Classifiers !

Ορισμοί:

Έστω ότι έχουμε $\vec{x}$ είσοδος του αλγορίθμου.

- x_j είναι j-th στοιχείο του $\vec{x}$
- x_j είναι η τιμή του διανύσματος για $j = 1, \dots d$ όπου d οι διαστάσεις του διανύσματος.

Έστω $f(\vec{x})$ η τιμή για το target της εισόδου $\vec{x}$

- η ακριβής μορφή της $f(x)$ είναι άγνωστη
- χρησιμοποιούμε δεδομένα ενός συνόλου $D = \{\vec{x}, f(\vec{x})\}$ το οποίο είναι γνωστό ώστε να προσπαθήσουμε να βρούμε μία καλή/ελαρμή αναπαράσταση της πραγματικής $f(x)$

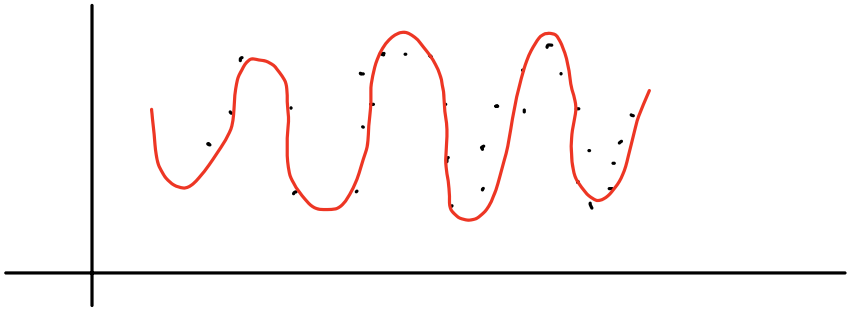
Σκοπός

δημιουργία ενός mapping από το $\vec{x}$ στο $f(\vec{x})$

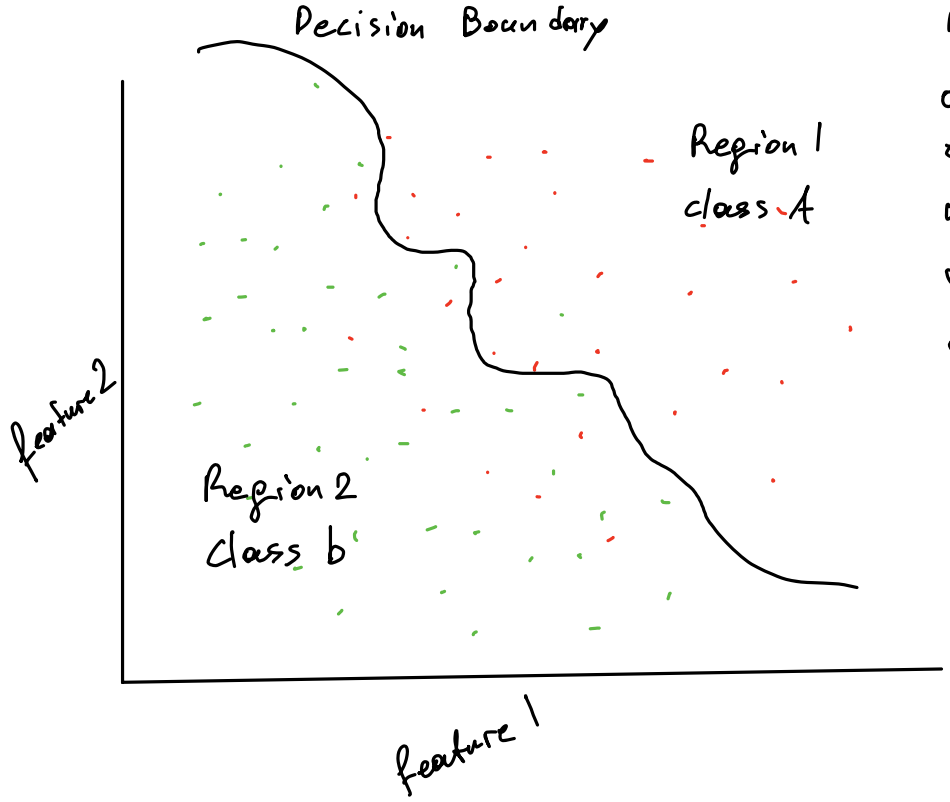
$$h(\vec{x}, \theta) \simeq f(\vec{x}) \quad \forall x_j : j \in \{1 \dots d\}$$

θ είναι οι παράμετροι που έχουμε βρεί μέσα από τη διαδικασία του classification.

Παράδειγμα

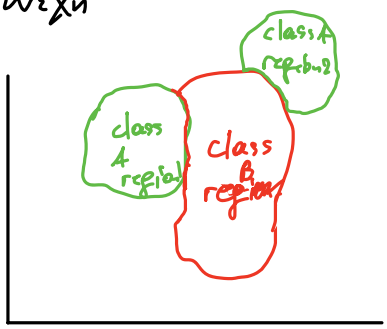


Χρήση σε Διακριτά Προβλήματα — Θέλουμε να βρούμε έναν κανόνα / χαρτογράφηση ο οποίος δίνοντάς του στις τιμές των feature 1 και feature 2 αντίστοιχα να μπορεί να μας πει αν περιμένουμε να δούμε κάποιαν κουκίδα (κλάση A) ή πράσινη κουκίδα (κλάση B)



Classification in Euclidean Space

- Η διαδικασία του classification είναι ο χωρισμός ενός ευκλείδιου χώρου N-διαστάσεων σε επιμέρους χωρία τα οποία:
  - καθε χωριό έχει διαφορετικό label (αντιστοιχεί δηλαδή σε διαφορετικό class)
  - Χωρία τα οποία αντιστοιχούν στο ίδια label δεν πρέπει να είναι συνεχή



- για κάθε καινούρια είσοδο $\vec{x}_{new}$ θα πρέπει ο classifier να εντοπίζει σωστά το region στο οποίο βρίσκεται το $\vec{x}_{new}$

- Decision boundaries: τα σύνορα μεταξύ των διαφορων regions

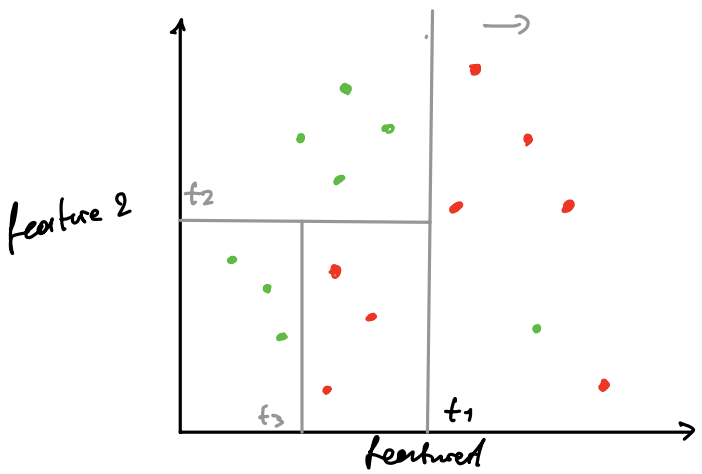
  Ο κάθε classifier χαρακτηρίζεται από τις ευθύνες των decision boundaries που δημιουργεί

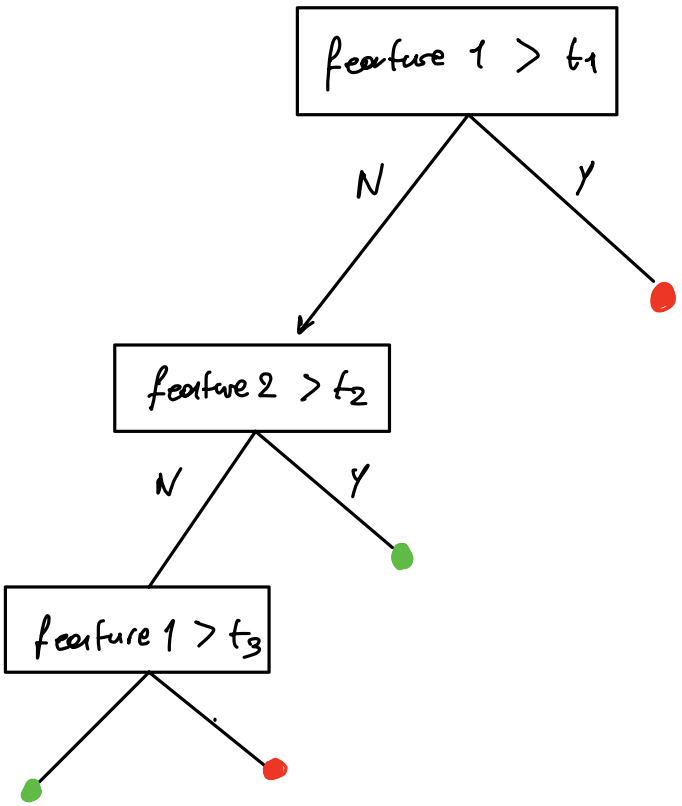
Απλή Εφαρμογή Classifier : Decision Trees

Το decision tree είναι μία απλή μορφή classifier η οποία χρησιμοποιώντας ευθείες γραμμές παράλληλες στους άξονες $x, y, z \ldots$ χωρίζει τον $\mathbb{R}^n$ χώρο που είχαμε σε χωρία, σε κάθε ένα εκ των οποίων αναθέτει μία τιμή κλάσης (label)

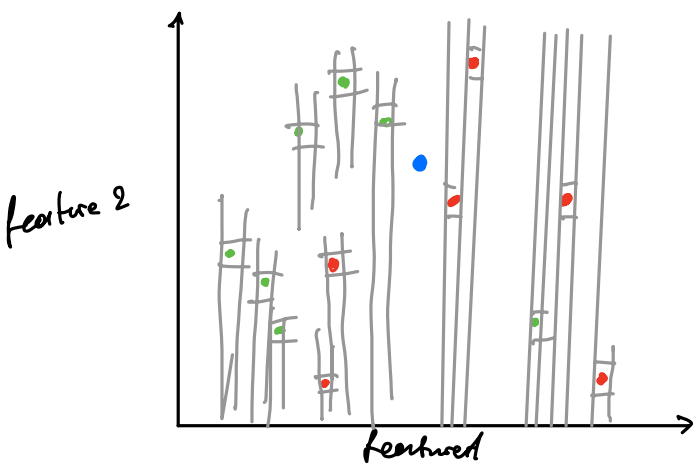
Ο τρόπος ανάθεσης στην περίπτωση του decision tree είναι μιας μορφής: $x_j > t$, όπου το t είναι ένα optimizable arbitrary value

Η διαδικασία αυτή μπορεί να δημιουργείται μέχρι $N-1$ χωρία για N δεδομένα.





Overfitting problem



training accuracy: 100%

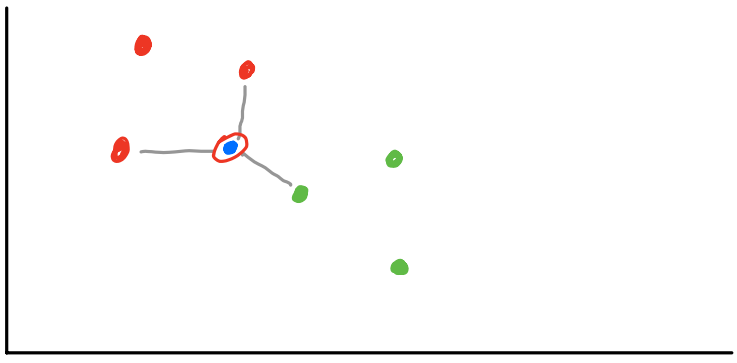
testing accuracy: very low

Thumb rule: τα χαρία δεν πρέπει να είναι πολύ σαφή στην στρατηγική ρίζα των στοιχείων που έχουμε

Classifier k-Nearest Neighbors

Υπόθεση: similar inputs give similar outputs

Κανόνας: για ένα test input x, δίνουμε την πιο κοινή τιμή των neighboring datapoints



Μαθηματικός ορισμός kNN

- test point x
- ορίζουμε ένα σύνολο k στοιχείων τα οποία έχουν την ελάχιστη Ευκλείδεια απόσταση από το x, S_x

$$S_x \subset D \quad : \quad |S_x| = k$$

$\downarrow$

το σύνολο των αρχικών datapoint

↳ δηλαδή είπαμε ότι θέλουμε να χρησιμοποιούμε k γείτονες

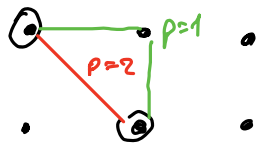
```
k>3
min1 = +∞
min2 = +∞
min3 = +∞
for i ∈ 1..|D|
    if dist(x_t, x_i) < min1
        min1 ← x1
        :
```

Distance function

$$dist(x, z) = \left(\sum_{r=1}^{d} |x_r - z_r|^p \right)^{1/p}$$

$p = 2$: Euclidean distance

$p = 1$: Manhattan distance



Principal Component Analysis - PCA

y	f_1	f_2	f_3	...	f_{145}	...	f_d
y_1							
y_2							
⋮							
y_n							

Θέλουμε γενικά $d \leq \sqrt{n}$

1. data matrix X $n \times d$

2. Κανονικοποίηση των data

$$X'_{ij} = \frac{X_{ij} - \mu_j}{\sigma_j}$$

(μ_j → mean, σ_j → std)

3. Covariance Matrix Σ

$$\Sigma = \frac{1}{n-1} X^T X$$

4. Ιδιοτιμές & Ιδιοδιανύσματα

$$\Sigma \vec{v} = \lambda \vec{v}$$

λ: ιδιοτιμή που αντιστοιχεί στο ιδιοδιάνυσμα $\vec{v}$

5. Ταξινόμηση των ιδιοτιμών από τη μεγαλύτερη στη μικρότερη