CS4801 : Classification

Sahely Bhadra
16/8/2017

1. Introduction to Classification
2. Nearest Neighbour Classification
3.Bayes decision rule
   1. Classification error
   2. Minimum error rate classification
      1. Two category
      2. Multi category

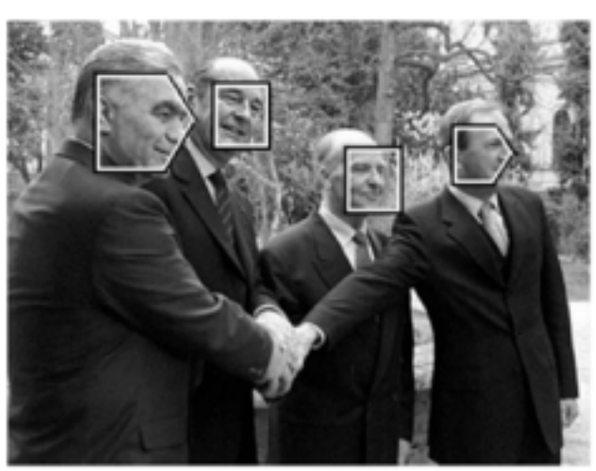
Classification

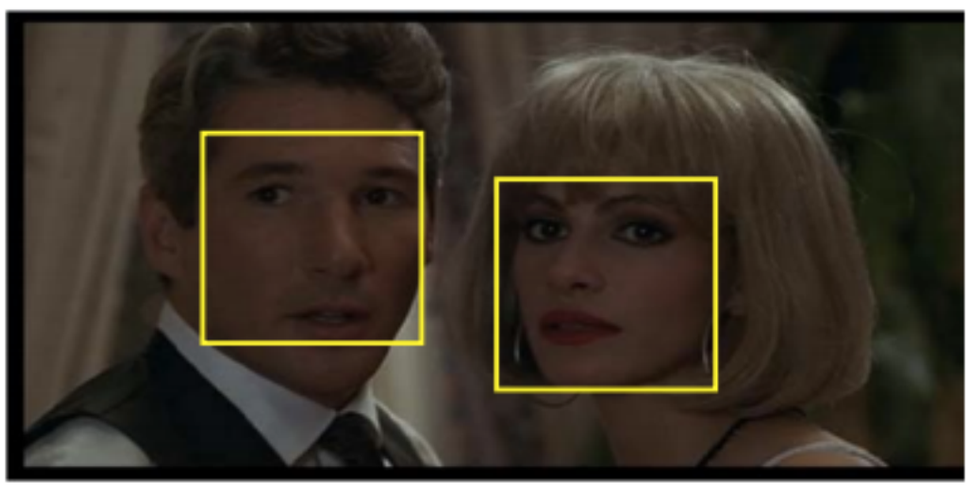
- In classification problems, each entity in some domain can be placed in one of a discrete set of categories: yes/no, friend/foe, good/bad/indifferent, blue/red/green, etc.
- Given a training set of labeled entities, develop a rule for assigning labels to entities in a test set
- Many variations on this theme:
  - binary classification
  - multi-category classification
- Many criteria to assess rules and their predictions
  - overall errors
  - costs associated with different kinds of errors

Dataset

- Each training data point to be classified is represented as a pair (x, y):
  - where x is a description of the object : f**eature vector**
  - where y is a **label** (assumed binary for now $y = \{+1, -1\}$)
- Success or failure of a machine learning classifier often depends on **choosing good set of features or descriptions of objects**
  - the choice of description can also be viewed as a learning problem
  - but good human intuitions are often needed here

Binary classification

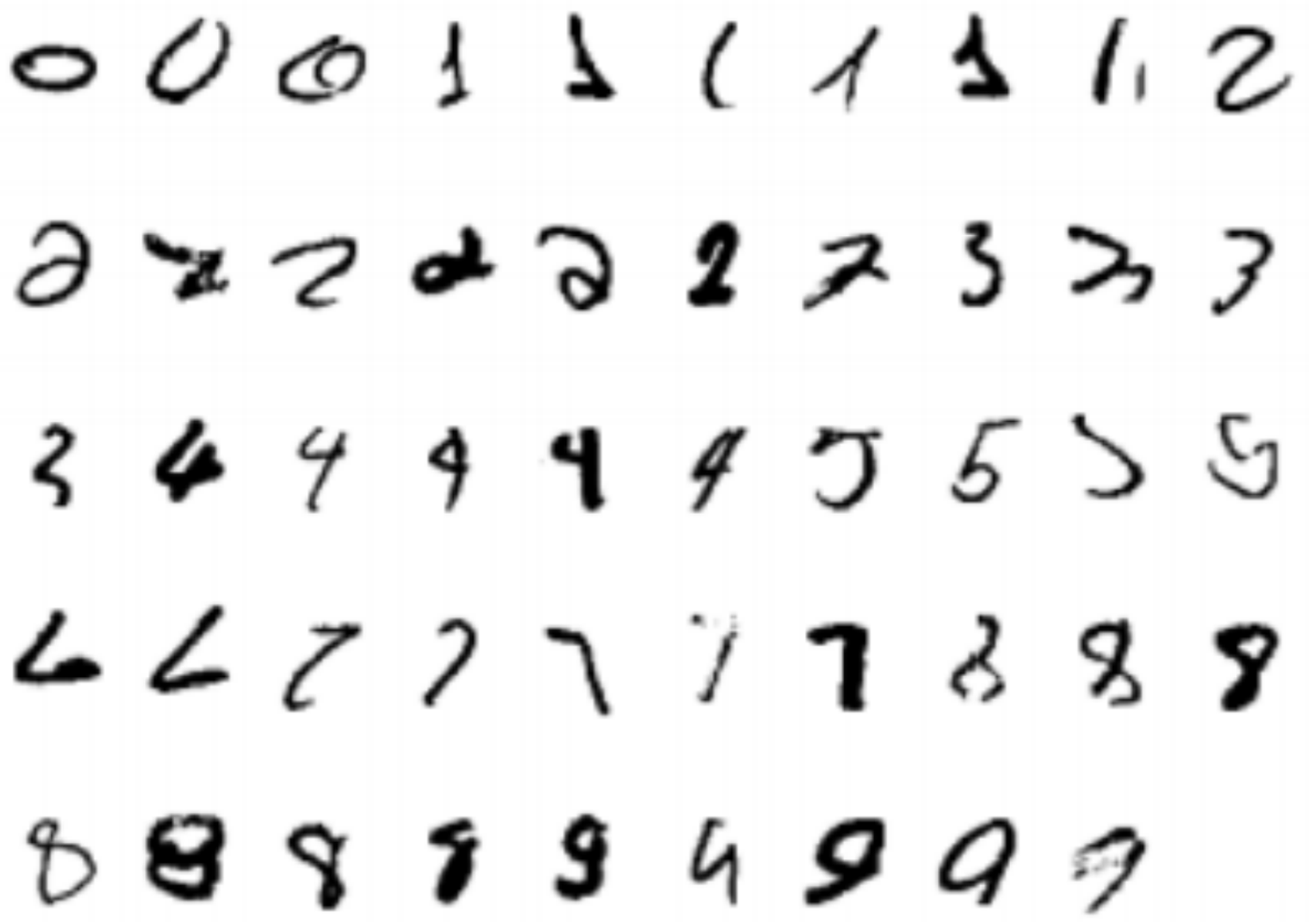




Face detection:

1.Facial part
2.Non facial part

Multiclass classification



Hand written digit classification

Class label : [0,1,2,…,9]

Examples of classifications

- Fraud detection (input: account activity, classes: fraud / no fraud)
- Web page spam detection (input: HTML/rendered page, classes: spam / ham)
- Speech recognition and speaker recognition (input: waveform, classes: phonemes or words)
- Medical diagnosis (input: symptoms, classes: diseases)
- Automatic essay grader (input: document, classes: grades)
- Customer service email routing and foldering
- Link prediction in social networks
- Catalytic activity in drug design
- … many many more

- Classification is an important commercial technology

State-of-art Classifiers

I) Instance-based methods:
   1) Nearest neighbour

II) Probabilistic models:
   1) Naïve Bayes
   2) Logistic Regression

III) Linear Models:
   1) Perceptron
   2) Support Vector Machine

IV) Decision Models:
   1) Decision Trees
   2) Boosted Decision Trees
   3) Random Forest

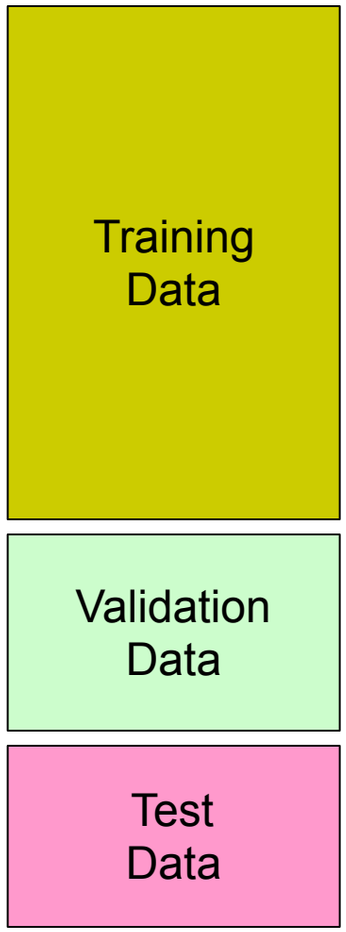
Training, Validation and Test

- Data: labeled instances, e.g. emails marked spam/ham
  - Training set
  - Validation set
  - Test set

- Training
  - **Estimate parameters** on training set
  - **Tune hyperparameters** on validation set
  - **Report results** on test set
  - Anything short of this yields **over-optimistic claims**

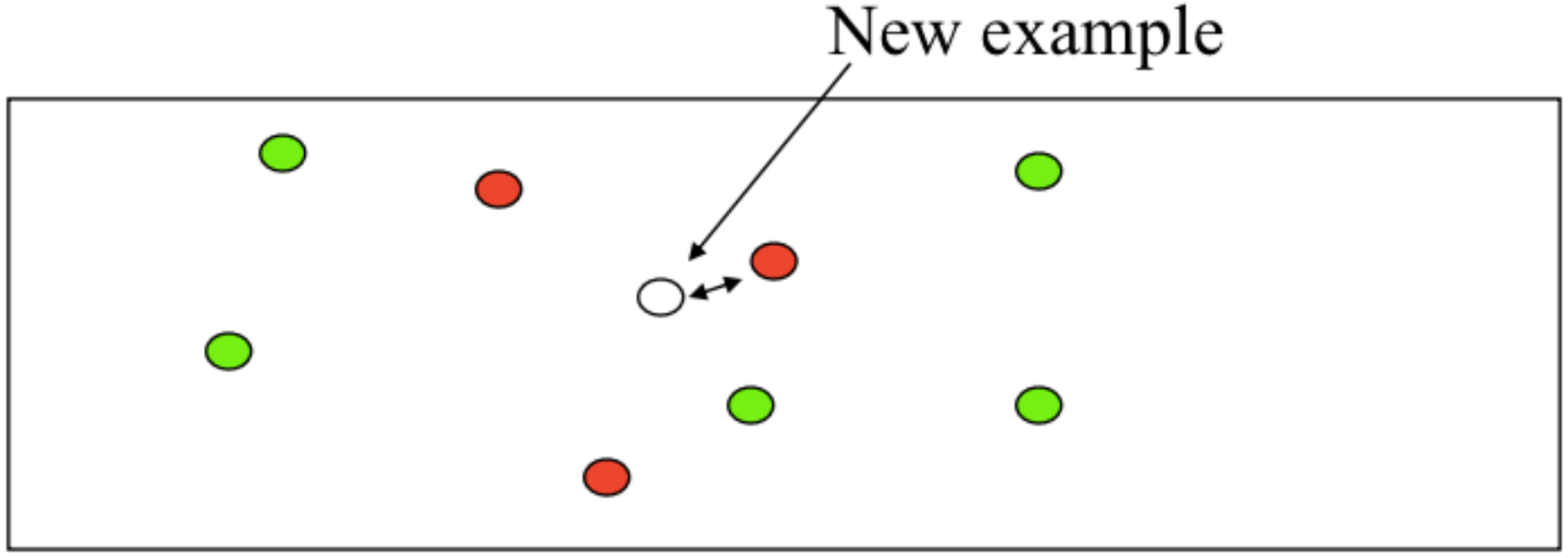
- Evaluation
  - Many different metrics
  - Ideally, the criteria used to train the classifier should be closely related to those used to evaluate the classifier

- Statistical issues
  - Want a classifier which does well on *test* data
  - Overfitting: fitting the training data very closely, but not generalizing well
  - Error bars: want realistic (conservative) estimates of accuracy

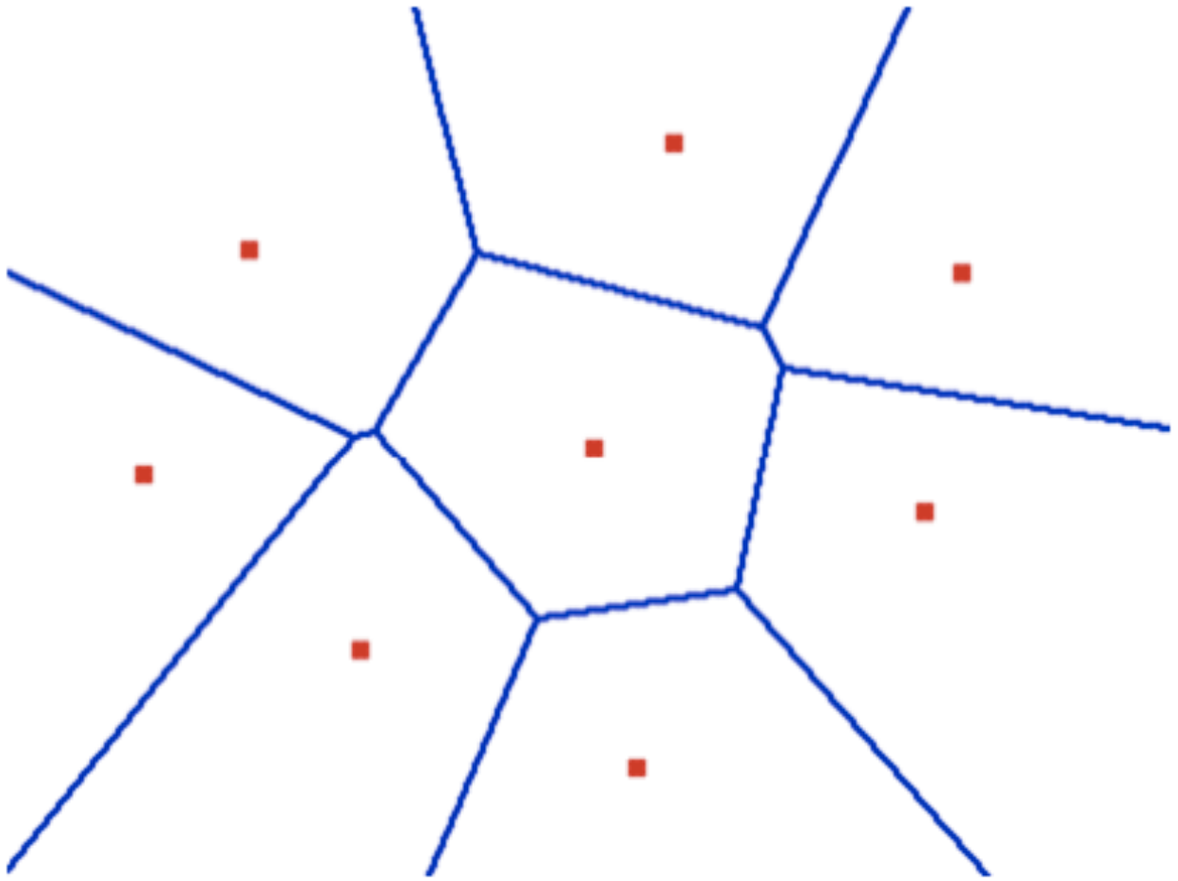


Nearest Neighbour Classifier

- **Remember** all training data
- Decide a **distance** function
- When a test data point comes find the nearest neighbour of the **closest point from training data**
- Assign the **label of nearest neighbour** to the test data

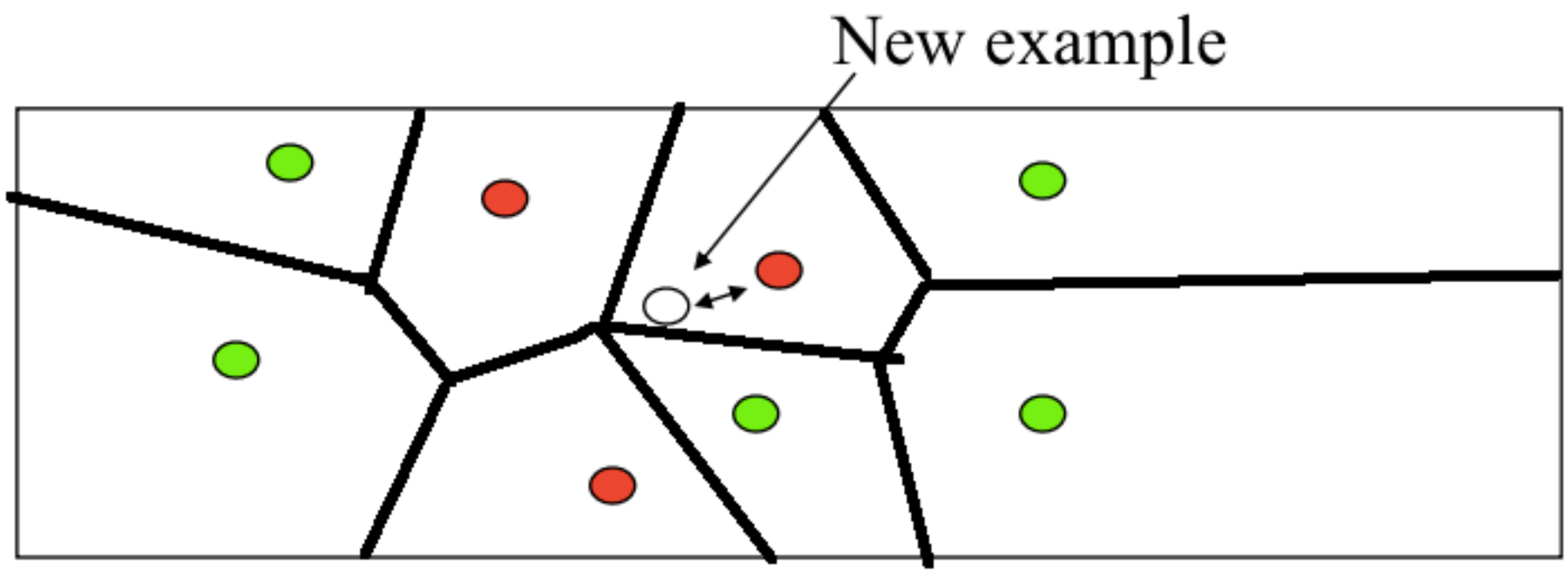


Voronoi Diagram



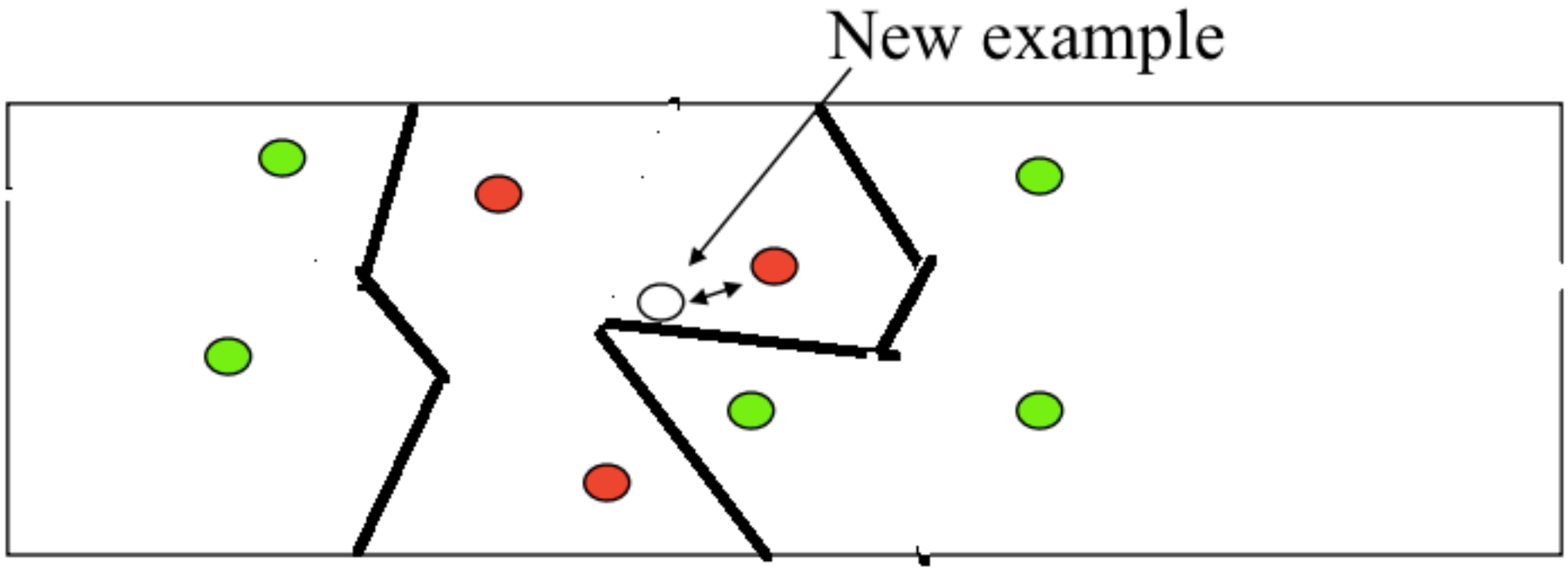
Given a set of points, a Voronoi diagram describes the areas that are nearest to any given point.

Voronoi Diagram

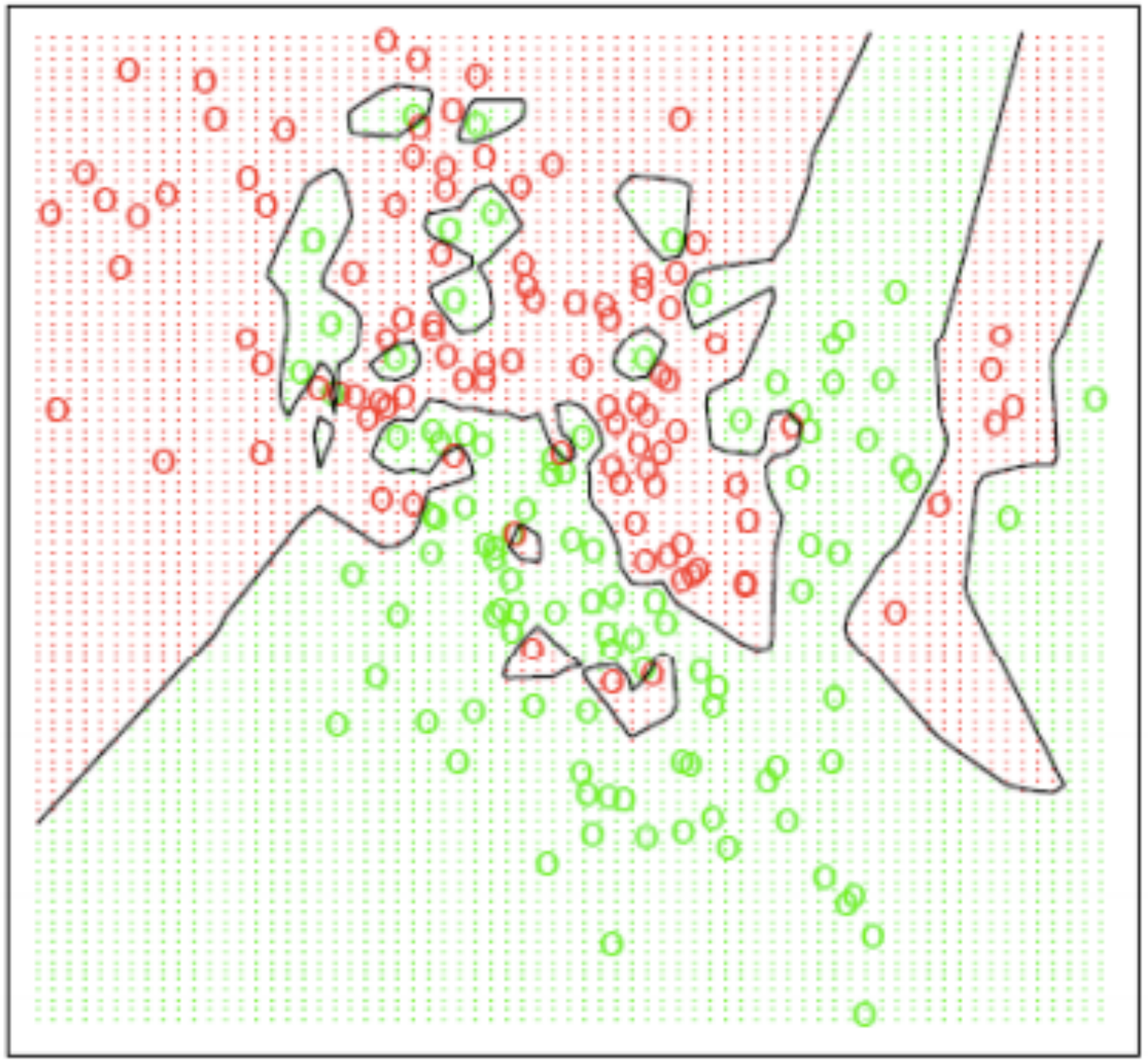


Given a set of points, a Voronoi diagram describes the areas that are nearest to any given point.

Decision Boundary : Voronoi diagram



NN classifier

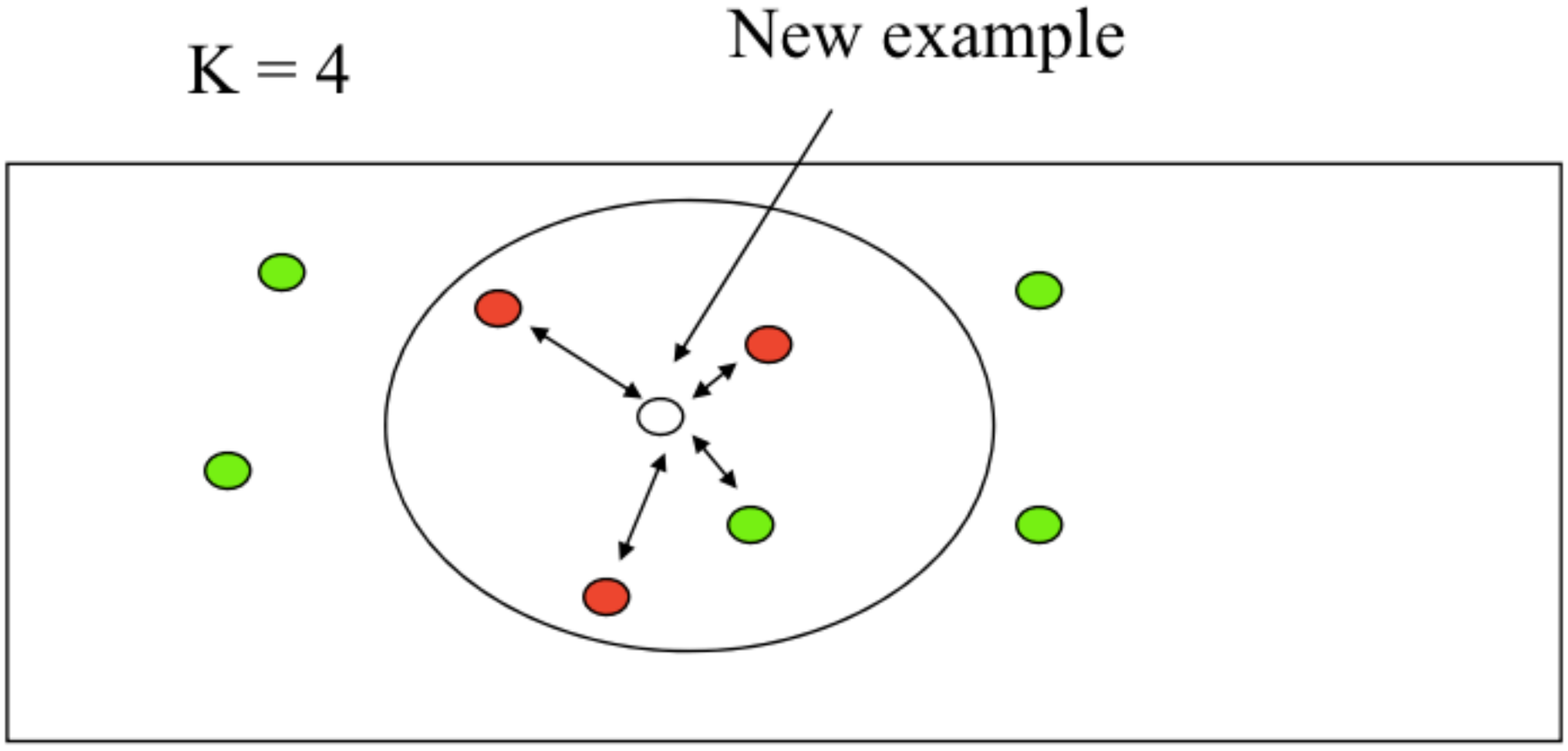


- **Overfitting**

- Solution :
  - Regularisation
  - Smoothing the boundary
- **k neighbours**

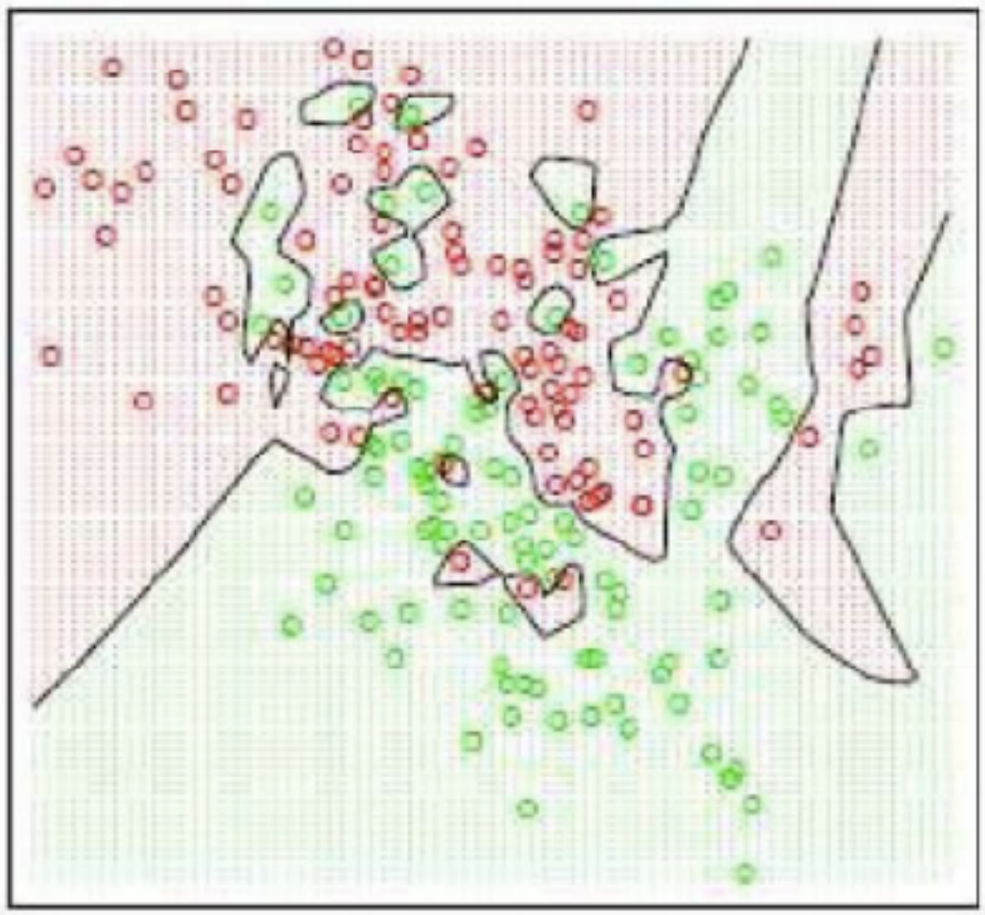
k-NN classifier

- Find the k nearest neighbors
- Have them **vote**
- Has a smoothing effect
- This is especially good when there is noise in the class labels.



Effect of “K”

K=1



K=15

Figures from Hastie, Tibshirani and Friedman (Elements of Statistical Learning)

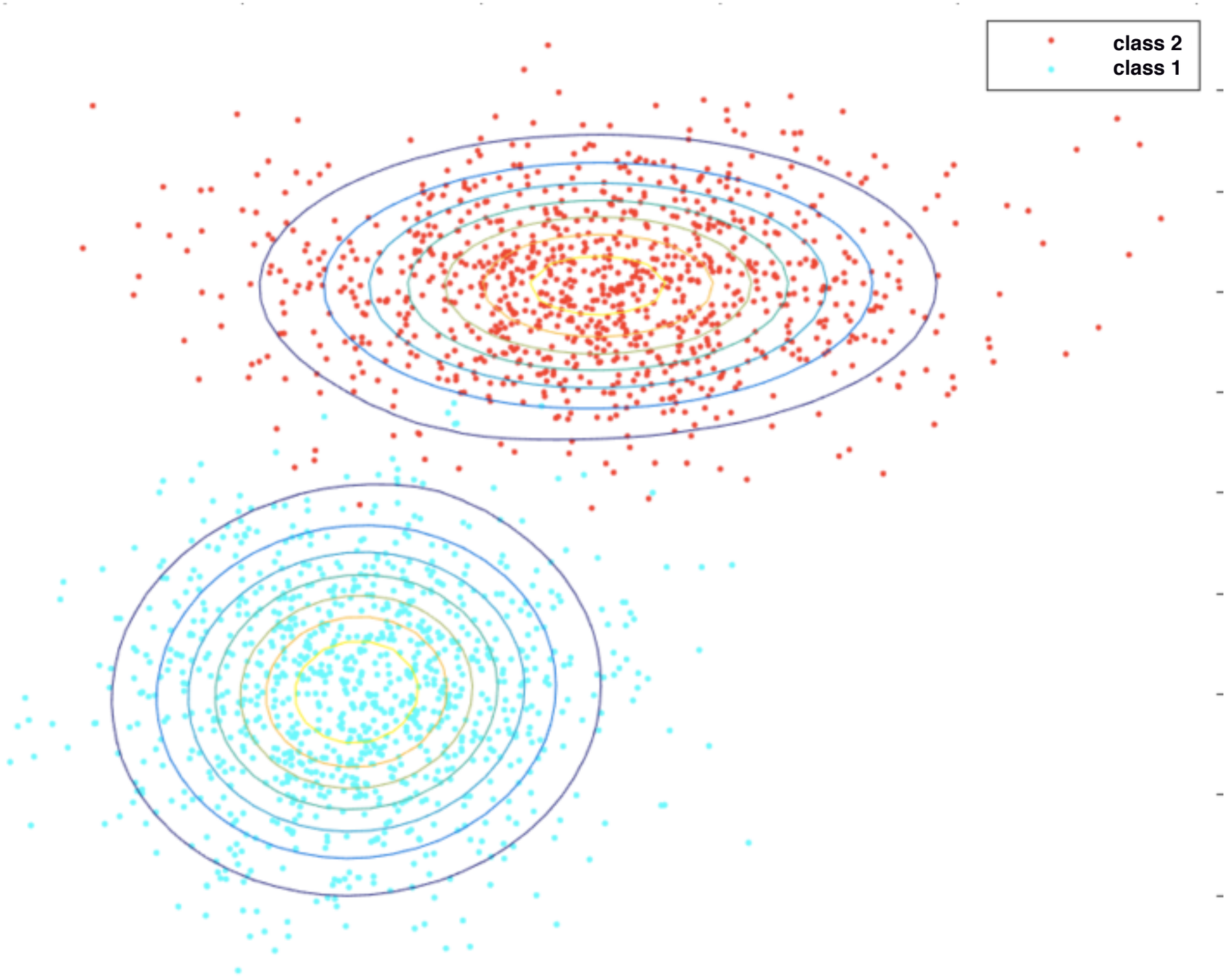
As K increases:

- Classification boundary becomes smoother
- Training error can increase

Choose (learn) K by **cross-validation**

- Split training data into training and validation
- Hold out validation data and measure error on this

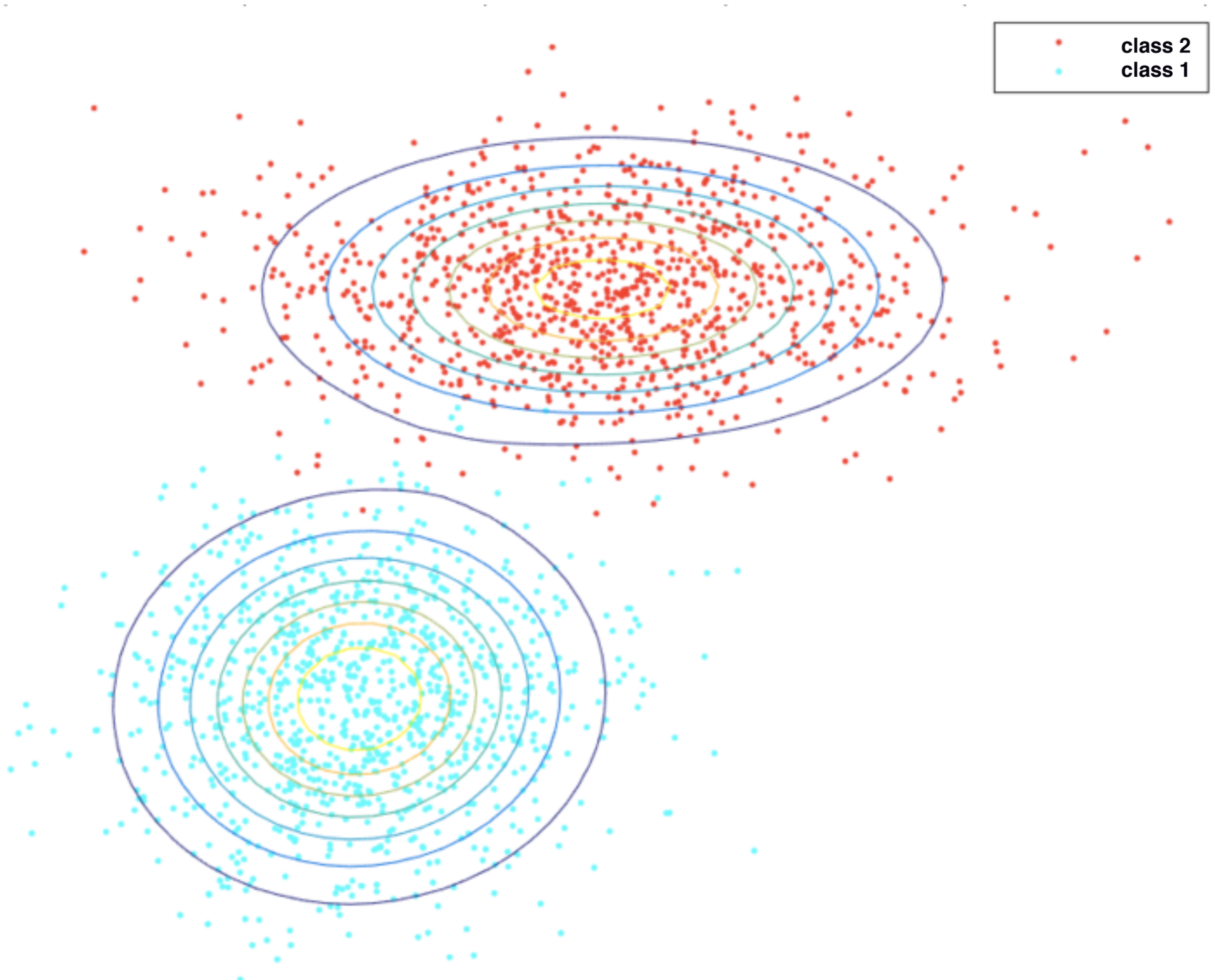
Probabilistic Decision Boundary



Probabilistic Decision Boundary

prior probability
$P(C_1)$ and $P(C_2)$

$P(C_1)$= fraction points having cyan colour

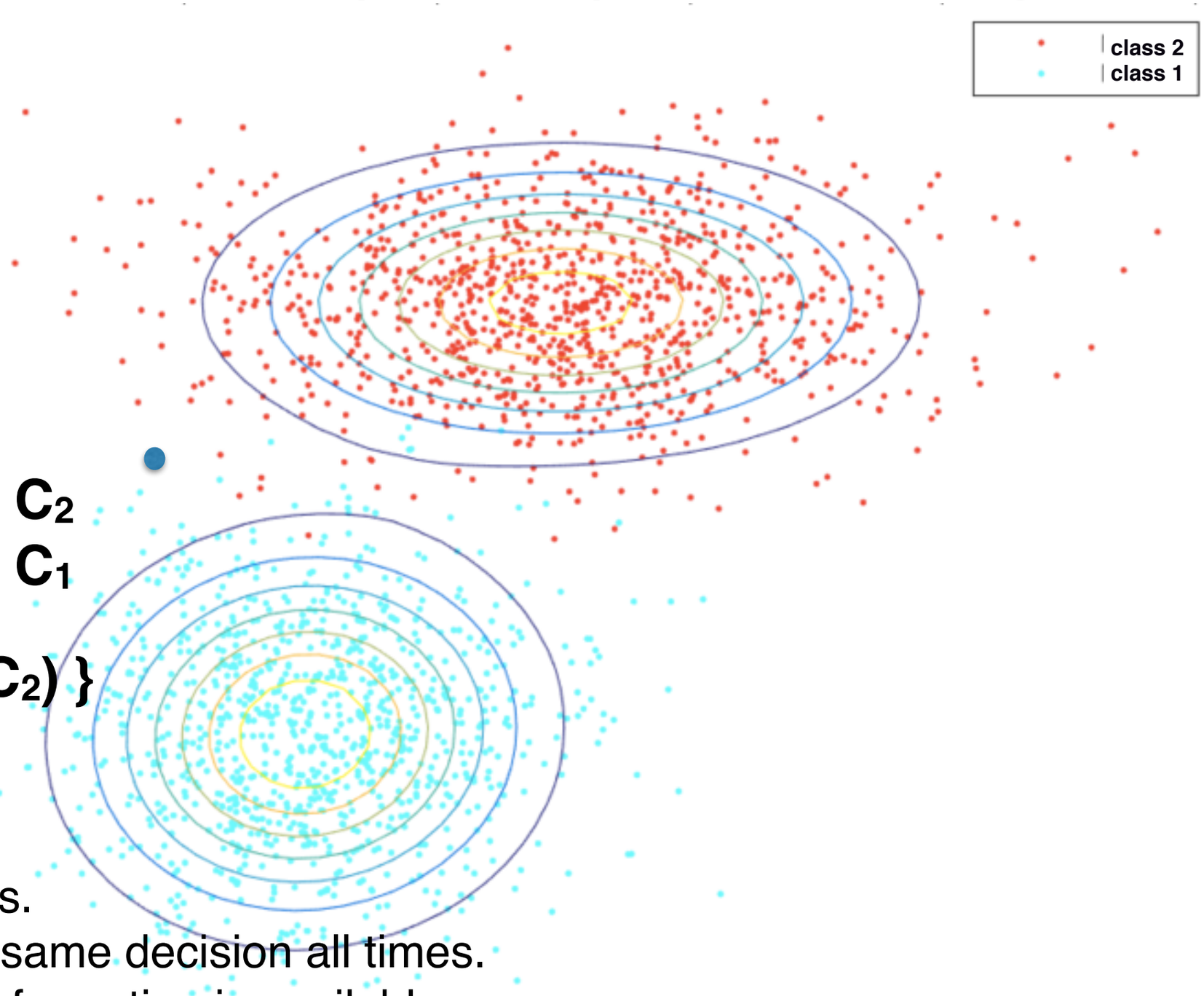


Decision Rule with prior probability

Decide:

Label =1 [class C_1]
if $P(C_1) > P(C_2)$

Label =-1 [class C_2]
if $P(C_2) > P(C_1)$



P(error)= $P(C_1)$ if decide C_2
$P(C_2)$ if decide C_1

P(error)= min { $P(C_1)$, $P(C_2)$ }

- Favours the most likely class.
- This rule will be making the same decision all times.
- – i.e., optimum if no other information is available

Classification error for random classifier

Decide:
Label =1 [class C_1]
if $P(C_1) > P(C_2)$
Label =-1 [class C_2]
if $P(C_2) > P(C_1)$

P(error)= $P(C_1)$ if decide C_2
$P(C_2)$ if decide C_1

P(error)= min { $P(C_1)$, $P(C_2)$ }

Binary classification

Random classifiers

Decide:
Label =1 and Label =-1
randomly with probability 0.5

Assumes $P(C_1) = P(C_2)=0.5$

p(error)= min { $P(C_1)$, $P(C_2)$ } = 0.5

Next Class

- 17/8
  - Bayes Classifier, Naive Bayes Classifier
  - Logistic Regression

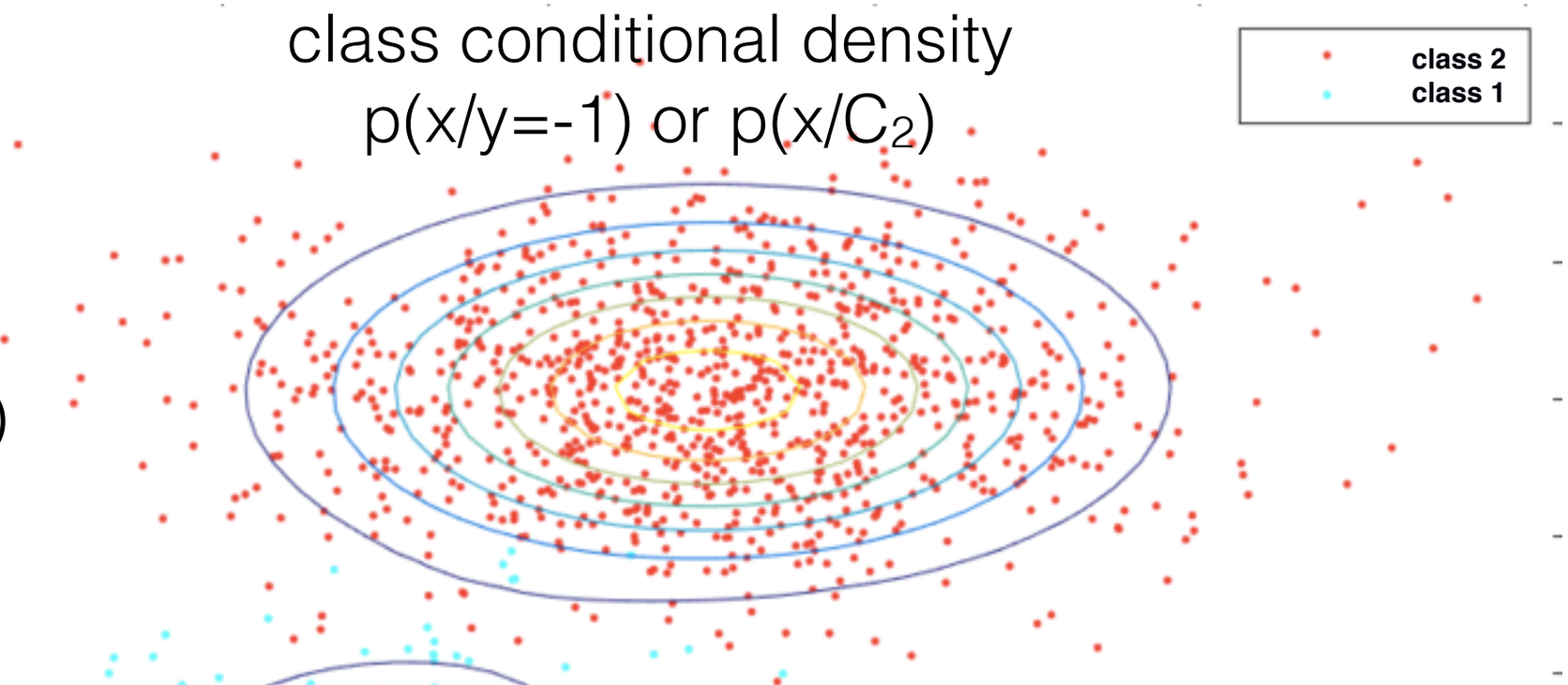
CS4801 : Bayesian Decision Theory

Sahely Bhadra
18/09/2018

2.Bayes decision rule
  1. Classification error
  2. Minimum error rate classification
    1. Two category
    2. Multi category
3.Logistic Regression
    1. Linear separator
    2. loss function of logistic regression

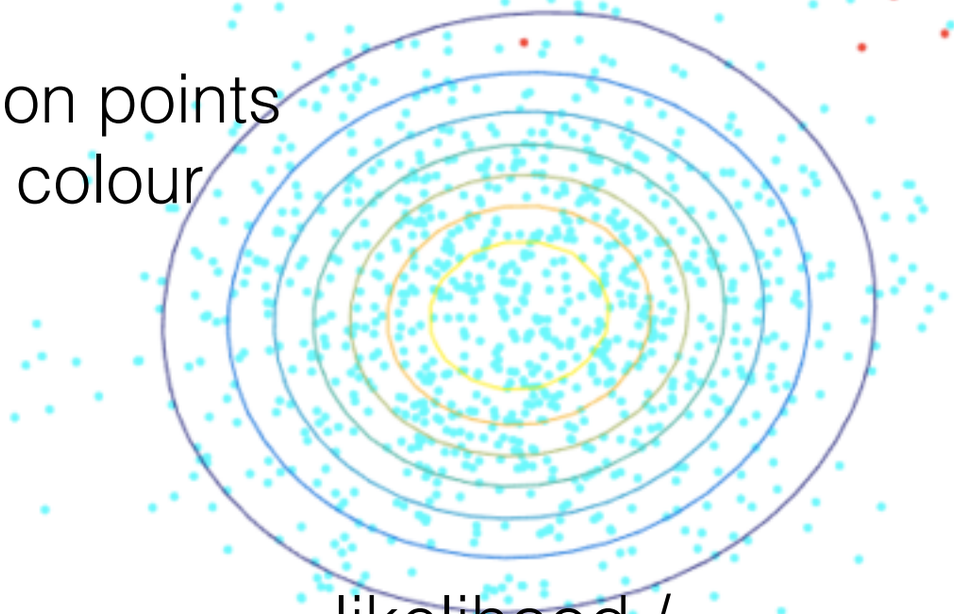
Probabilistic Decision Boundary

likelihood /
class conditional density
p(x/y=-1) or p(x/C_2)

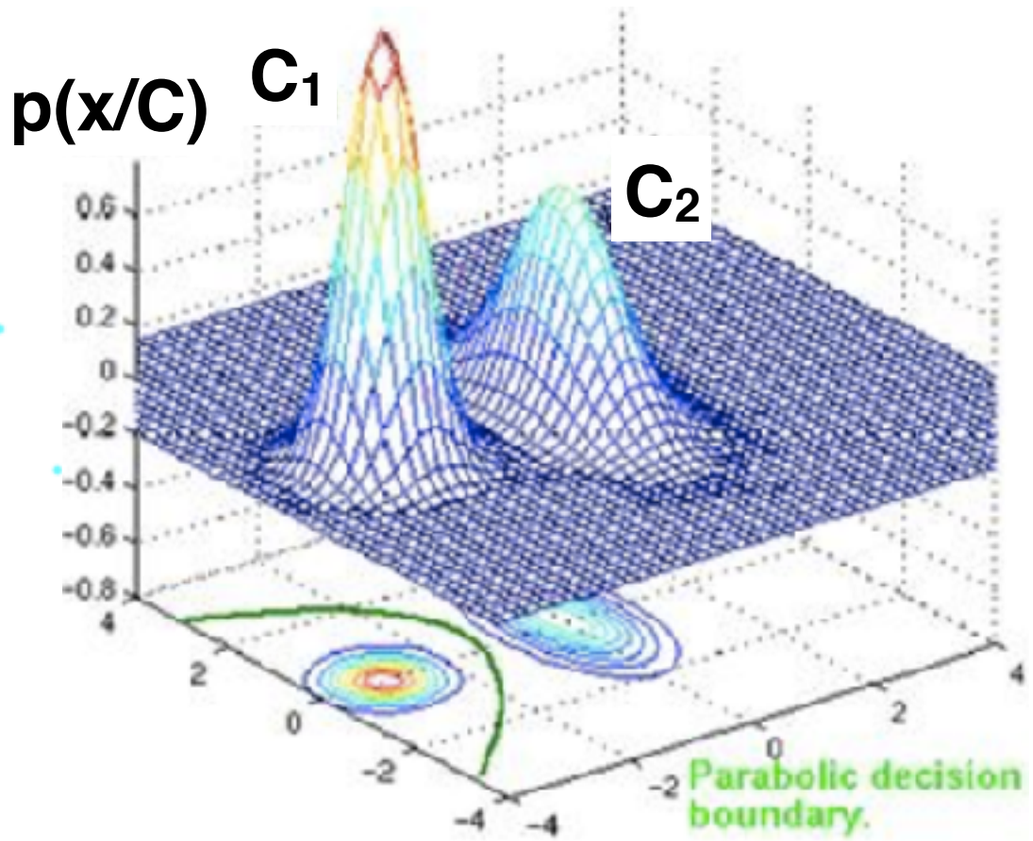


prior probability
P(C_1) and P(C_2)

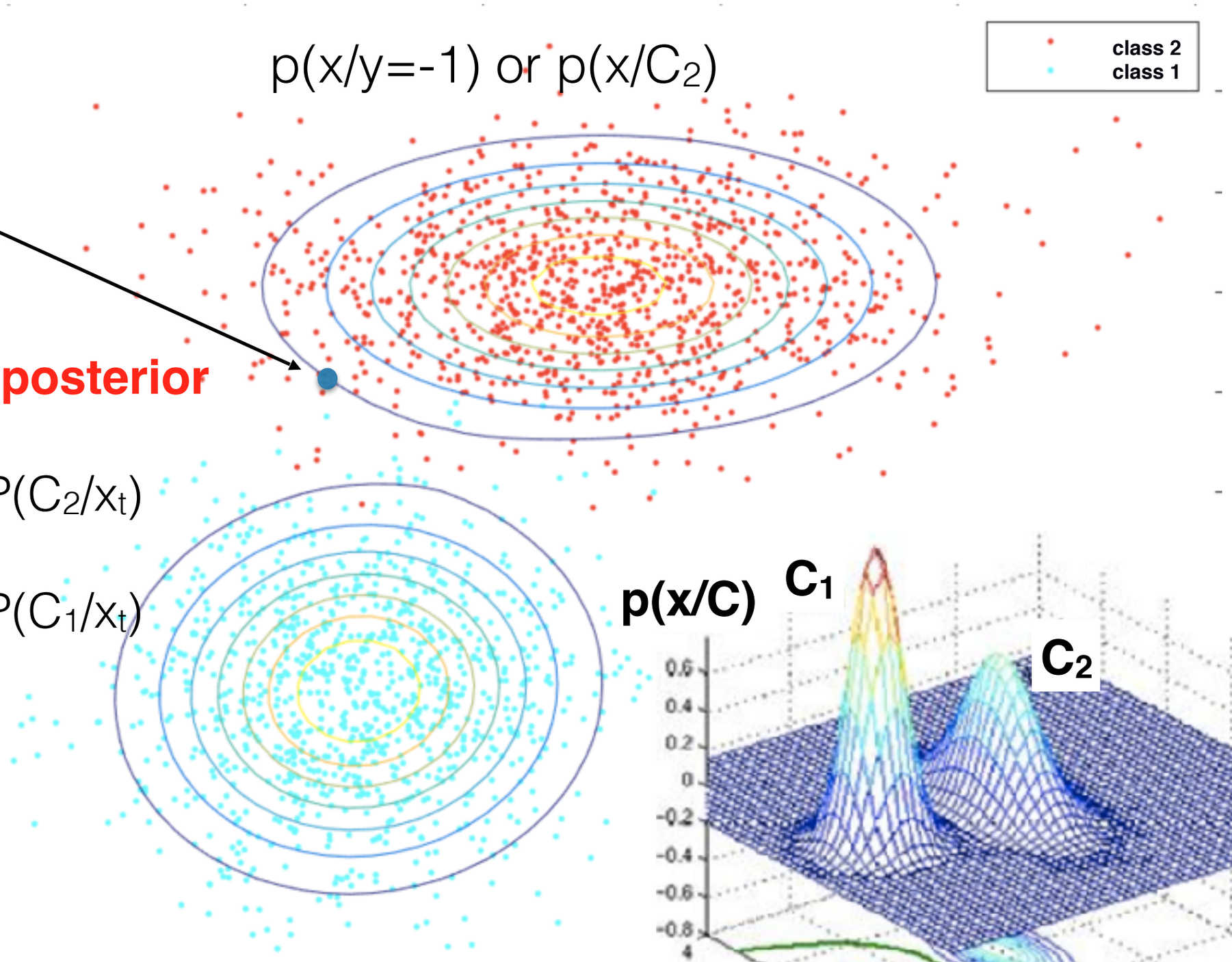
P(C_1)= fraction points
having cyan colour



likelihood /
class conditional density
p(x/y=1) or p(x/C_1)



Bayes Classifier



New point (x_t)

Classification on posterior

Label =1

if $P(C_1/x_t) > P(C_2/x_t)$

Label =-1

if $P(C_2/x_t) > P(C_1/x_t)$

Bayes Classifier

Classification on posterior

Label =1

if $P(C_1/x_t) > P(C_2/x_t)$

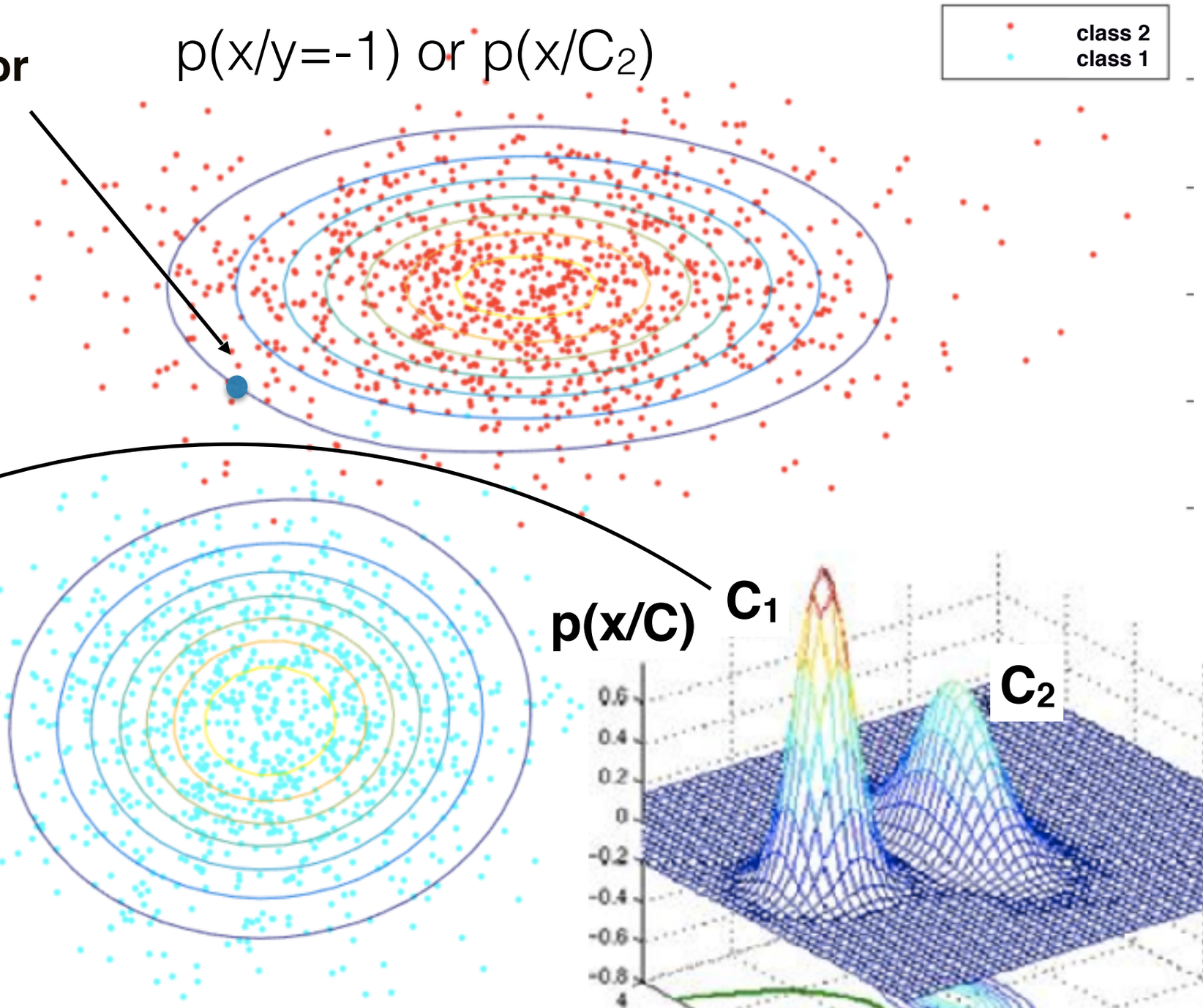
Label =-1

if $P(C_2/x_t) > P(C_1/x_t)$

Decision Function $G(x_t)$

$sign[P(C_1/x_t) - P(C_2/x_t)]$

p(x/y=-1) or $p(x/C_2)$



p(x/y=1) or $p(x/C_1)$

Probabilistic Decision Boundary

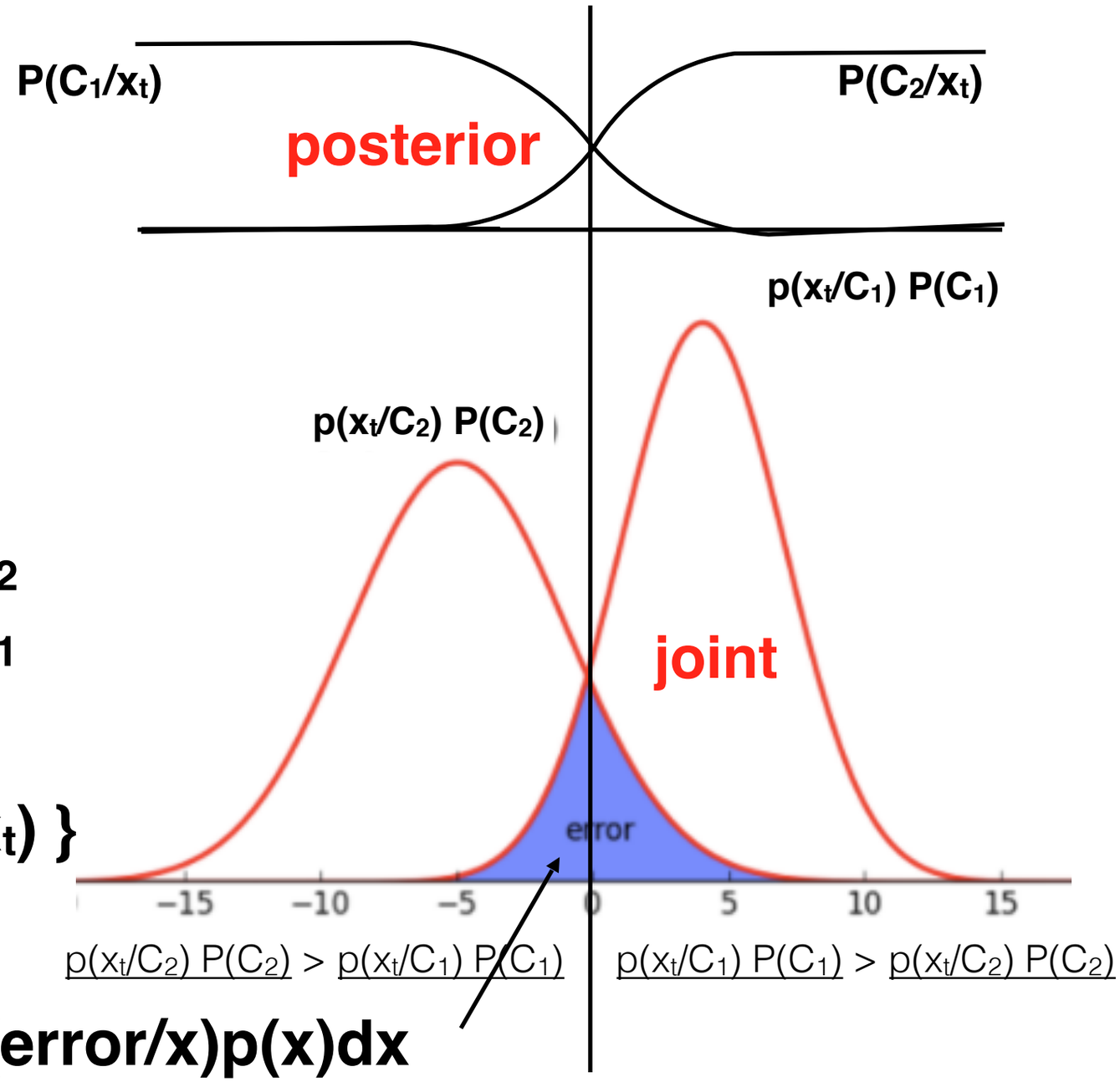
Classification on posterior

Label =1

if $P(C_1/x_t) > P(C_2/x_t)$

Label =-1

if $P(C_2/x_t) > P(C_1/x_t)$



$P(error/x) = P(C_1/x_t)$ if decide C_2

$P(C_2/x_t)$ if decide C_1

$P(error/x) = \min \{ P(C_1/x_t), P(C_2/x_t) \}$

$$P(error) = \int_{-\infty}^{\infty} P(error, x)dx = \int_{-\infty}^{\infty} P(error/x)p(x)dx$$

The Bayes rule is optimum, that is, it minimises the average probability error!

Recap on classifier

A. No information
Random classifier
For k class classification problem
assign class label k to a text point with probability 1/k

B. Prior or class probability is known

P(error)= $P(C_1)$ if decide C_2
$P(C_2)$ if decide C_1

P(error)= min { $P(C_1)$, $P(C_2)$ }

Recap on classifier

A. No information
Random classifier
For k class classification problem assign class label k to a text point with probability 1/k

B. Prior or class probability is known

$$P(error) = \begin{cases} P(C_1) & \text{if decide } C_2 \\ P(C_2) & \text{if decide } C_1 \end{cases}$$

$$P(error) = \min \{ P(C_1), P(C_2) \}$$

C. Posterior is known
Bayes classifier
The Bayes rule is optimum, that is, it minimises the average probability error!

$$P(error/x) = \begin{cases} P(C_1/x_t) & \text{if decide } C_2 \\ P(C_2/x_t) & \text{if decide } C_1 \end{cases}$$

$$P(error/x) = \min \{ P(C_1/x_t), P(C_2/x_t) \}$$

$$P(error) = \int_{-\infty}^{\infty} P(error, x)dx = \int_{-\infty}^{\infty} P(error/x)p(x)dx$$

Expected Loss

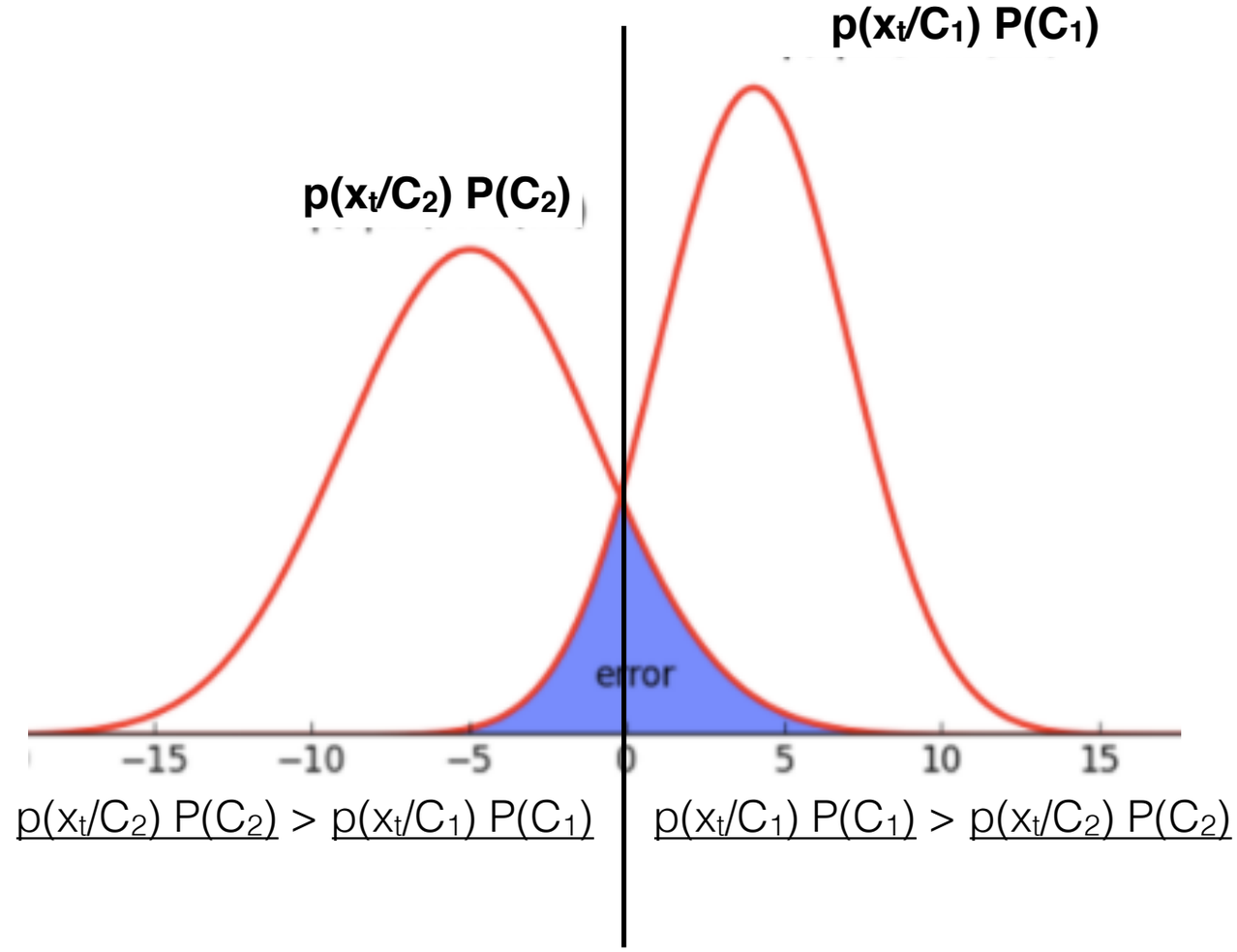
- Classification = g(x) : x -> {C_1,C_2}
- cost(g(x) , y)= **L(g(x) , y)** =cost of assigning class label ‘f(x)’ when the correct label is ‘y’.

Expected loss

$$\text{Risk}(g) = E[R(x)] = \int_x R(x)\, p(x)\, dx$$

$$= \int_{g(x)=C_1} L(g(x),C_2)p(x/C_2)\, P(C_2)\, dx$$

$$+ \int_{g(x)=C_2} L(g(x),C_1)p(x/C_1)\, P(C_1)\, dx$$



$$\text{Risk}(g) = E[L(f(x),y)] = \int_x \int_y L(g(x),y)p(x,y)\, dxdy$$

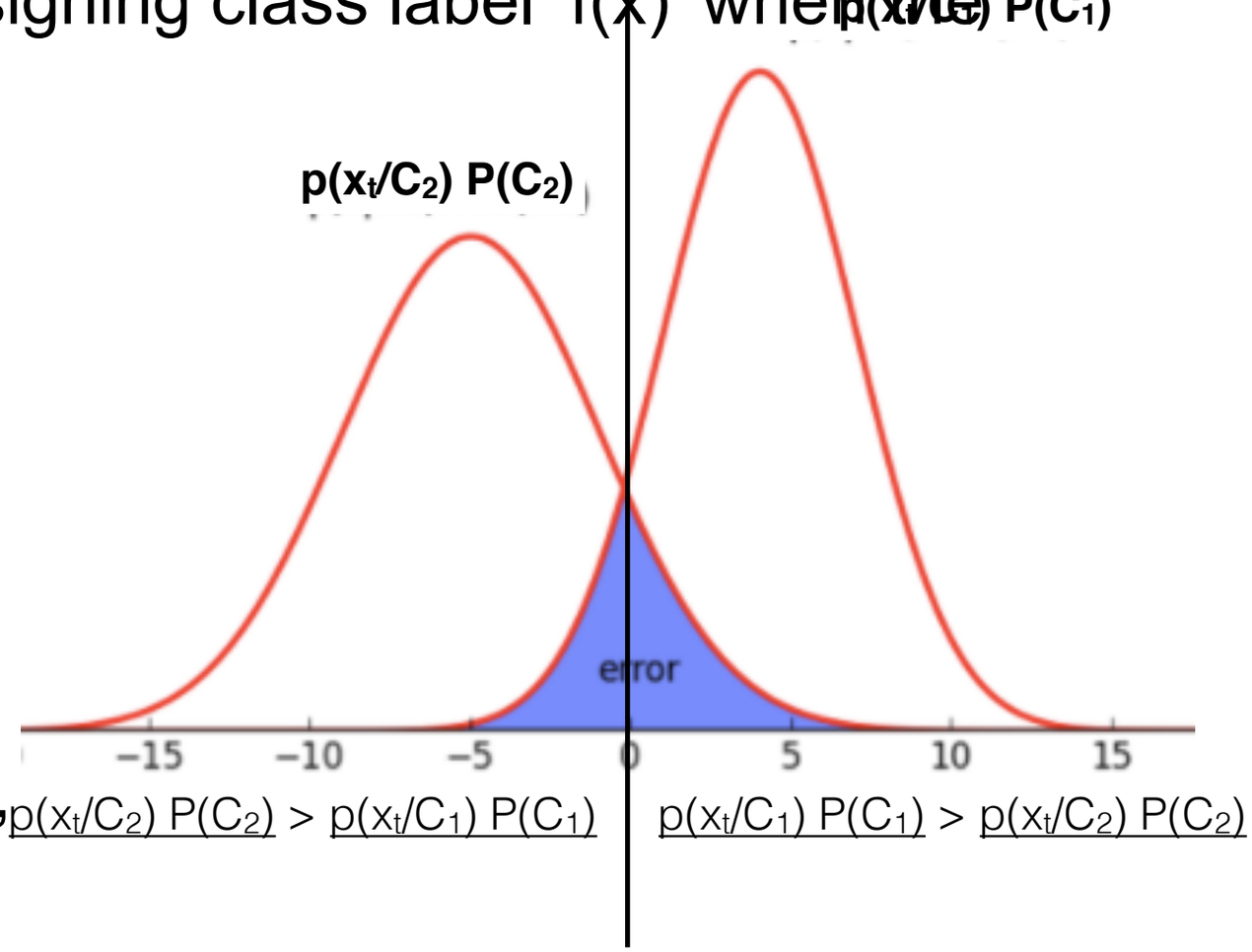
Bayes Error

- Classification = g(x) : x -> {C_1,C_2}
- cost(g(x) , y)= **L(g(x) , y)** =cost of assigning class label ‘f(x)’ when the correct label is ‘y’.

Expected loss or risk of classifier ‘f’

Risk(g) = E[L(g(x),y)]

$$= \int_{xy} L(g(x),y)p(x,y)\, d(x,y)$$



A classifier f^* is called **Bayes optimal**, or **Bayes classifier**, if it minimises Risk(g).

$p(x_t/C_2)\,P(C_2) > p(x_t/C_1)\,P(C_1)$ | $p(x_t/C_1)\,P(C_1) > p(x_t/C_2)\,P(C_2)$

Decide The minimum expected loss $Risk(g^*)$ is called the **Bayes error**.

Bayes classifier

Classification = g(x) : x -> {C_1,C_2}
cost(g(x) , y)= **L(g(x) , y)** =cost of assigning class label ‘f(x)’ when the correct label is ‘y’.

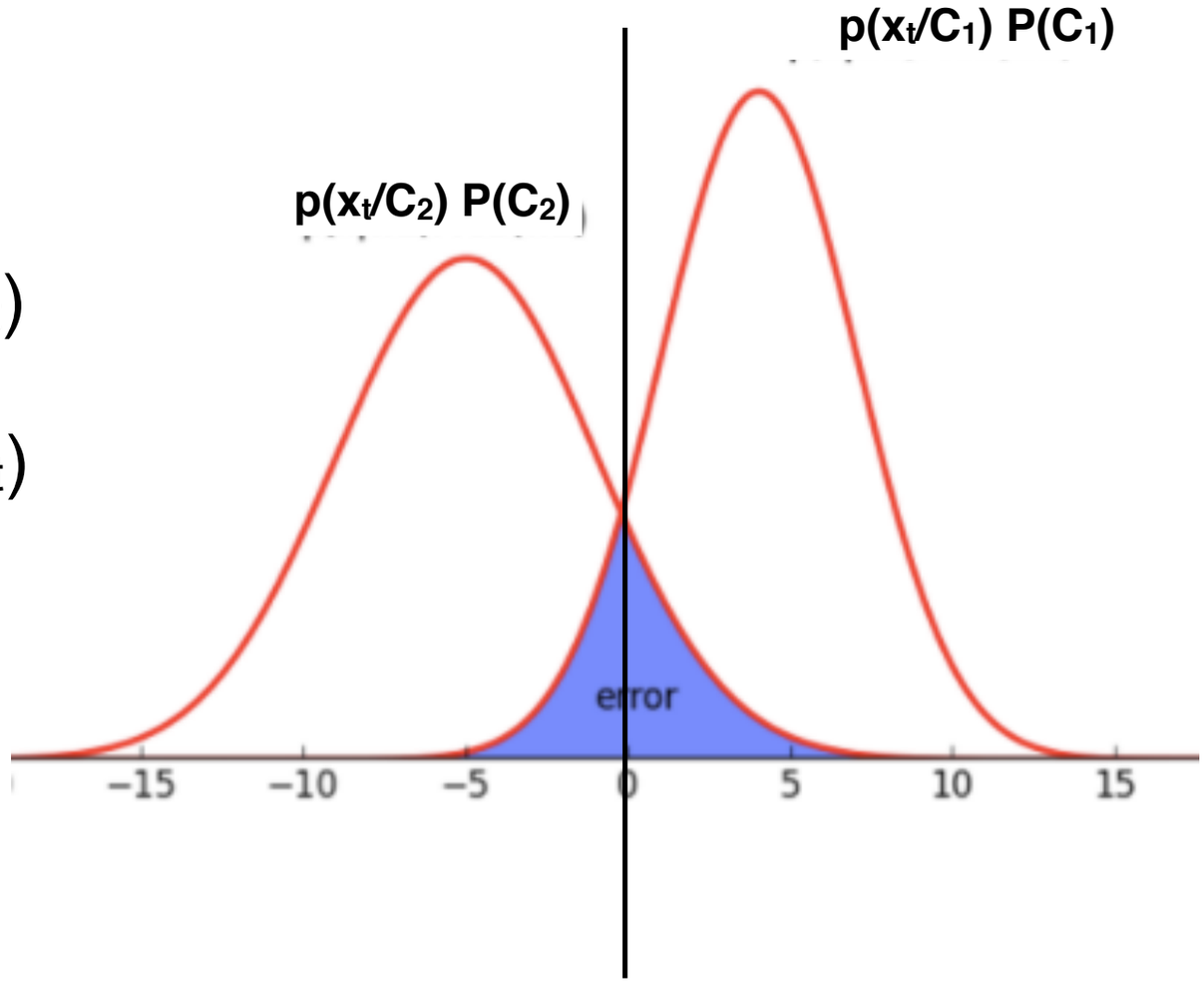
Risk(g) = E[L(g(x),y)]

Risk(g=C_1/x) =L_{11} P(C_1/x_t) +L_{12} P(C_2/x_t)

Risk(g=C_2/x) =L_{21} P(C_1/x_t) +L_{22} P(C_2/x_t)

Bayes classifier : minimises Risk(g)

Decide g=C_1
if
Risk(g=C_1/x) < Risk(g=C_2/x)
(L_{11} - L_{21}) P(C_1/x_t) < (L_{22} - L_{12}) P(C_2/x_t)



Decide The minimum expected loss Risk(g^*) is called the **Bayes error**.

Bayes classifier : Zero One Loss

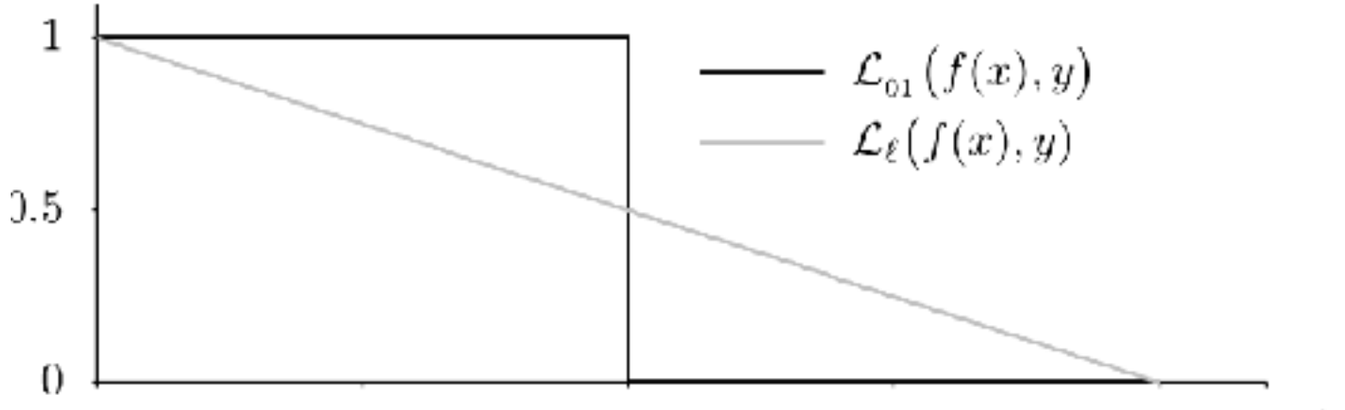
Bayes classifier : minimises Risk(f)

Decide $f=C_1$ if $(L_{11} - L_{21})\, P(C_1/x_t) < (L_{22} - L_{12})\, P(C_2/x_t)$ otherwise $f=C_2$

For Zero One Loss

$L_{11} = L_{22} = 0$

$L_{12} = L_{21} = 1$

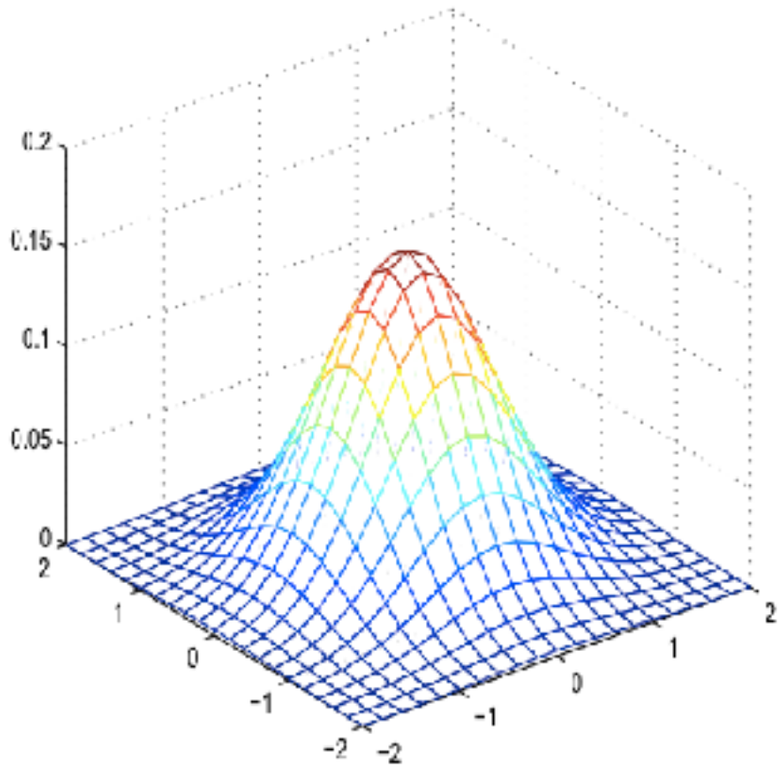


Decide $f=C_1$

if $(0 - 1)\, P(C_1/x_t) < (0 - 1)\, P(C_2/x_t)$ or $P(C_2/x_t) < P(C_1/x_t)$

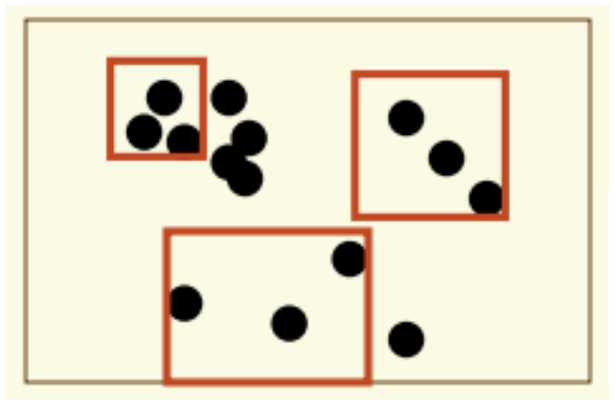
otherwise $f=C_2$

Non parametric density estimation



$$P = \int_{\mathcal{R}} p(\mathbf{x}) d\mathbf{x} \approx p(\mathbf{x}) \int_{\mathcal{R}} d\mathbf{x} = p(\mathbf{x}) V$$

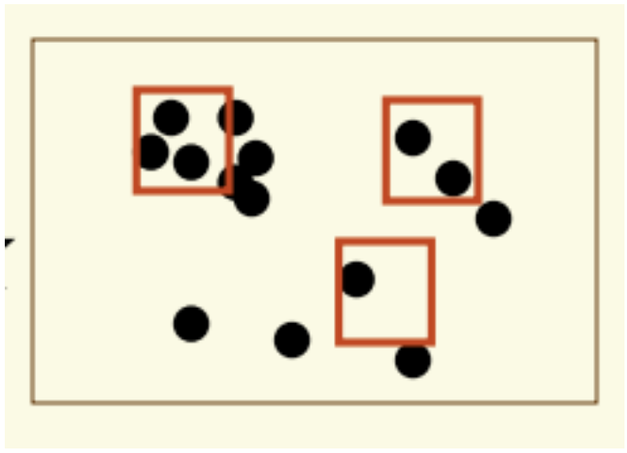
Nearest Neighbour (fixed k)



$$P = k/n$$

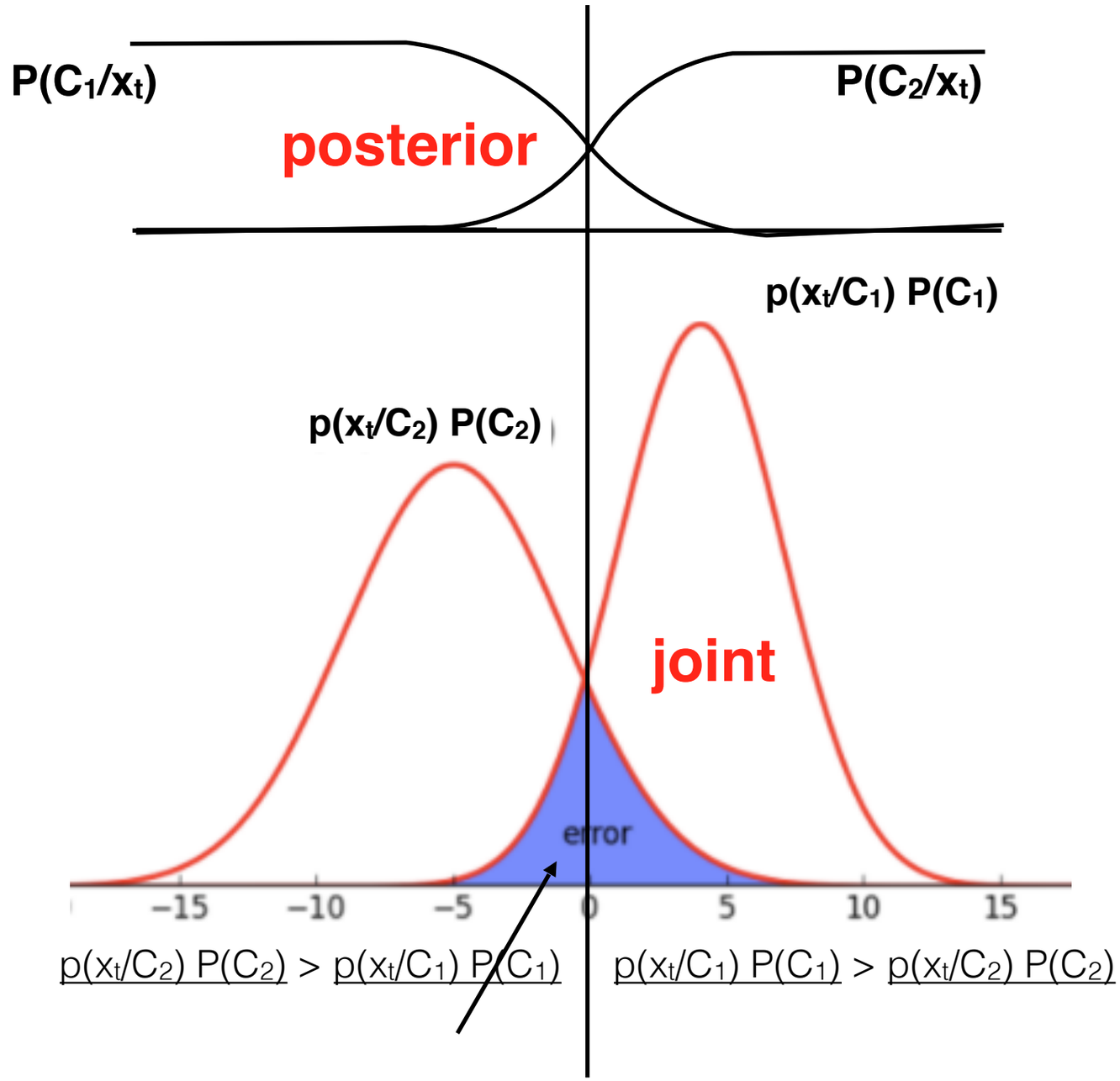
$$p(\mathbf{x}) = \frac{k/n}{V}$$

Parzen window (Fixed V)



$$P = k/n$$

$$p(\mathbf{x}) = \frac{k/n}{V}$$


P(C1/xt)
P(C2/xt)
posterior
p(xt/C1) P(C1)
p(xt/C2) P(C2)
joint
error
−15
−10
−5
0
5
10
15
p(xt/C2) P(C2) > p(xt/C1) P(C1)
p(xt/C1) P(C1) > p(xt/C2) P(C2)

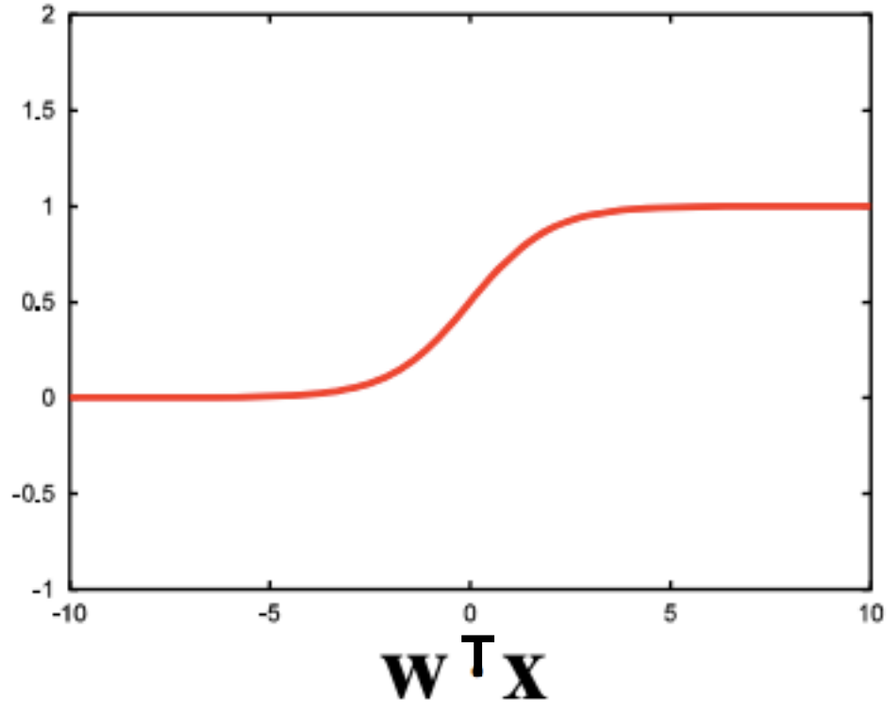
Logistic Regression

- Posterior is defined as

$$p(y=1 \mid \mathbf{x};\mathbf{w}) = g(\mathbf{x},\mathbf{w}) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}}$$

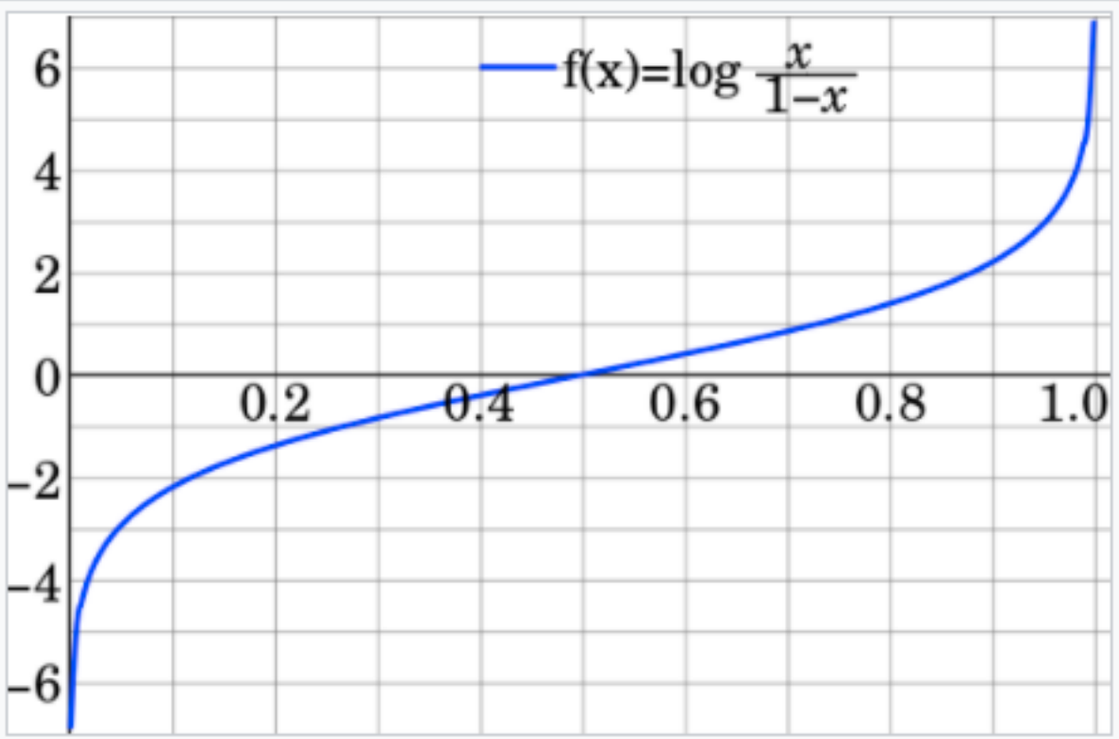
$$p(y=0 \mid \mathbf{x};\mathbf{w}) = 1 - g(\mathbf{x},\mathbf{w})$$

$$g(\mathbf{x},\mathbf{w}) = \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})}$$



Logistic Regression

$$\operatorname{logit}(p) = \log\left(\frac{p}{1-p}\right) = \log(p) - \log(1-p) = -\log\left(\frac{1}{p} - 1\right).$$



Plot of logit(*p*) in the domain of 0 to 1, where the base of logarithm is *e*

$$\operatorname{logit}^{-1}(\alpha) = \operatorname{logistic}(\alpha) = \frac{1}{1+\exp(-\alpha)} = \frac{\exp(\alpha)}{\exp(\alpha)+1}$$

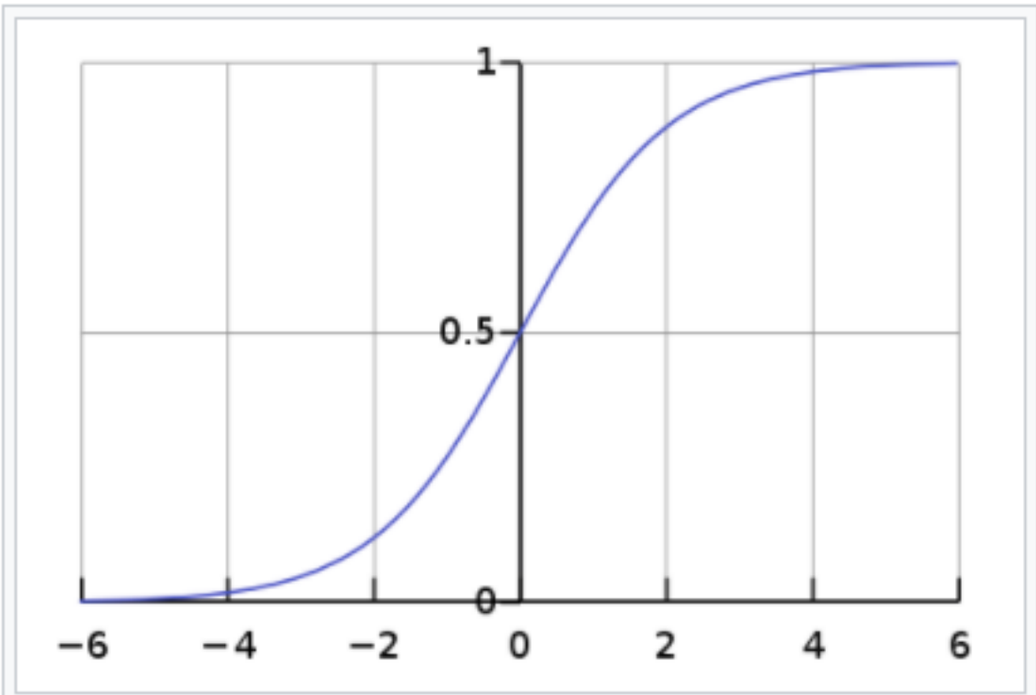


Figure 1. The standard logistic function $\sigma(t)$; note that $\sigma(t) \in (0,1)$ for all t.

Logistic Regression

Logistic Regression : parametric assumption for posterior distribution

$$P(y=C_1/x) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}}$$

$$P(y=C_2/x) = 1- P(y=C_1/x)$$

Hence assuming $y=C_1 \Rightarrow y=+1$ and $y=C_2 \Rightarrow y=-1$

$$P(y/x) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}.y}}$$

Logistic Regression : Loss function

Logistic Regression : parametric assumption for posterior distribution

Hence assuming $y=C_1 \Rightarrow y=+1$ and $y=C_2 \Rightarrow y=-1$

$$P(y/x) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}.y}}$$

By maximising log P(y/x):

$$\min_{\mathbf{w}\in\mathbb{R}^d} \sum_i^N \underbrace{\log\left(1+e^{-y_i f(\mathbf{x}_i)}\right)}_{\text{loss function}} + \lambda \underbrace{||\mathbf{w}||^2}_{\text{regularization}}$$

Can be solved using Gradient Descent

Logistic Regression : Linear Classifier

P(y=1/x) = $\frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}}$

Hence we predict y=1 if $\frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}}$ >= 0.5

or $1 \geq e^{-\mathbf{w}^T\mathbf{x}}$

or $w^Tx>=0$

Hence Logistic regression is a Linear Function

Multi classes case

Choose class K to be the “reference class” and represent each of the other classes as a logistic function of the odds of class *k* versus class K:

$$\log \frac{P(y=1|\mathbf{x})}{P(y=K|\mathbf{x})} = \mathbf{w}_1{}^{\mathsf{T}}\mathbf{x}$$

$$\log \frac{P(y=2|\mathbf{x})}{P(y=K|\mathbf{x})} = \mathbf{w}_2{}^{\mathsf{T}}\mathbf{x}$$

$$\vdots$$

$$\log \frac{P(y=K-1|\mathbf{x})}{P(y=K|\mathbf{x})} = \mathbf{w}_{K-1}{}^{\mathsf{T}}\mathbf{x}$$

$$P(y=k|\mathbf{x}) = \frac{\exp(\mathbf{w}_k{}^{\mathsf{T}}\mathbf{x})}{1+\sum_{l=1}^{K-1}\exp(\mathbf{w}_l{}^{\mathsf{T}}\mathbf{x})} \qquad P(y=K|\mathbf{x}) = \frac{1}{1+\sum_{l=1}^{K-1}\exp(\mathbf{w}_l{}^{\mathsf{T}}\mathbf{x})}$$

Next Class

- Perceptron

CS4801 : Perceptron and SVM

Sahely Bhadra
24/8/2017

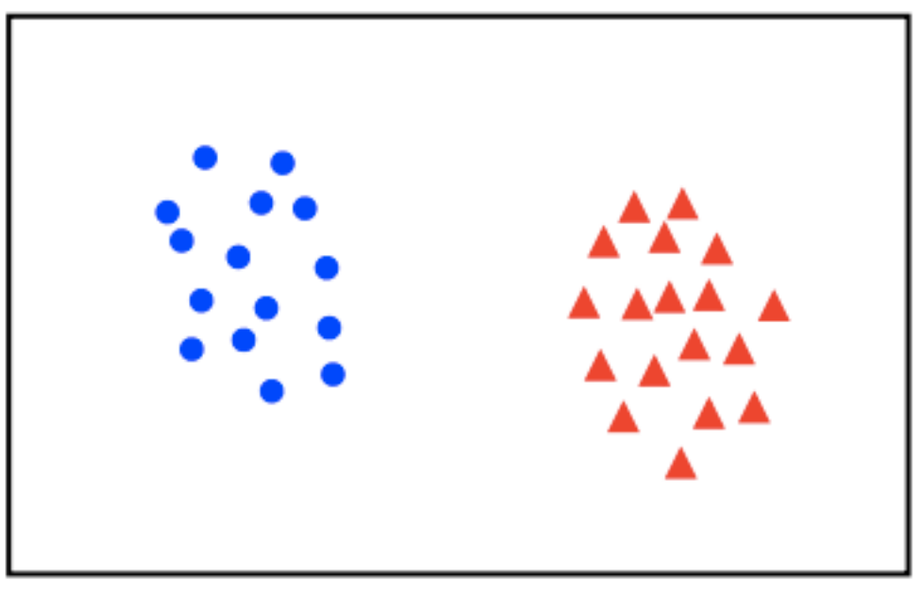
1. Linear separability
2. Perceptron loss
3. Perceptron algorithm

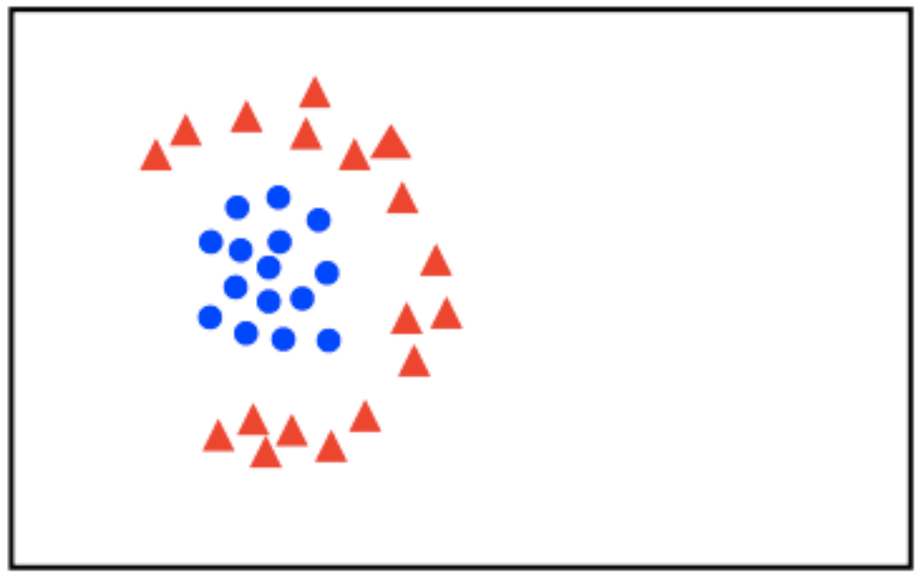
Classification

Given training data $(\mathbf{x}_i, y_i)$ for $i = 1 \ldots N$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$, learn a classifier $f(\mathbf{x})$ such that

$$f(\mathbf{x}_i) \begin{cases} \geq 0 & y_i = +1 \\ < 0 & y_i = -1 \end{cases}$$

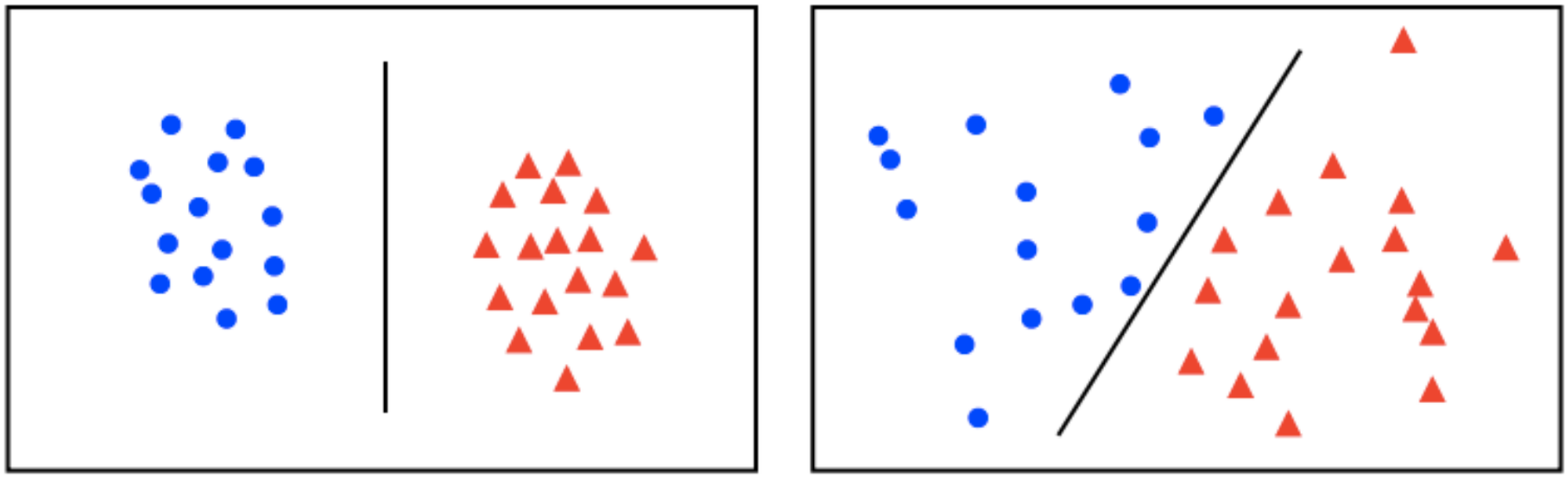
i.e. $y_i f(\mathbf{x}_i) > 0$ for a correct classification.



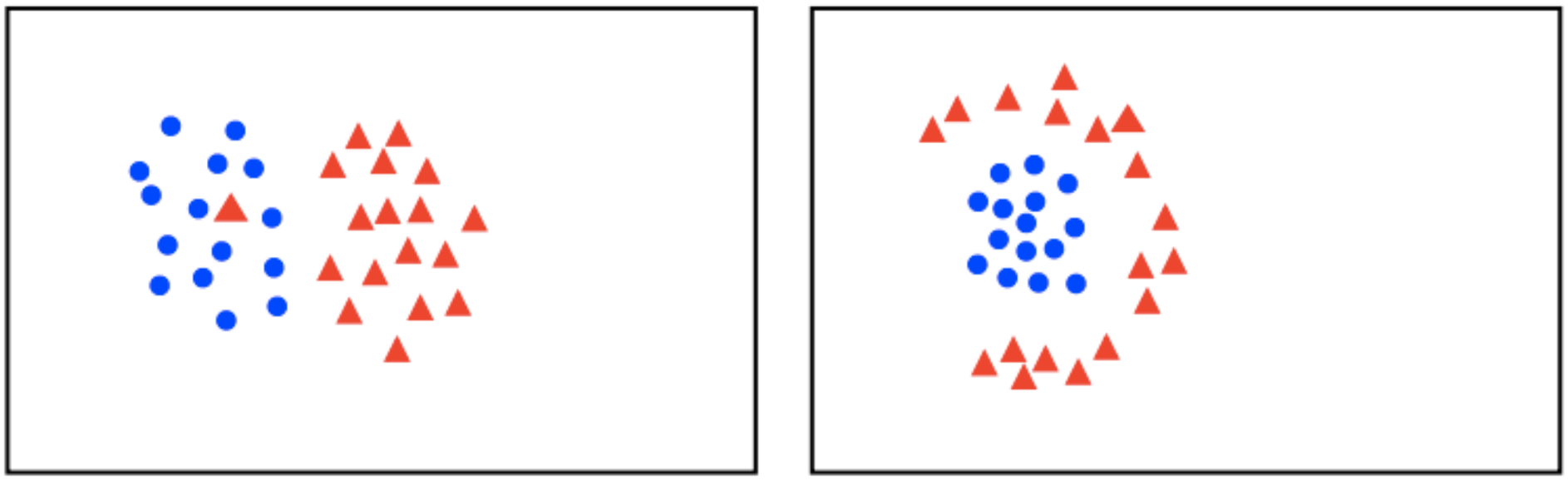


Linear Separability

linearly separable



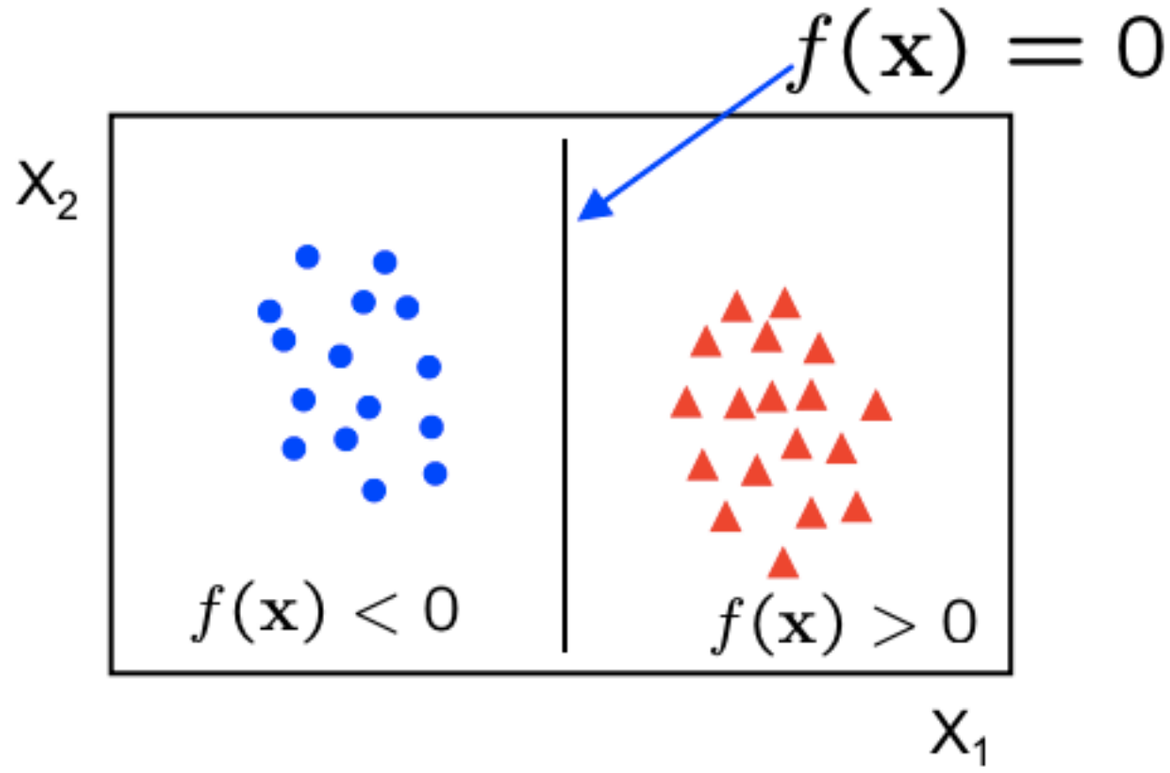
not linearly separable



Linear Classifier

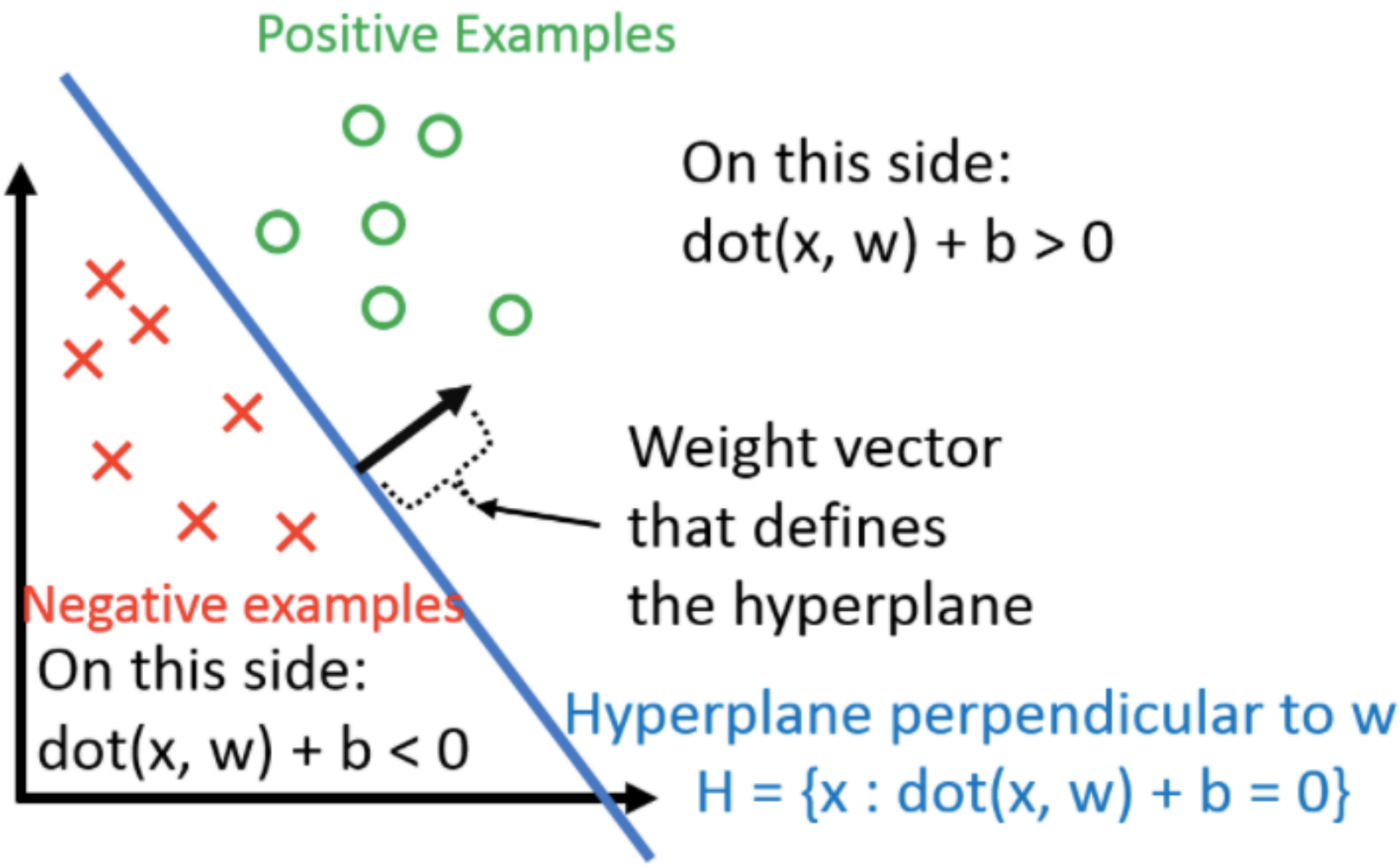
A linear classifier has the form

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$



- in 2D the discriminant is a line
- $\mathbf{w}$ is the normal to the line, and b the bias
- $\mathbf{w}$ is known as the weight vector

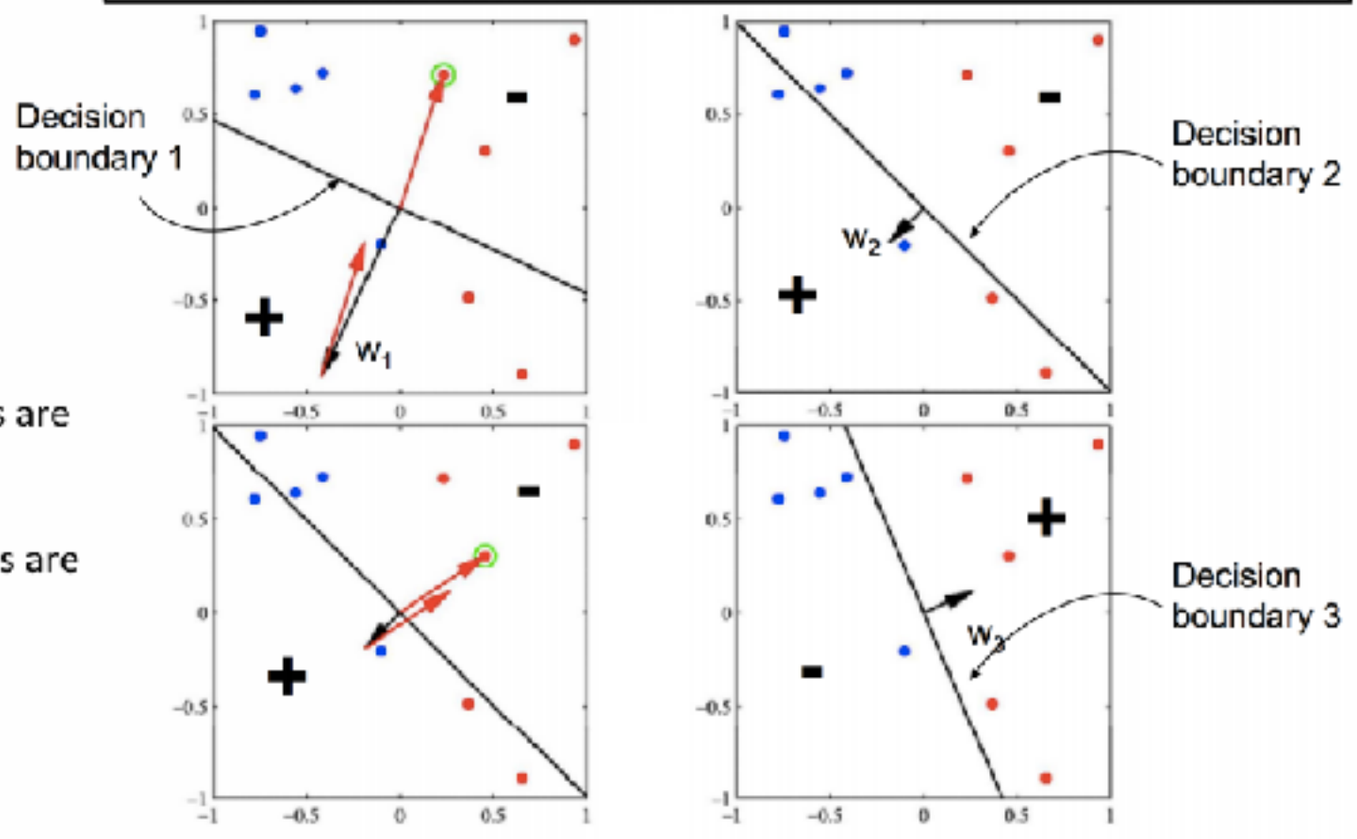
Linear Classifier :weight vector that define the hyper plane



Perceptron

$$perceptron\ loss = \sum_i \max\{0, -y_i \ w^T x_i \ \}$$

When an error is made, moves the weight in a direction that corrects the error



Red points are labeled +

Blue points are labeled -

Perceptron Algorithm

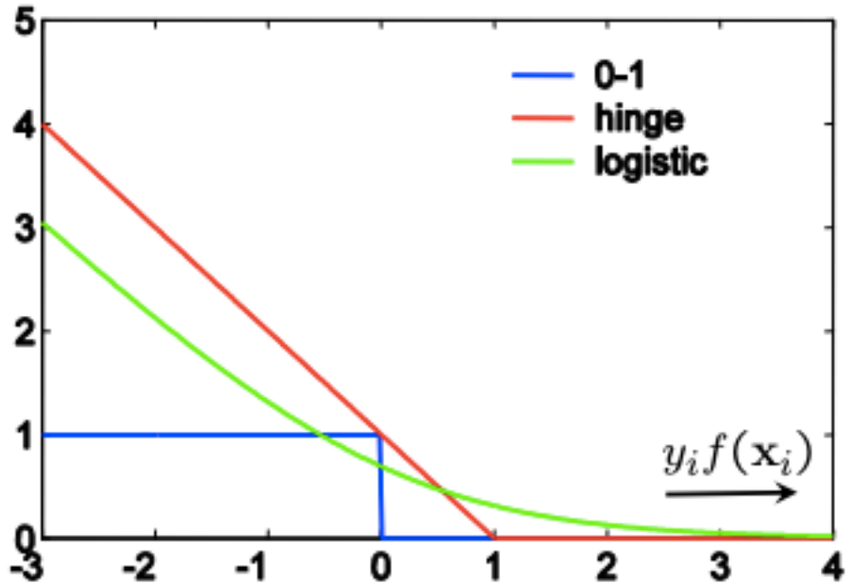
Initialize $\vec{w} = \vec{0}$
while TRUE **do**
  $m = 0$
  for $(x_i, y_i) \in D$ **do**
    if $y_i(\vec{w}^T \cdot \vec{x_i}) \leq 0$ **then**
      $\vec{w} \leftarrow \vec{w} + y\vec{x}$
      $m \leftarrow m + 1$
    end if
  end for
  if $m = 0$ **then**
    break
  end if
end while

Disadvantage : Will not converge for linearly non-separable data
Non-unique solution (few solutions have high generalisation error

Classification Loss

Zero one loss
Logistic loss
Hinged loss

Square loss ?



Next Class

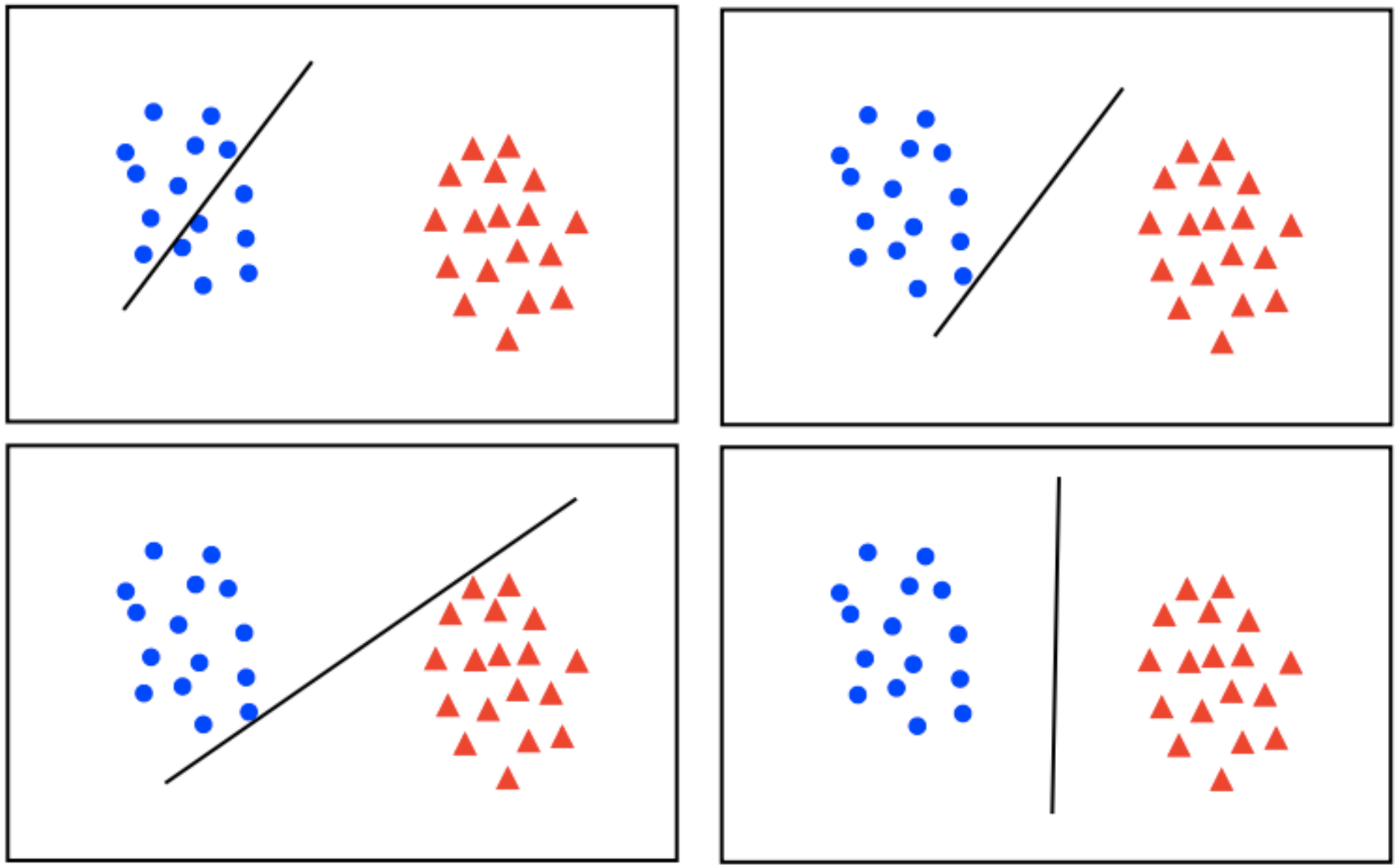
- 28/8
  - Support Vector Machine

CS4801 : Support Vector Machine

Sahely Bhadra
28/8/2017

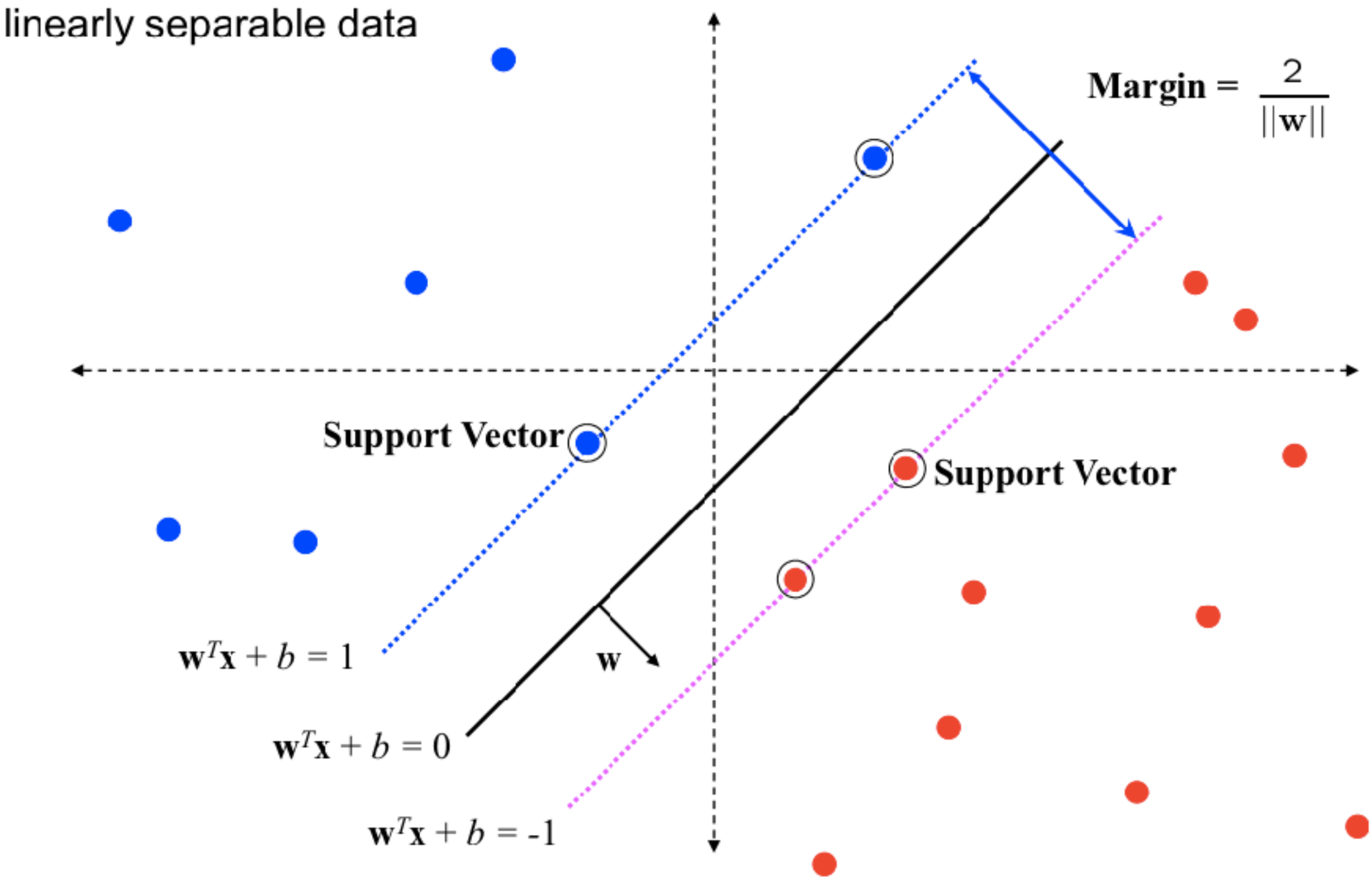
1. Margin for SVM
2. Loss function for SVM
3. Slack : linearly non separable data set
4. Pegasus : gradient based SVM solver

Maximum Margin



- maximum margin solution: most stable under perturbations of the inputs and hence better generalisation performance

Margin


linearly separable data
Margin = 2 / ||w||
Support Vector
Support Vector
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = -1$
w

Margin

- Since $\mathbf{w}^\top \mathbf{x} + b = 0$ and $c(\mathbf{w}^\top \mathbf{x} + b) = 0$ define the same plane, we have the freedom to choose the normalization of $\mathbf{w}$

- Choose normalization such that $\mathbf{w}^\top \mathbf{x}_+ + b = +1$ and $\mathbf{w}^\top \mathbf{x}_- + b = -1$ for the positive and negative support vectors respectively

- Then the margin is given by distance between two parallel line

$$\mathbf{w}^\top \mathbf{x}_+ + b = +1 \quad \text{and} \quad \mathbf{w}^\top \mathbf{x}_+ + b = -1$$

$$\frac{\mathbf{w}}{||\mathbf{w}||} \cdot (\mathbf{x}_+ - \mathbf{x}_-) = \frac{\mathbf{w}^\top (\mathbf{x}_+ - \mathbf{x}_-)}{||\mathbf{w}||}$$

$$= \frac{\mathbf{w}^\top (\mathbf{x}_+ - \mathbf{x}_-) + b - b}{||\mathbf{w}||} = \frac{+1 - -1}{||\mathbf{w}||} = \frac{2}{||\mathbf{w}||}$$

By using $\mathbf{w}^\top \mathbf{x}_+ + b = +1$

Perceptron with margin

The optimization problem becomes

$$\min_{\mathbf{w}\in\mathbb{R}^d} \quad ||\mathbf{w}||^2$$

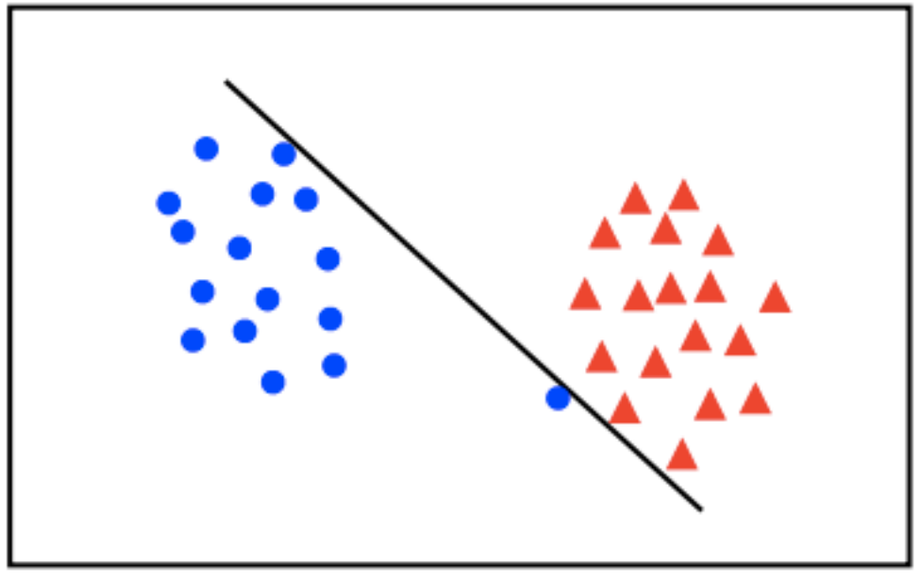
subject to

$$y_i\left(\mathbf{w}^\top \mathbf{x}_i + b\right) \geq 1 \qquad \text{for } i = 1 \ldots N$$

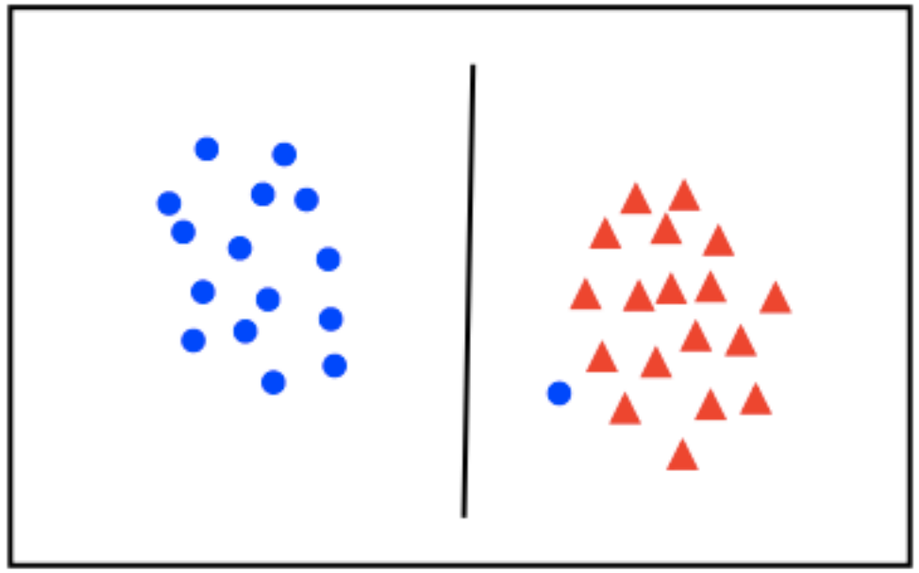
- This is now optimizing a *quadratic* function subject to *linear* constraints
- Quadratic optimization problems are a well-known class of mathematical programming problem, and many (intricate) algorithms exist for solving them (with many special ones built for SVMs)

DisAdvantage : Will not converge for linearly non-separable data

SVM: Trade off



• the points can be linearly separated but there is a very narrow margin



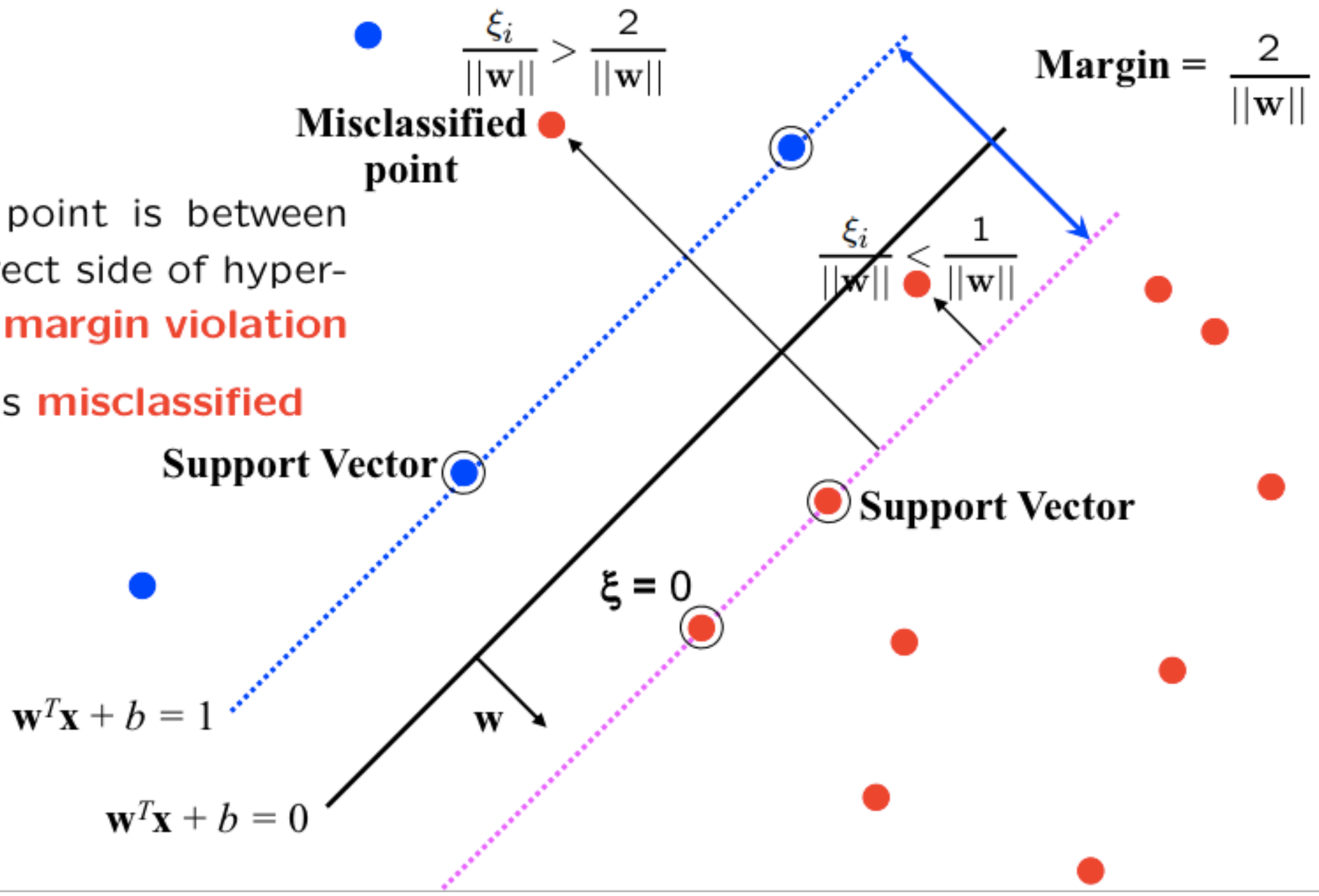
• but possibly the large margin solution is better, even though one constraint is violated

In general there is a trade off between the margin and the number of mistakes on the training data

SVM: soft margin

$$\xi_i \geq 0$$

- for $0 < \xi \leq 1$ point is between margin and correct side of hyper-plane. This is a **margin violation**
- for $\xi > 1$ point is **misclassified**



SVM: Support Vector Machine

The optimization problem becomes

$$\min_{\mathbf{w}\in\mathbb{R}^d,\xi_i\in\mathbb{R}^+} ||\mathbf{w}||^2+C\sum_i^N \xi_i$$

subject to

$$\begin{aligned} y_i\left(\mathbf{w}^\top\mathbf{x}_i+b\right) &\geq 1-\xi_i \text{ for } i=1\ldots N \\ \xi_i &\geq 0 \end{aligned}$$

- Every constraint can be satisfied if ξ_i is sufficiently large
- C is a regularization parameter:
  - small C allows constraints to be easily ignored $\rightarrow$ large margin
  - large C makes constraints hard to ignore $\rightarrow$ narrow margin
  - $C=\infty$ enforces all constraints: hard margin
- This is still a quadratic optimization problem and there is a unique minimum. Note, there is only one parameter, C.

Next Class

- 28/8
  - Solving SVM

CS4801 : Pegasos and SVM for multiclass

Sahely Bhadra
29/8/2017

1.Lossfunction of SVM
2.Pegasos :Primal Estimated sub-GrAdient SOlver for SVM
3. Imbalanced class
4.multi-class classification

SVM : imbalanced classification

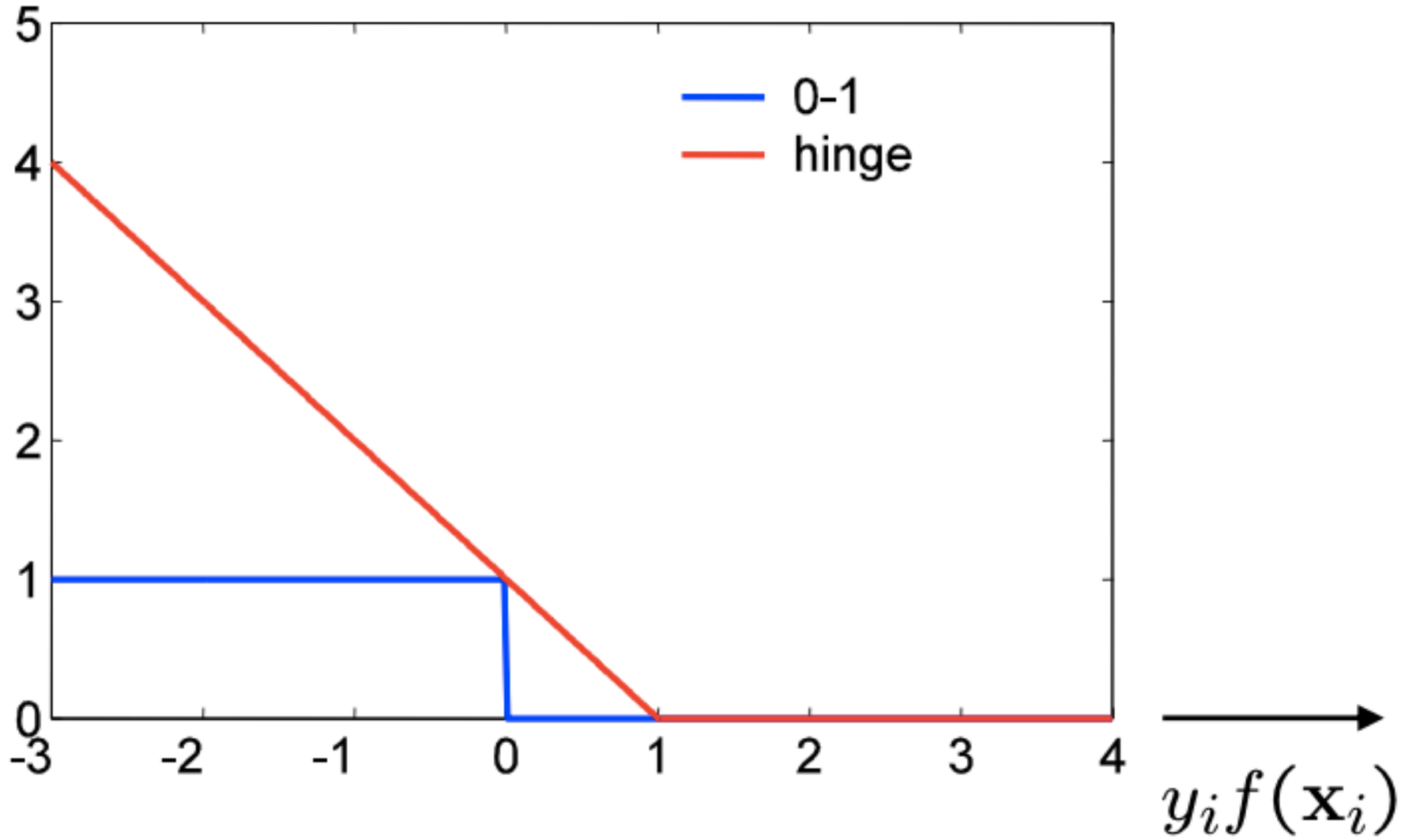
The optimization problem becomes

$$\min_{\mathbf{w}\in\mathbb{R}^d,\xi_i\in\mathbb{R}^+} ||\mathbf{w}||^2 + C_+ \sum_{i:y_i=+1}^{N} \xi_i + C_- \sum_{i:y=-1}^{N} \xi_i$$

subject to

$$y_i\left(\mathbf{w}^\top \mathbf{x}_i + b\right) \geq 1 - \xi_i \text{ for } i = 1 \dots N$$
$$\xi_i \geq 0$$

SVM: maximize margin



- SVM uses "hinge" loss $\max(0, 1 - y_i f(\mathbf{x}_i))$
- an approximation to the 0-1 loss

Pegasus : Gradient based SVM solver

$$\mathcal{C}(\mathbf{w}) = \frac{1}{N}\sum_{i}^{N}\left(\frac{\lambda}{2}||\mathbf{w}||^2 + \mathcal{L}(\mathbf{x}_i, y_i; \mathbf{w})\right)$$

The iterative update is

$$\begin{aligned}\mathbf{w}_{t+1} &\leftarrow \mathbf{w}_t - \eta\nabla_{\mathbf{w}_t}\mathcal{C}(\mathbf{w}_t) \\ &\leftarrow \mathbf{w}_t - \eta\frac{1}{N}\sum_{i}^{N}(\lambda\mathbf{w}_t + \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{x}_i, y_i; \mathbf{w}_t))\end{aligned}$$

where η is the learning rate.

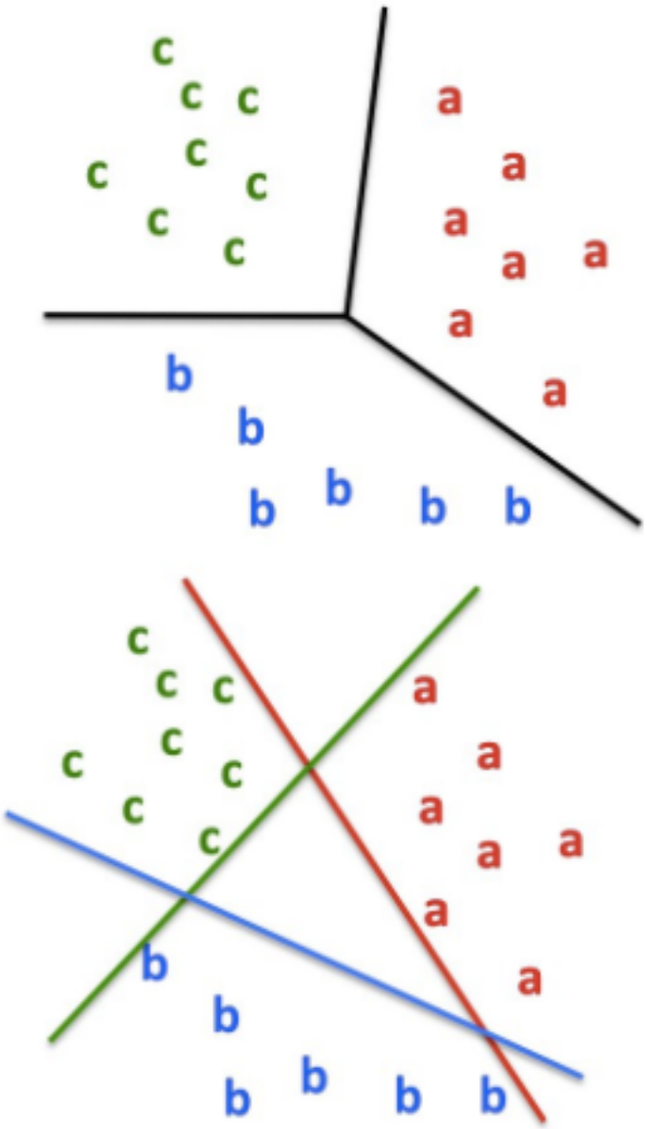
Then each iteration t involves cycling through the training data with the updates:

$$\begin{aligned}\mathbf{w}_{t+1} &\leftarrow \mathbf{w}_t - \eta(\lambda\mathbf{w}_t - y_i\mathbf{x}_i) && \text{if } y_i f(\mathbf{x}_i) < 1 \\ &\leftarrow \mathbf{w}_t - \eta\lambda\mathbf{w}_t && \text{otherwise}\end{aligned}$$

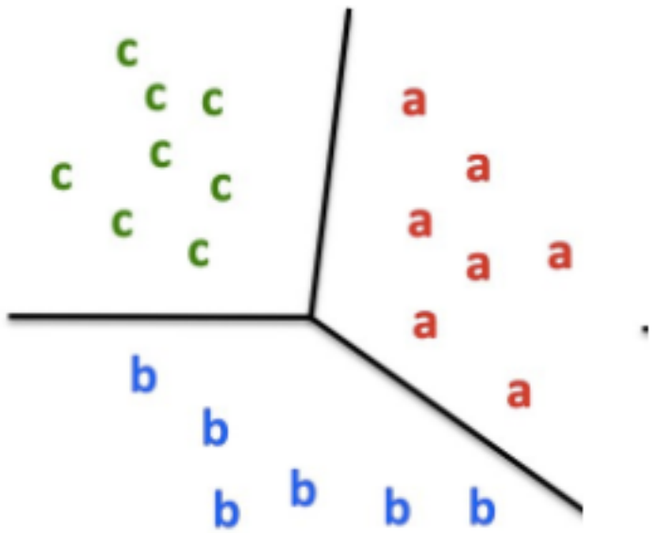
In the Pegasos algorithm the learning rate is set at $\eta_t = \frac{1}{\lambda t}$

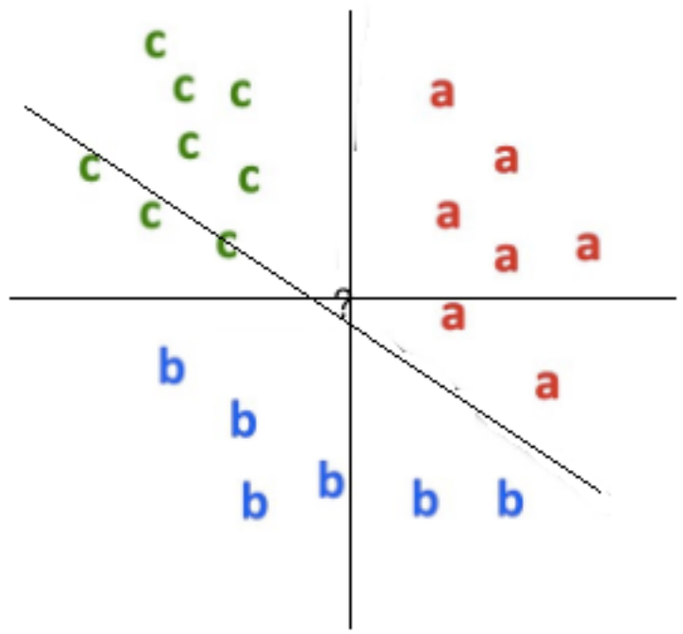
Multi-class vs. Binary classification

- Multi-class:
  - classes mutually exclusive:
    - instance is either a or b or c
    - even if it's an outlier
  - NB, kNN, DT, logistic
- Binary:
  - one-vs-rest:
    - {a} vs {not a}, {b} vs {not b}
  - classes may overlap
    - instance can be both a and b
    - can be in none of the classes
  - SVM, logistic, perceptron



Multi-class vs. Binary classification





- Binary:
  - one-vs-one:
    - {a} vs {b} , {b} vs {c}
  - classes may overlap
    - instance can be both a and b

Next Class

- 28/8
  - Dual SVM

CS4801 : Dual SVM and Non-linear SVM

Sahely Bhadra
30/8/2017

1. Dual SVM
2. Kernel / Non linear SVM

SVM: Recap

- Primal SVM :

The optimization problem becomes

$$\min_{\mathbf{w}\in\mathbb{R}^d,\xi_i\in\mathbb{R}^+} ||\mathbf{w}||^2 + C_+\sum_{i:y_i=+1}^{N}\xi_i + C_-\sum_{i:y=-1}^{N}\xi_i$$

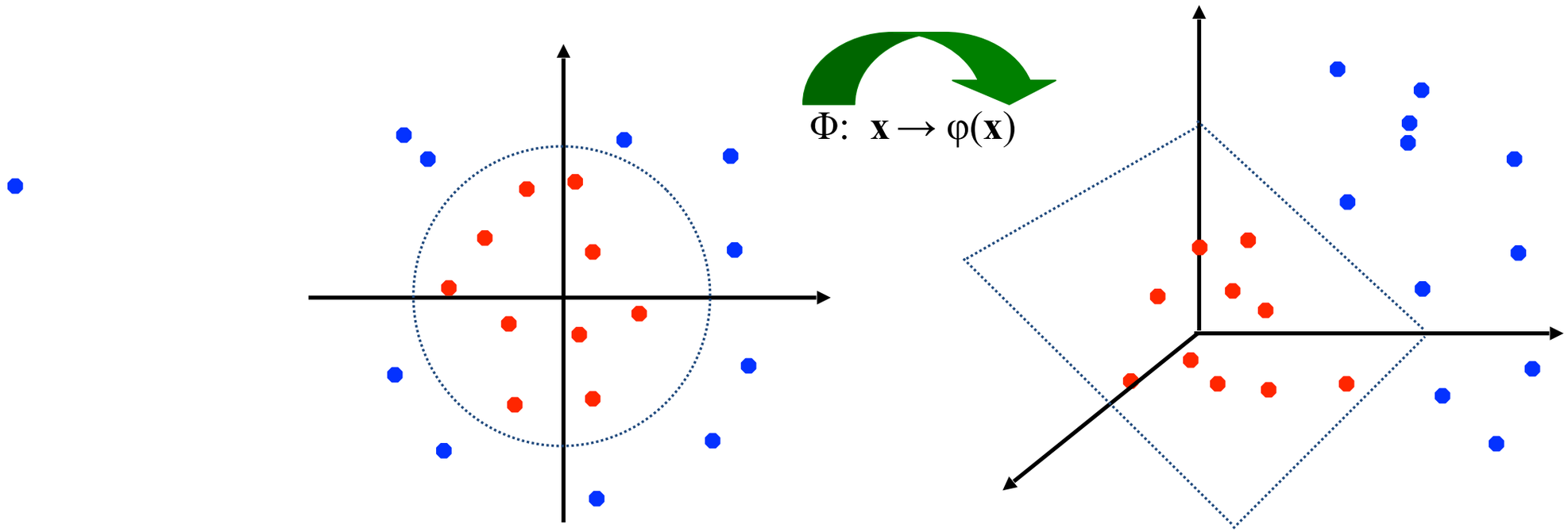
subject to

$$y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right) \geq 1-\xi_i \text{ for } i = 1\ldots N$$
$$\xi_i \geq 0$$

- Pegasos : Primal Estimated sub-GrAdient SOlver for SVM (primal)
- Support vectors : point which are on supportive hyper plan and points for which are wrong side of supportive hyperplane.
- Multi-class classification with SVM :
  - Multiclass SVM(not covered in class)
  - one vs one
  - one vs rest

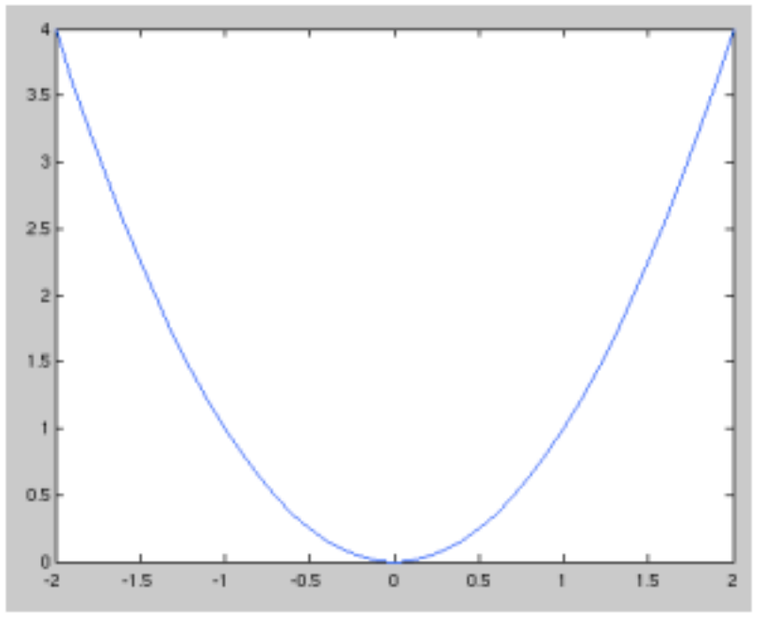
SVM: non-linear classification

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable



Lagrangian Dual

$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq b$$



Moving the constraint to objective function
Lagrangian:

$$L(x, \alpha) = x^2 - \alpha(x - b)$$
$$\text{s.t.} \quad \alpha \geq 0$$

Solve:

$$\min_x \max_\alpha \; L(x, \alpha)$$
$$\text{s.t.} \quad \alpha \geq 0$$

Lagrangian Dual of SVM

SVM Primal :

$$\begin{aligned} &\text{minimize}_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}.\mathbf{w} + C\sum_j \xi_j \\ &\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 - \xi_j, \ \forall j \\ &\xi_j \geq 0, \ \forall j \end{aligned}$$

SVM Dual :

$$\begin{aligned} &\text{maximize}_{\alpha} \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \mathbf{x}_j \\ &\sum_i \alpha_i y_i = 0 \\ &C \geq \alpha_i \geq 0 \end{aligned}$$

Solved using SMO or

Dual co-ordinate Gradient Ascent

(not covered in details in class)

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = y_k - \mathbf{w}.\mathbf{x}_k$$

for any k where $C > \alpha_k > 0$

No explicit feature but dot products

SVM Dual :

$$\text{maximize}_\alpha \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \mathbf{x}_j$$

$$\sum_i \alpha_i y_i = 0$$
$$C \geq \alpha_i \geq 0$$

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = y_k - \mathbf{w}.\mathbf{x}_k$$

for any k where $C > \alpha_k > 0$

Classifier :

$$sign\,(\mathbf{w} \cdot \mathbf{x} + b)$$

$$sign\,(\sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + y_k - \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x}_k)$$

Kernel

- Similarity measure
- dot product in some feature vectors in some Hilbert Space

$$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

- kernel **Gram matrix** should be always positive semidefinite

Given a kernel k and a set of n points $x_1, \ldots, x_n \in \mathcal{X}$ the $n \times n$ matrix

$$K = (k(x_i, x_j))_{ij},$$

is called the **kernel matrix** (or **Gram Matrix**) K of the kernel k with respect to $x_1, \ldots, x_n$.

- PSD for any vector 'y' if $y^T K y >= 0$ then K is PSD matrix
- THEORY : RKHS Reproducing kernel Hilbert Space (not covered in class)

Mercer's Condition

There exists a mapping ϕ and an expansion

$$K(\mathbf{x},\mathbf{y}) = \sum \phi(\mathbf{x})_i \phi(\mathbf{y})_i$$

if and only if, for any $g(\mathbf{x})$ such that $\int g(\mathbf{x})^2 d\mathbf{x}$ is finite, then $\int K(\mathbf{x},\mathbf{y}) g(\mathbf{x}) g(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0$

Hilbert space

Hilbert Space

- A vector space *H* endowed with an inner product and associated norm and metric such that every Cauchy sequence in *H* has a limit in *H*

Example of a Hilbert Space: A Euclidean space $\Re^n$

- A vector space
- Has an inner product
  - Dot product $< x, y >= x^T y$
- Has the norm $\|x\| = \sqrt{x^T x} = \sqrt{< x, x >}$
- Has metric $\|x - y\|$

Kernel: Positive semidefinite

Let k, k_1 and k_2 be positive definite kernels on $\mathcal{X} \times \mathcal{X}$.

i) *for any $\alpha \geq 0$, $k(x,y) = \alpha\, k_1(x,y)$ is positive definite,*

ii) $k(x,y) = k_1(x,y) + k_2(x,y)$ *is positive definite (***pointwise addition***),*

iii) $k(x,y) = k_1(x,y)k_2(x,y)$ *is positive definite (***pointwise multiplication***),*

iv) *the* **pointwise limit** *k of a sequence of positive definite kernels k_n on $\mathcal{X} \times \mathcal{X}$ is positive definite,*

v) *for any $f : \mathcal{X} \to \mathbb{R}$, $k'(x,y) = f(x)f(y)k(x,y)$ is positive definite, especially $k(x,y) = f(x)f(y)$,*

vi) *for any $\phi : \mathcal{X} \to \mathcal{H}$ where $\mathcal{H}$ is a dot product space, $k(x,y) = \langle \phi(x), \phi(y) \rangle$ is positive definite,*

Few Examples

- Linear kernel $K(\mathbf{u}, \mathbf{v}) = \mathbf{u} \cdot \mathbf{v}$

- Polynomial kernel $K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v} + 1)^d$

- Gaussian kernel $K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{||\mathbf{u} - \mathbf{v}||}{2\sigma^2}\right)$

- Sigmoid $K(\mathbf{u}, \mathbf{v}) = \tanh(\eta \mathbf{u} \cdot \mathbf{v} + \nu)$

Kernel in logistic regression

$$P(Y = 1 \mid x, \mathbf{w}) = \frac{1}{1 + e^{-(\mathbf{w}\cdot\Phi(\mathbf{x})+b)}}$$

- Define weights in terms of support vectors:

$$\mathbf{w} = \sum_i \alpha_i \Phi(\mathbf{x}_i)$$

$$P(Y = 1 \mid x, \mathbf{w}) = \frac{1}{1 + e^{-(\sum_i \alpha_i \Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x})+b)}}$$

$$= \frac{1}{1 + e^{-(\sum_i \alpha_i K(\mathbf{x},\mathbf{x}_i)+b)}}$$

- Derive simple gradient descent rule on α_i

Evaluation of Classification

Hypothesized class	True class: p	True class: n
Y	True Positives	False Positives
N	False Negatives	True Negatives
Column totals:	P	N

$$\text{fp rate} = \frac{FP}{N} \qquad \text{tp rate} = \frac{TP}{P}$$

$$\text{precision} = \frac{TP}{TP+FP} \qquad \text{recall} = \frac{TP}{P}$$

$$\text{accuracy} = \frac{TP+TN}{P+N}$$

$$\text{F-measure} = \frac{2}{1/\text{precision}+1/\text{recall}}$$

Fig. 1. Confusion matrix and common performance metrics calculated from it.

Receiver Operating Characteristics

$$tp\ rate \approx \frac{\text{Positives correctly classified}}{\text{Total positives}}$$

$$fp\ rate \approx \frac{\text{Negatives incorrectly classified}}{\text{Total negatives}}$$

$$\text{sensitivity} = \text{recall}$$

$$\text{specificity} = \frac{\text{True negatives}}{\text{False positives} + \text{True negatives}} = 1 - fp\ rate$$

$$\text{positive predictive value} = \text{precision}$$

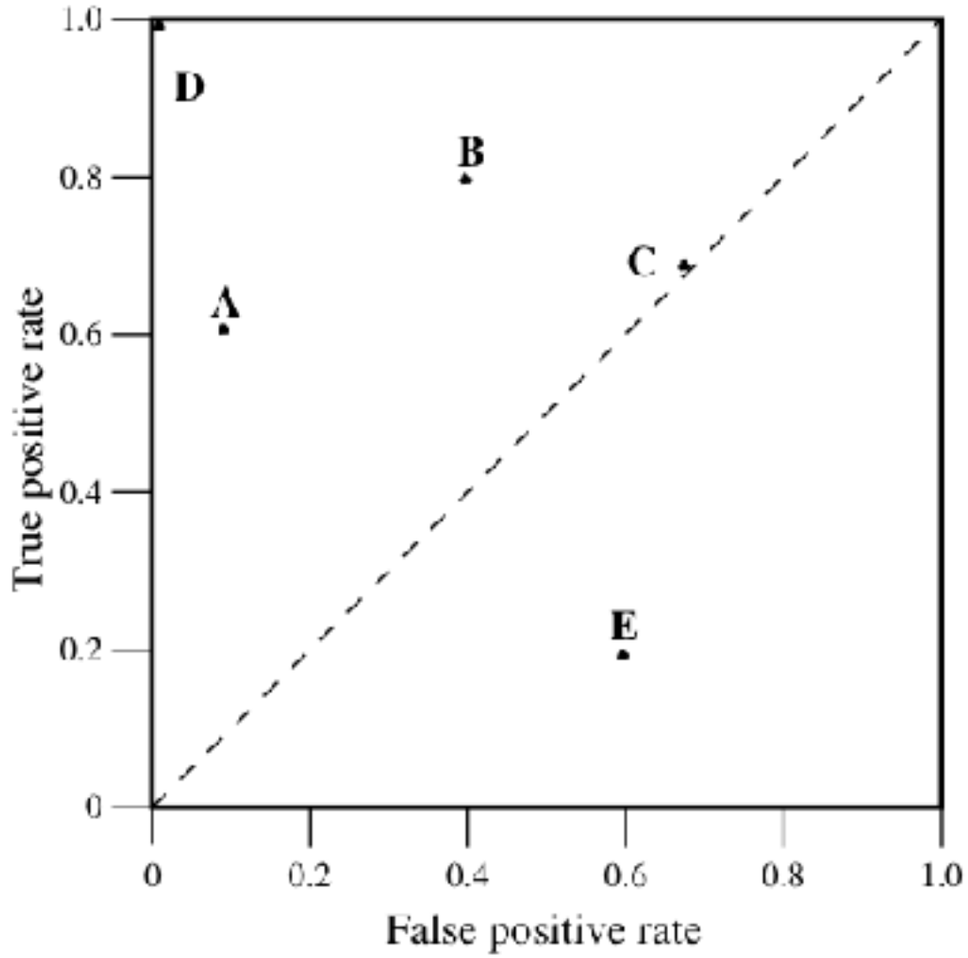
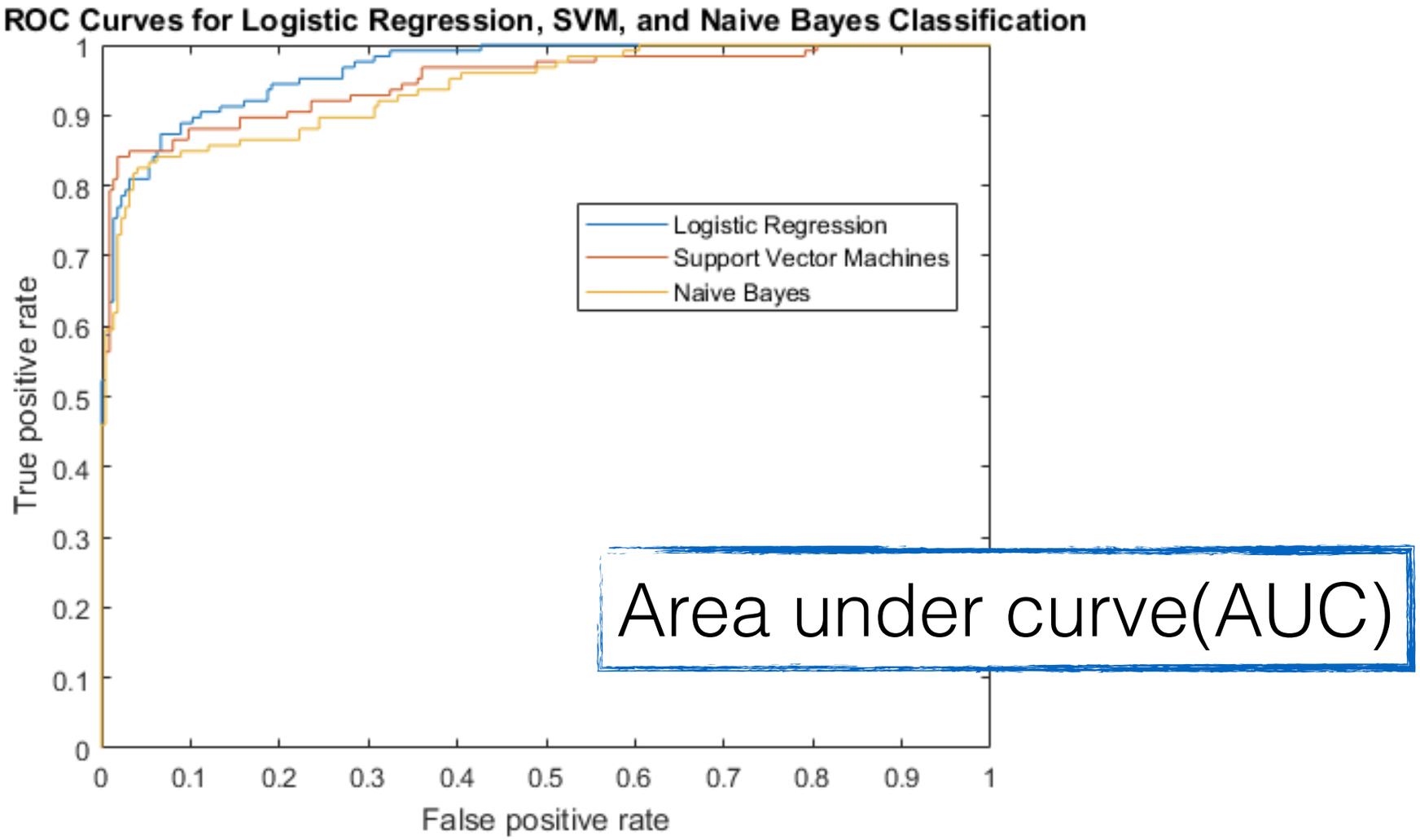


Fig. 2. A basic ROC graph showing five discrete classifiers.

Receiver Operating Characteristics

$$tp\ rate \approx \frac{\text{Positives correctly classified}}{\text{Total positives}}$$

$$fp\ rate \approx \frac{\text{Negatives incorrectly classified}}{\text{Total negatives}}$$



Area under curve(AUC)

Feature scaling

- feature scaling

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- scaling to unite vector

$$x' = \frac{x}{||x||}$$

$$\tilde{K}(x, y) = \frac{K(x, y)}{\sqrt{K(x, x)K(y, y)}} \in \mathbb{R}$$

- standardisation

$$x' = \frac{x - \bar{x}}{\sigma}$$

- Scale both training and testing data with same scaling factor.

Few more Supervised learning

- We will come back later
  - Decision Tree
  - Artificial Neural Network
- We will not cover
  - Semi-supervised Learning
  - Learning Ranks , Ordinal Regression
  - Reinforcement Learning

Feature scaling

6/9/2017 onward

- Unsupervised Learning
  - Clustering
    - k means and gaussian mixture model
  - Dimentionality reduction
    - PCA,
    - CCA,
    - LDA