IRDS: Bonus Slides

Charles Sutton
University of Edinburgh

Hello there

I will not present these slides in class.

Next lecture we will discuss how to choose features for learning algorithms.

This means you need to understand a bit about learning algorithms.

There are just an outline of topics that will help you to appreciate the next lecture.

These slides:

- List a few representative algorithms
- What you should know about them
- With links to readings to learn about them

To be ready for the next lecture, what you really need:

- to know how the classifiers represent the decision boundary
- *not* the algorithm for how the classifier is learnt
    - (good to know, but not necessary for next lecture)

List of Algorithms

(with readings)

Here are the ones we will “discuss”

- Linear regression
  - Fitting nonlinear functions by adding basis functions
  - BRML Sec 17.1, 17.2
- Logistic regression
  - BRML Sec 17.4
  - (just first few pages, don’t worry about training algorithms)
- k-nearest neighbour
  - BRML Sec 14.1, 14.2
- Decision trees
  - HTF Sec 9.2

Why these?

- practical
- have different types of decision boundaries
  - so representative for purposes of next lecture

Key to previous slide

- BRML : Barber. *Bayesian Reasoning and Machine Learning.* CUP, 2012. http://web4.cs.ucl.ac.uk/staff/D.Barber/pmwiki/pmwiki.php?n=Brml.HomePage

- HTF : Hastie, Tibshirani, and Friedman. The Elements of Statistical Learning 2nd ed, Springer, 2009. http://statweb.stanford.edu/~tibs/ElemStatLearn/

Linear regression

Let $\mathbf{x} \in \mathbb{R}^d$ denote the feature vector. Trying to predict $y \in \mathbb{R}$

Simplest choice a linear function. Define parameters $\mathbf{w} \in \mathbb{R}^d$

$$\hat{y} = f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^\top \mathbf{x} = \sum_{j=1}^{d} w_j x_j$$

(to keep notation simple assume that always $x_d = 1$)

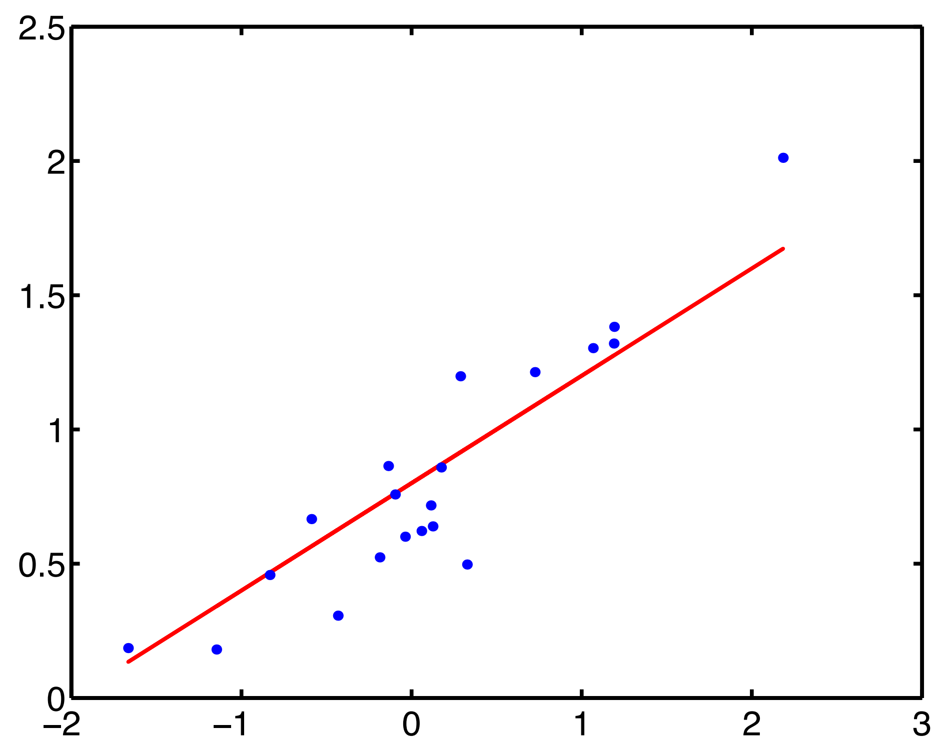
Given a data set

$$\mathbf{x}^{(1)} \dots \mathbf{x}^{(N)}, y^{(1)}, \dots, y^{(N)}$$

find the best parameters

$$\min_{\mathbf{w}} \sum_{i=1}^{N} \left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)} \right)^2$$

which can be solved easily
(but I won't say how)



Nonlinear regression

What if we want to learn a nonlinear function?

Trick: Define new features, e.g., for scalar x, define $\phi(x) = (1, x, x^2)^\top$

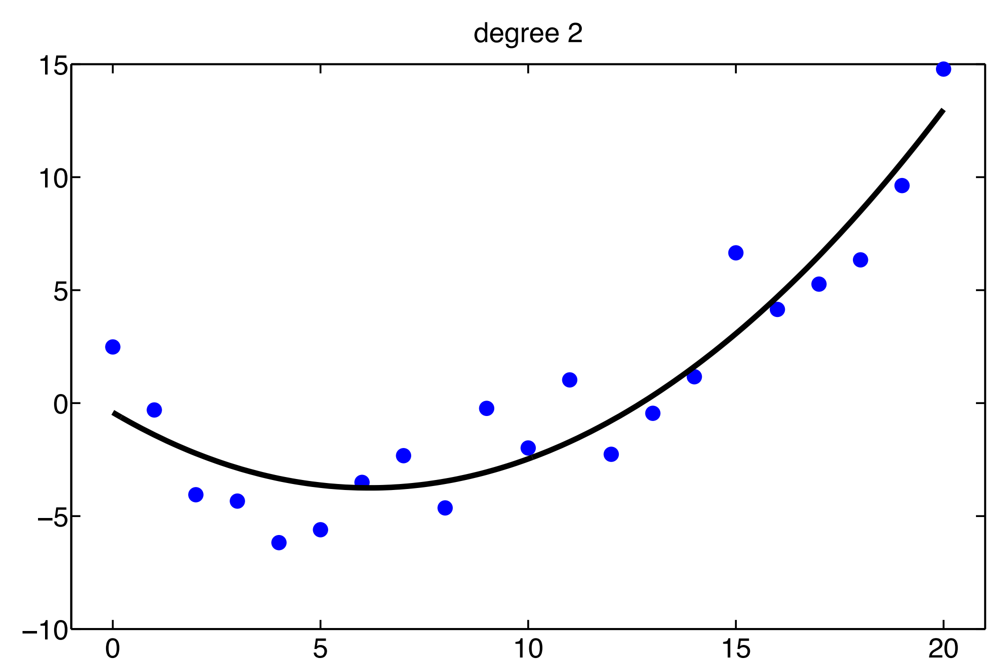
$$\hat{y} = f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^\top \phi(\mathbf{x})$$

this is still linear in $\mathbf{w}$

To find parameters,
the minimisation problem is now

$$\min_{\mathbf{w}} \sum_{i=1}^{N} \left(y^{(i)} - \mathbf{w}^\top \phi(\mathbf{x}^{(i)}) \right)^2$$

exactly the same form as before
(because $\mathbf{x}$ is fixed)
so still just as easy



Logistic regression

(a classification method, despite the name)

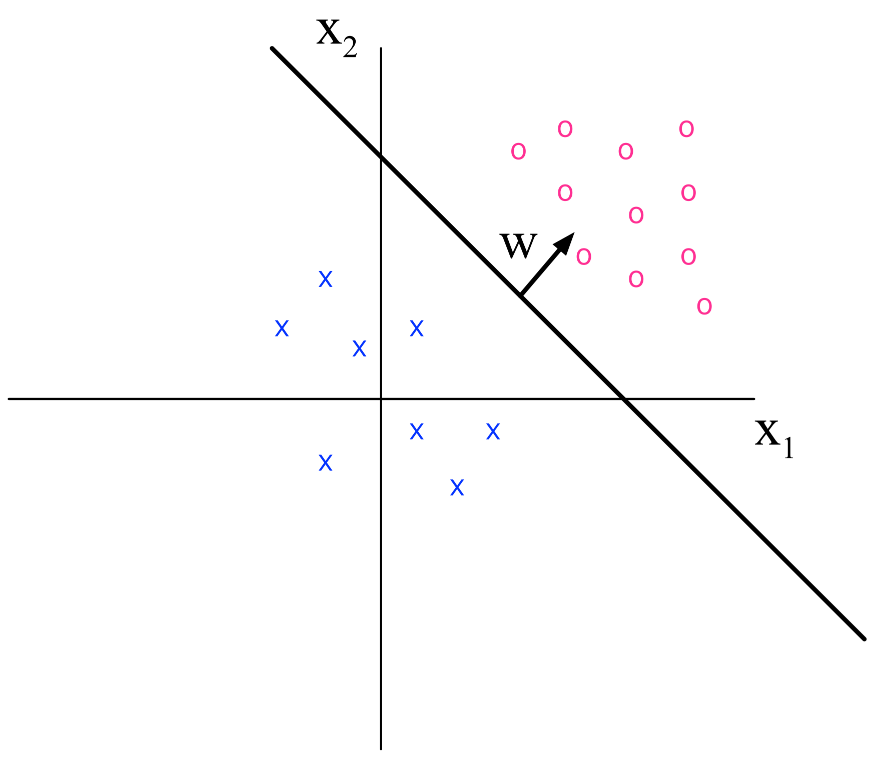
Linear regression was easy.
Can we do linear classification too?

Define a discriminant function

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

Then predict using

$$y = \begin{cases} 1 & \text{if } f(\mathbf{x}, \mathbf{w}) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$



yields linear decision boundary

Can get class probabilities from this idea, using *logistic regression:*

$$p(y = 1|\mathbf{x}) = \frac{1}{1 + \exp\{-\mathbf{w}^\top \mathbf{x}\}}$$

(to show decision boundaries same, compute log odds $\log \dfrac{p(y=1|\mathbf{x})}{p(y=0|\mathbf{x})}$

K-Nearest Neighbour

simple method for classification or regression

Define a distance function between feature vectors $D(\mathbf{x}, \mathbf{x}')$

To classify a new feature vector $\mathbf{x}$

1. Look through your training set. Find the *K* closest points. Call them $N_K(\mathbf{x})$

   (this is **memory-based** learning.)

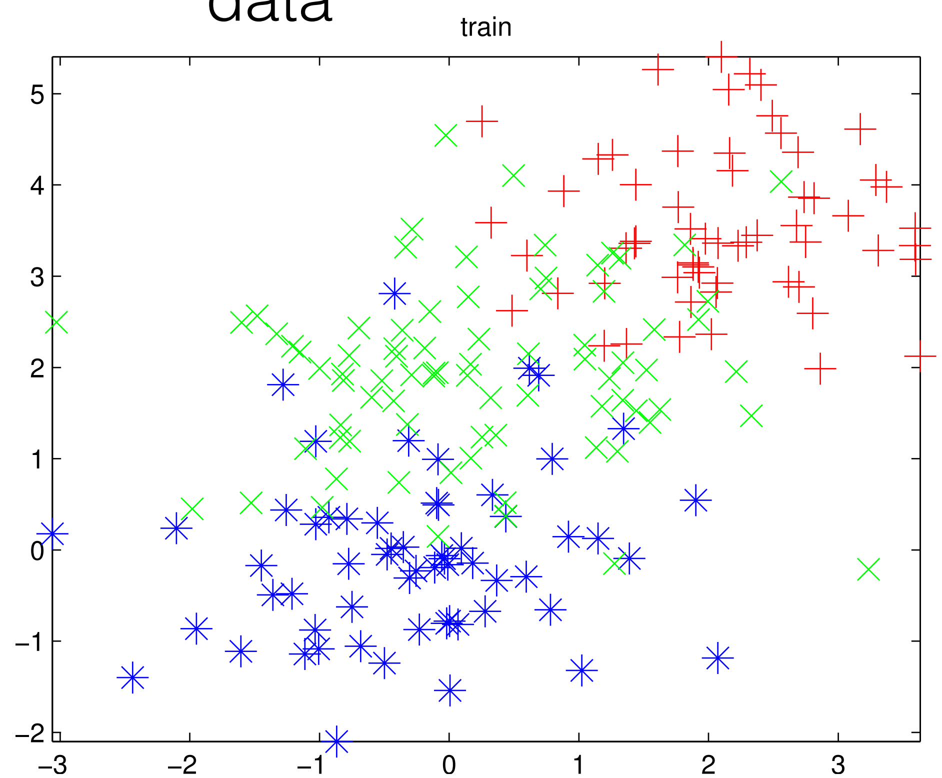
2. Return the majority vote.
3. If you want a probability, take the proportion

$$p(y = c|\mathbf{x}) = \frac{1}{K} \sum_{(y', \mathbf{x}') \in N_K(\mathbf{x})} \mathbb{I}\{y' = c\}$$

(the running time of this algorithm is terrible. See IAML for better indexing.)

K-Nearest Neighbour

data

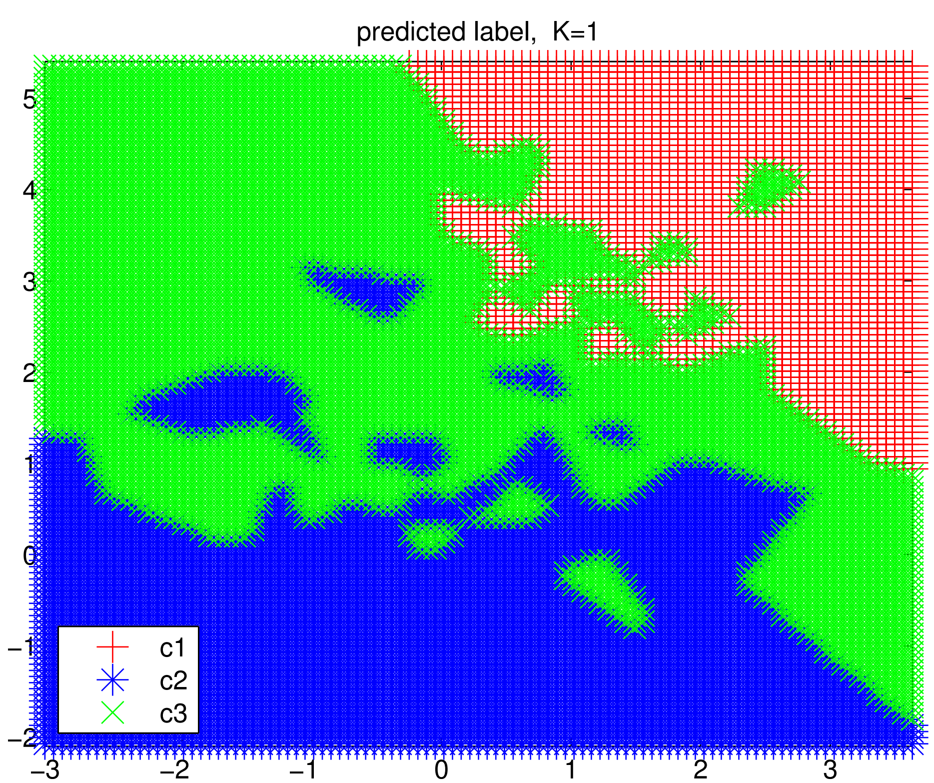


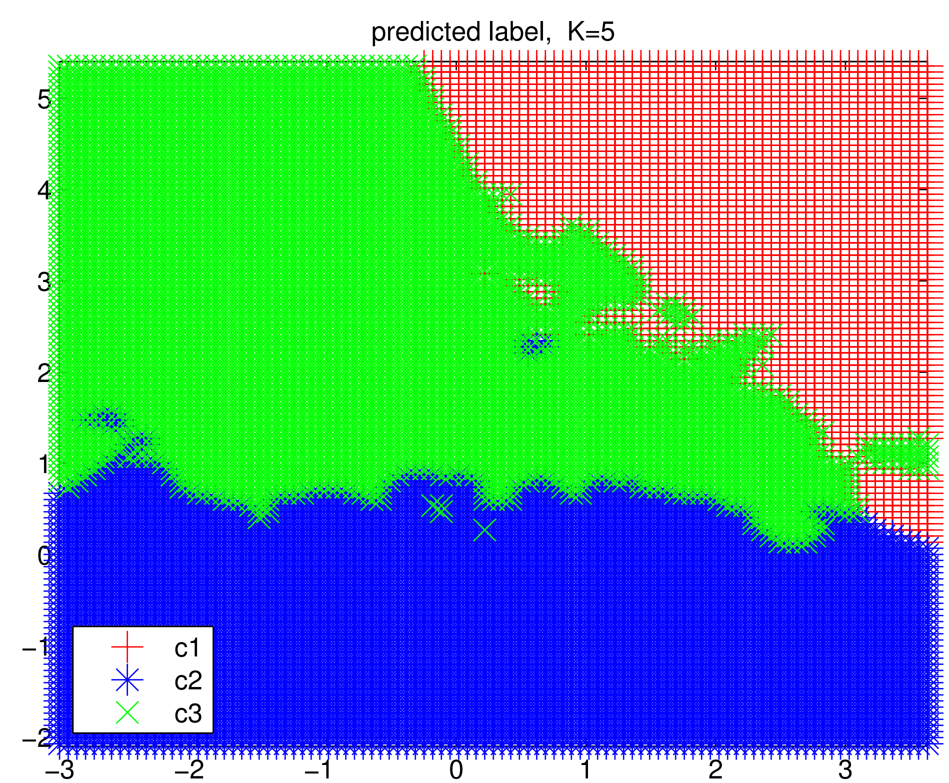
Decision boundaries can be highly nonlinear

The bigger the K, the smoother the boundary

This is **nonparametric**: the complexity of the boundary varies depending on the amount of training data

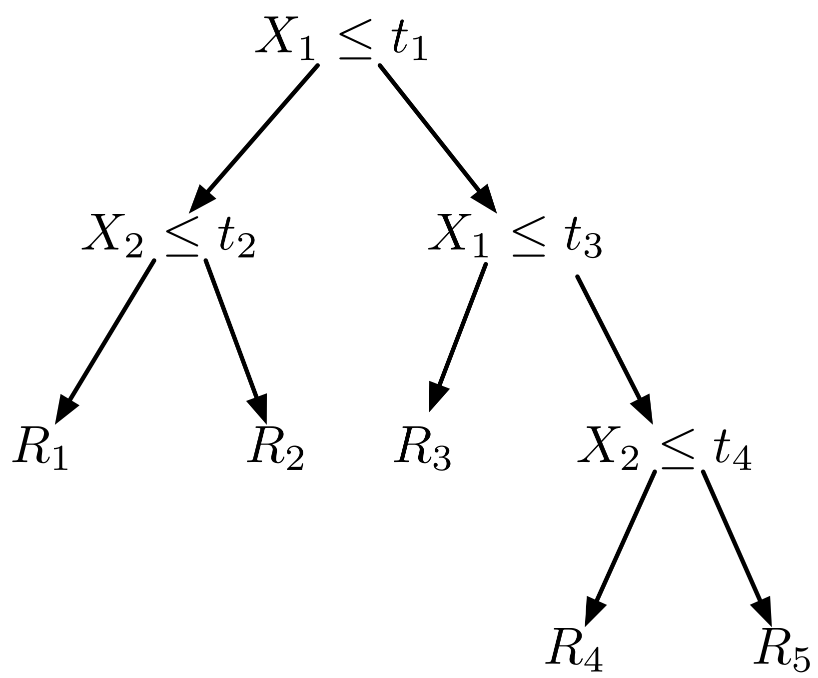
k=1

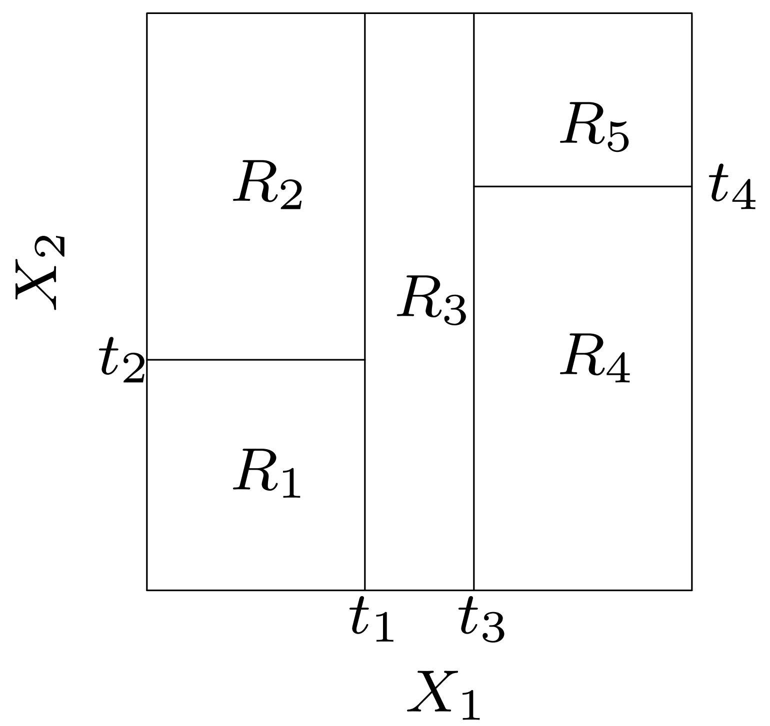


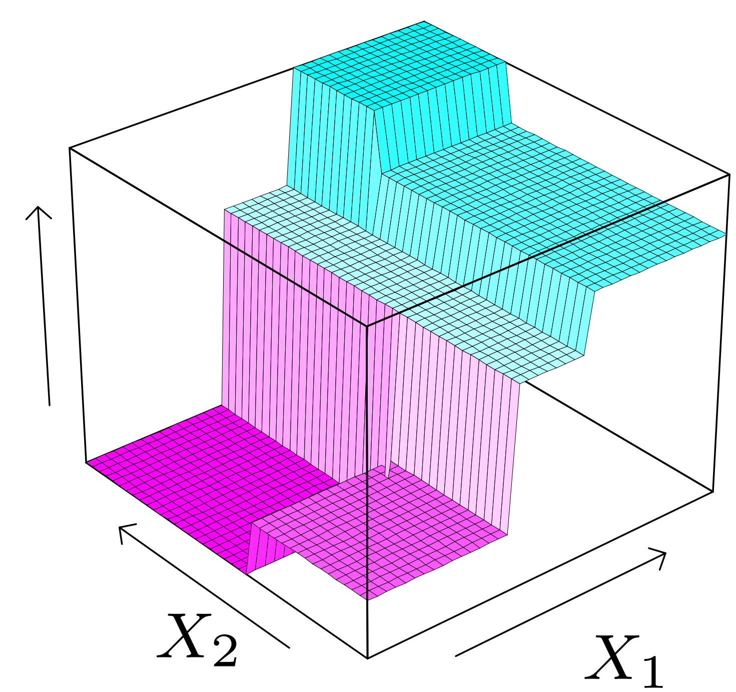


k=5

Decision Trees







Can be used for classification or regression

Can handle discrete or continuous features

Interpretable but tend not to work as well
as other methods.

(figure from Hastie, Tibshirani, and Friedman, 2009)