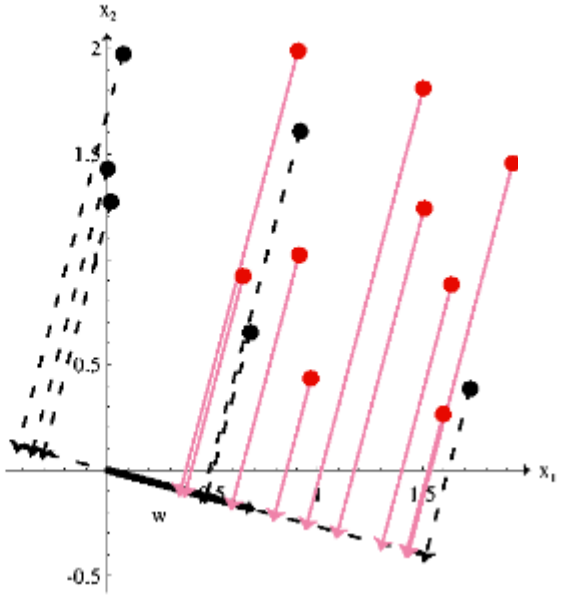
$$\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2. \qquad \mathbf{S}_i = \sum_{\mathbf{x} \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^t$$

$$w_0 = -\mathbf{w}^{\mathbf{T}}(\mathbf{m_1} + \mathbf{m_2})/2$$

Other Interpretations of the FD

- **Probabilistic interpretation**
  - Asymptotically optimal classifier for two normally-distributed classes with equal covariance.
- **Optimizing the Fisher's criterion**





$$J(\mathbf{w}) = \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

The Boring Theory

- ~~Algebra & Geometry~~
- ~~Fisher's Discriminant~~

$$\mathbf{w} = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2).$$

- Least-squares approach
- Perceptron
- Other Methods

The "bias term" convention

$$f(\boldsymbol{x}) = w_0 + \boldsymbol{w}^T \boldsymbol{x}$$

is equivalent to

$$f(\boldsymbol{x}') = \boldsymbol{w}'^T \boldsymbol{x}'$$

where $\boldsymbol{x}'$ and $\boldsymbol{w}'$ are *augmented* vectors:

$$\boldsymbol{x}' = \begin{pmatrix} 1 \\ \boldsymbol{x} \end{pmatrix} \quad \boldsymbol{w}' = \begin{pmatrix} w_0 \\ \boldsymbol{w} \end{pmatrix}$$

The "bias term" convention

$$f(\boldsymbol{x}) = w_0 + \boldsymbol{w}^T \boldsymbol{x}$$

is equivalent to

NB: Many algorithms (e.g. Fisher's discriminant) treat the bias term in a special way!

where $\boldsymbol{x}'$ and $\boldsymbol{w}'$ are *augmented* vectors:

$$\boldsymbol{x}' = \begin{pmatrix} 1 \\ \boldsymbol{x} \end{pmatrix} \quad \boldsymbol{w}' = \begin{pmatrix} w_0 \\ \boldsymbol{w} \end{pmatrix}$$

LSE Methods

- Recall linear regression:

$$\boldsymbol{w} =$$

LSE Methods

- Recall linear regression:

$$\boldsymbol{w} = \left(\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{y}$$

LSE Methods

- Recall linear regression:

$$\boldsymbol{w} = \boldsymbol{X}^{+}\boldsymbol{y}$$

LSE Methods

- Recall linear regression:

$$\boldsymbol{w} = \boldsymbol{X}^{+}\boldsymbol{y}$$

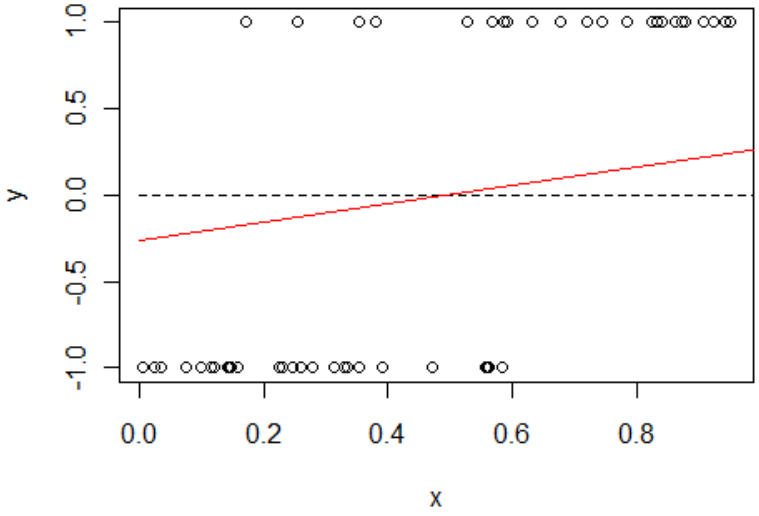
Why not apply the same idea directly to solve the classification tasks?

LSE Methods

- Recall linear regression:

$$\boldsymbol{w} = \boldsymbol{X}^{+}\boldsymbol{y}$$

Why not apply the same idea directly to solve the classification tasks?



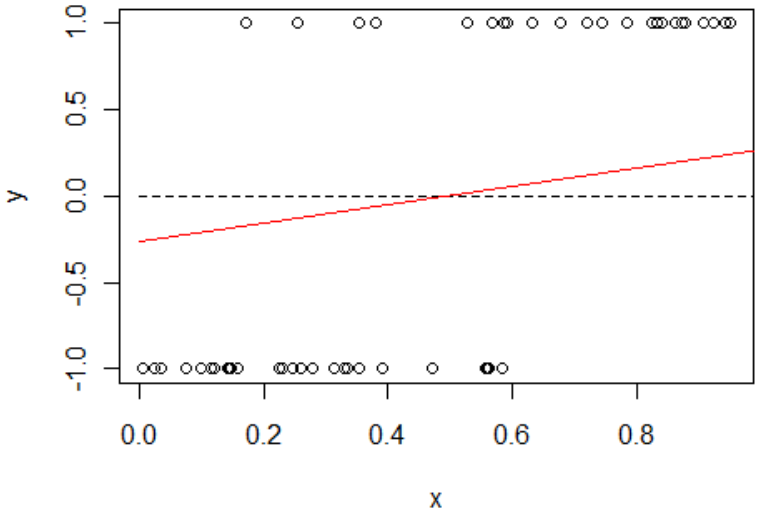
It makes no sense!

LSE Methods

- Recall linear regression:

$$\boldsymbol{w} = \boldsymbol{X}^{+}\boldsymbol{y}$$

Why not apply the same idea directly to solve the classification tasks?



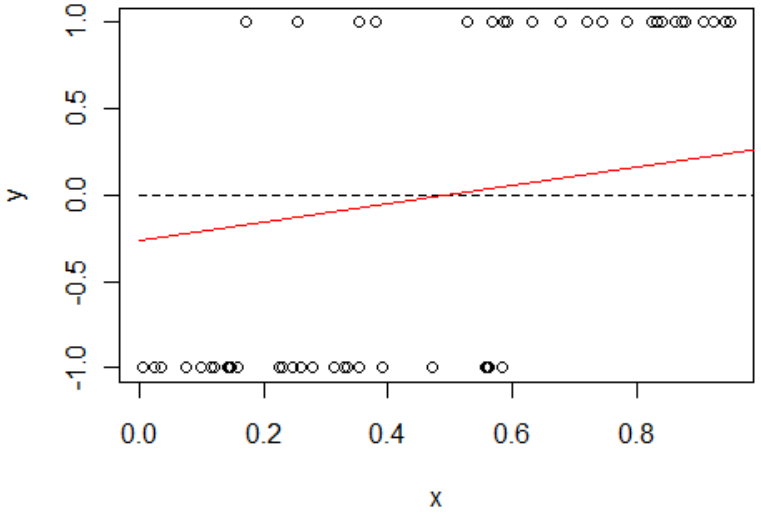
It is too sensitive to outliers

LSE Methods

- Recall linear regression:

$$\boldsymbol{w} = \boldsymbol{X}^{+}\boldsymbol{y}$$

Why not apply the same idea directly to solve the classification tasks?



… but it works!*

*sometimes

LSE Methods – Why?

Suppose there exists some good classifier

$$y_{\mathrm{i}} = \mathrm{sign}\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}_i\right),$$

LSE Methods – Why?

Suppose there exists some good classifier

$$y_{\mathrm{i}} = \mathrm{sign}\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}_i\right),$$

If we knew the *margin* that the classifier has at each point:

$$y'_i = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}_{\boldsymbol{i}}$$

we could recover $\boldsymbol{w}$ by fitting a linear regression on $(y'_i, \boldsymbol{x}_{\boldsymbol{i}})$.

$$\boldsymbol{w} = \boldsymbol{X}^{+}\boldsymbol{y}'$$

LSE Methods – Why?

Suppose there exists some good classifier

$$y_i = \text{sign}\left(\boldsymbol{w}^T \boldsymbol{x}_i\right),$$

Unfortunately, we do not know the margins.

LSE Methods – Examples

- It is possible to show that if we pick margins y_i' as follows:

$$y'_i = y_i / P(y_i)$$

where $P(c) =$ proportion of elements of class c in the sample,

then LSE result **is equivalent to the Fisher's discriminant**.

Ho-Kashyap Algorithm

- It is possible to iteratively search for the margin vector, optimizing the error

$$\left|\left|\boldsymbol{Xw} - \boldsymbol{y}'\right|\right|^2$$

with the condition that components of $\boldsymbol{y}'$ are *never decreased* in absolute value.

$$\boldsymbol{y}'_0 := \boldsymbol{rand} \cdot \boldsymbol{y}$$
$$\boldsymbol{\Delta y}'_i := \mu\left((\boldsymbol{Xw} - \boldsymbol{y}'_i) \cdot \boldsymbol{y}\right)^+ \cdot \boldsymbol{y}$$

The Boring Theory

- ~~Algebra & Geometry~~
- ~~Fisher's Discriminant~~

$$\mathbf{w} = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2).$$

- ~~Least-squares approach~~

$$\mathbf{w} = \mathbf{X}^{+}\mathbf{y}$$

- Perceptron
- Other Methods

Perceptron

- Consider the error function:

$$\sum -y_i \, \boldsymbol{w}^T \boldsymbol{x}_i$$

where the sum is taken **over all misclassified examples**.

- Devise a batch and stochastic gradient-descent optimization procedures.

Perceptron

- Consider the error function:

$$\sum -y_i \, \boldsymbol{w}^T \boldsymbol{x}_i$$

where the sum is taken **over all misclassified examples**.

- **Batch perceptron**

$$\boldsymbol{\Delta w} = \mu \sum y_i \boldsymbol{x}_i$$

(sum over misclassified examples)

Perceptron

- Consider the error function:

$$\sum -y_i \, \boldsymbol{w}^T \boldsymbol{x}_i$$

where the sum is taken **over all misclassified examples**.

- **Stochastic (classic) perceptron**

$$\boldsymbol{\Delta w} = \mu \, y_i \boldsymbol{x}_i$$

(for randomly picked misclassified example)

Perceptron properties

- Always converges for linearly-separable data (independently of the step-size!)
- Never converges for non-separable data.
- μ does not matter, but smart choices are possible. E.g. *relaxation to margin*:

$$\mu = \frac{m - \boldsymbol{w}^T \boldsymbol{x_i} y_i}{\left|\left|\boldsymbol{x_i}\right|\right|^2}$$

The Boring Theory

- ~~Algebra & Geometry~~
- ~~Fisher's Discriminant~~ $\mathbf{w} = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2).$
- ~~Least-squares approach~~ $\mathbf{w} = \mathbf{X}^{+}\mathbf{y}$
- ~~Perceptron~~ $\Delta\mathbf{w} = \mu\, y_i \mathbf{x_i}$
- Other Methods

Other methods

- Logistic regression:

$$\sum_i y_i \log p_{\boldsymbol{w}}(\boldsymbol{x_i}) + (1 - y_i)\log(1 - p_{\boldsymbol{w}}(\boldsymbol{x_i}))$$

- SVM:

$$\sum_i \text{hinge\_loss}(y_i, \boldsymbol{w}_i^T \boldsymbol{x_i}) + \lambda \left|\left|\boldsymbol{w_i}\right|\right|^2$$

- …

The Boring Theory

- ~~Algebra & Geometry~~
- ~~Fisher's Discriminant~~ $\mathbf{w} = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2).$
- ~~Least-squares approach~~ $\mathbf{w} = \mathbf{X}^{+}\mathbf{y}$
- ~~Perceptron~~ $\Delta\mathbf{w} = \mu\, y_i \mathbf{x_i}$
- ~~Other Methods~~ other objective funcs

Quiz

- How many linear classification algorithms were mentioned in the lecture?
- How many of them could you implement?
- How many of them could you use from R?