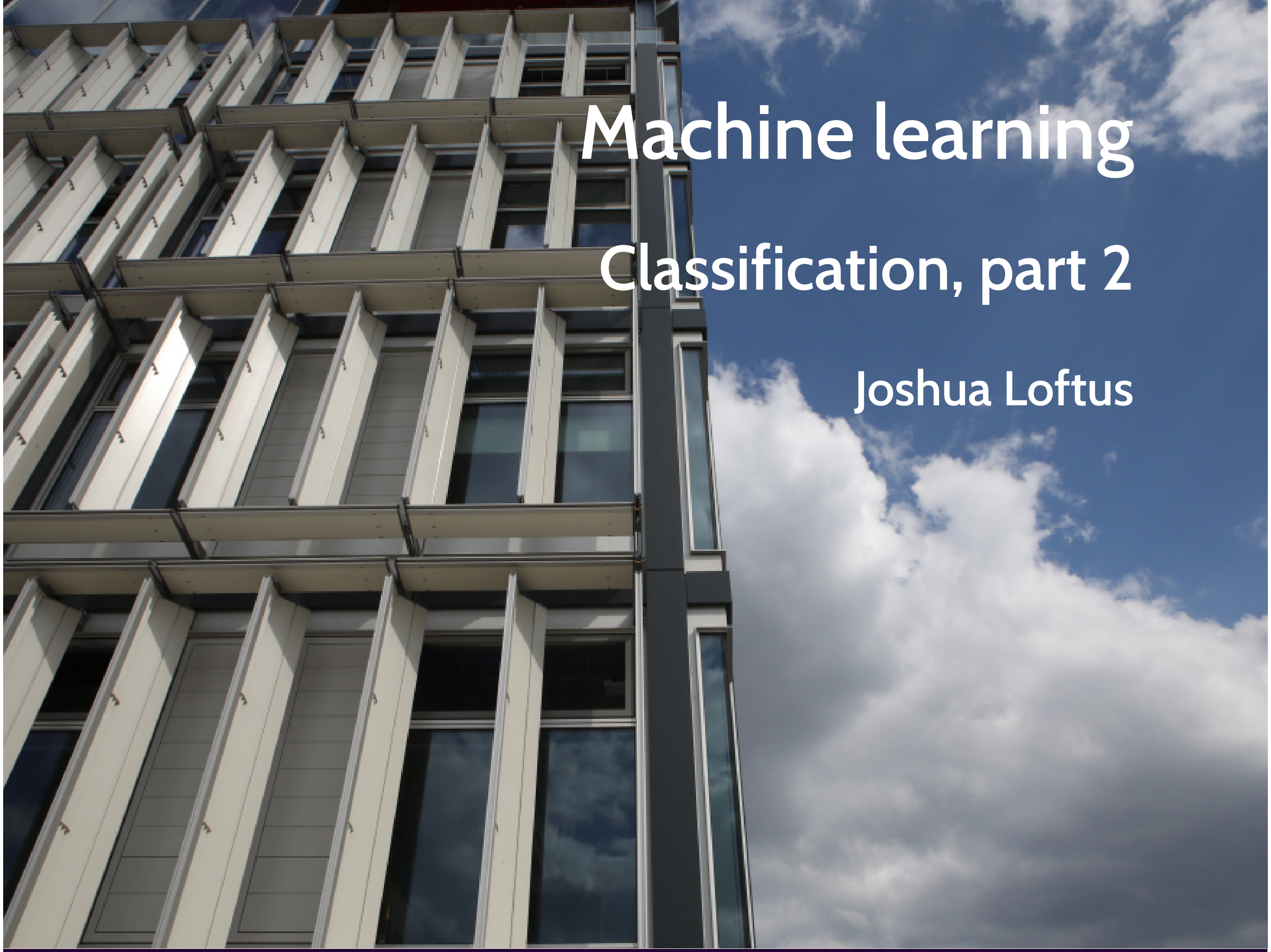


Machine learning

Classification, part 2

Joshua Loftus

Support Vector Machines

Classification with linear decision boundaries

We already saw this with logistic regression

$$\hat{y} = 1 \iff \hat{p} > g^{-1}(c) \iff \mathbf{x}^T\beta > c$$

But we also know that logistic regression fails, for example, if the classes are perfectly separable (zero classification error)

What can we do in that case?

Notation for linear classification

Define a **linear classifier** $f(\mathbf{x})$ by

$$f(\mathbf{x}) = \beta_0 + \mathbf{x}^T\beta$$

with classification boundary $f(\mathbf{x}) = 0$, and decision rule

$$G(\mathbf{x}) = \text{sign}[f(\mathbf{x})]$$

Notation change: It's convenient to assume $y \in \{\pm 1\}$ instead of 0-1

$$\text{Misclassification } \leftrightarrow y \cdot G(\mathbf{x}) < 0$$

How to choose β?

We just want a linear classification boundary

Forget modeling the class probabilities

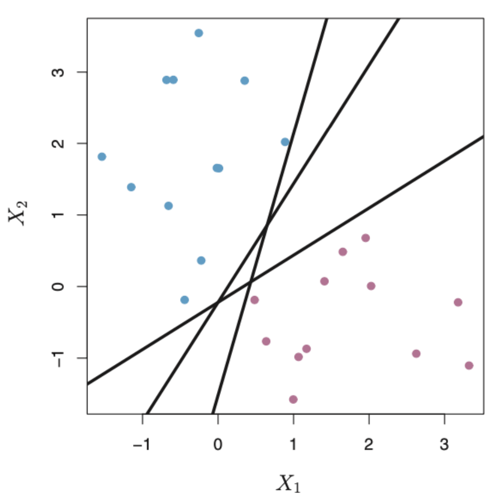
Consider the separable case... The classification task should be "easy" but we can't do it with logistic regression

First some geometric intuition

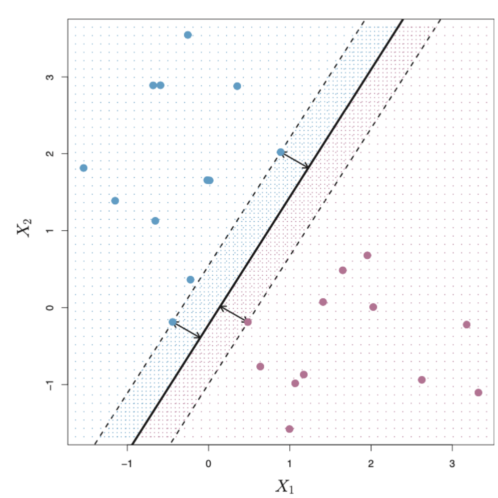
for the separable case

Then we'll figure out how to extend our new proposed solution to the non-separable case

Geometric intuition: maximize distance



Many linear classifiers with zero training error



Unique classifier with largest distance

Figures from Chapter 9 of ISLR

Maximizing the "margin" (separable case)

Exercise: Distance from $\mathbf{x}$ to the decision boundary $\{\mathbf{z} : f(\mathbf{z}) = 0\}$, defined as the minimum distance to any point on the boundary,

$$\min \|\mathbf{x} - \mathbf{z}\| \text{ s.t. } f(\mathbf{z}) = 0$$

is given by (hint: orthogonal projection)

$$\frac{|f(\mathbf{x})|}{\|\beta\|}$$

and the smallest such distance in the training data is

$$\min_{1 \leq i \leq n} \frac{|f(\mathbf{x}_i)|}{\|\beta\|}$$

Constrained maximisation (separable case)

We could make the margin infinitely large by just sending the decision boundary $\to \infty$ away from all the data... 🤪

Recall that we want to choose *from among those linear classifiers that have **zero** classification errors*

Solve the *constrained* optimization problem

So there are infinitely many β where on our training data

$$\text{maximize} \left[\min_{1 \leq i \leq n} \frac{|f(\mathbf{x}_i)|}{\|\beta\|} \right]$$

subject to (s.t.)

$$y_i f(\mathbf{x}_i) > 0 \text{ for } 1 \leq i \leq n$$

Hey

Think about this

$$\max_{\beta} \left[\min_{1 \leq i \leq n} \frac{|f(\mathbf{x}_i)|}{\|\beta\|} \right]$$

$$\text{s.t.} \quad y_i f(\mathbf{x}_i) > 0 \text{ for } 1 \leq i \leq n$$

Notice something?...

Recurring theme: model/optimization/fit depends most strongly (or in this case *only*) on point(s) closest to the boundary

Reformulating optimization (separable case)

Exercise: convince yourself this is equivalent to

$$\max_{M,\beta} M$$

$$\text{s.t. } y_i f(\mathbf{x}_i)/\|\beta\| \geq M \text{ for } 1 \leq i \leq n$$

(we have introduced a new variable, M, to optimize over)

Then, use re-scaling to show it's equivalent to

$$\text{minimize } \|\beta\|$$

$$\text{s.t. } y_i(\beta_0 + \mathbf{x}_i^T \beta) \geq 1 \text{ for } 1 \leq i \leq n$$

Since $\text{minimize } \|\beta\| \leftrightarrow \text{minimize } \|\beta\|^2$ this is a quadratic program with linear inequality constraints

ML = optimization

Can use standard **convex optimization** methods/software

This is nice because there's a whole field of mathematical research dedicated to problemsl like these

- Algorithms converging to *global* optimum
- Guaranteed convergence rates

To learn more check out LSE's MA333 which uses this book

Is this really necessary?

Community now focused on non-convex (deep learning) methods. "It just works (better)"

Non-separable case

Idea: allow a "budget" for constraint violations

If observation i is misclassified then let $\xi_i / ||\beta||$ be its distance from the boundary. Solve

$$\text{minimize } ||\beta||^2$$

$$\text{s.t. for } 1 \leq i \leq n,$$

$$y_i(\beta_0 + \mathbf{x}_i^T \beta) \geq 1 - \xi_i$$

$$\xi_i \geq 0, \sum \xi_i \leq C$$

Complexity: C is a tuning parameter (more about this in slide after next one)

"Support vectors"

(Warning: challenging, more advanced, not on the exam)

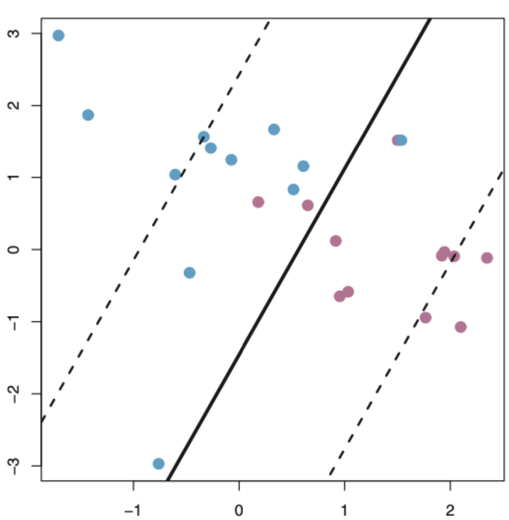
Exercise: use careful calculus to show the optimal $\hat{\beta}$ can be written as a linear combination of the feature vectors $\mathbf{x}_i$.

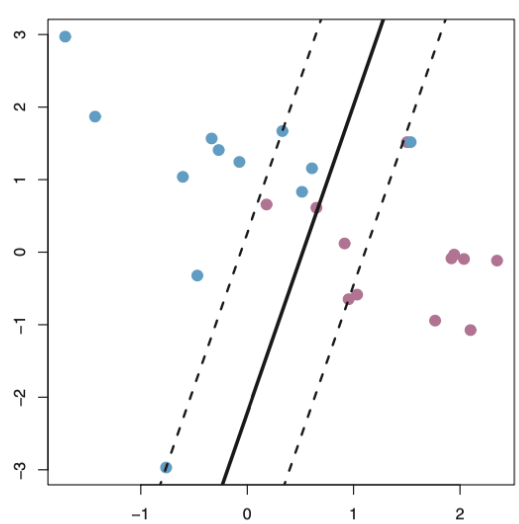
Exercise: also show that $\hat{\beta}$ can be written as a *sparse* linear combination of $\mathbf{x}_i$ (with nonzero coefficients only for those observations on or violating the constraint)

(Hint: see ESL 12.2.1)

Exact mathematical statement related to our *recurring theme* -- solution depends only on a few observations

Bias-variance trade off (ISLR 9.7)





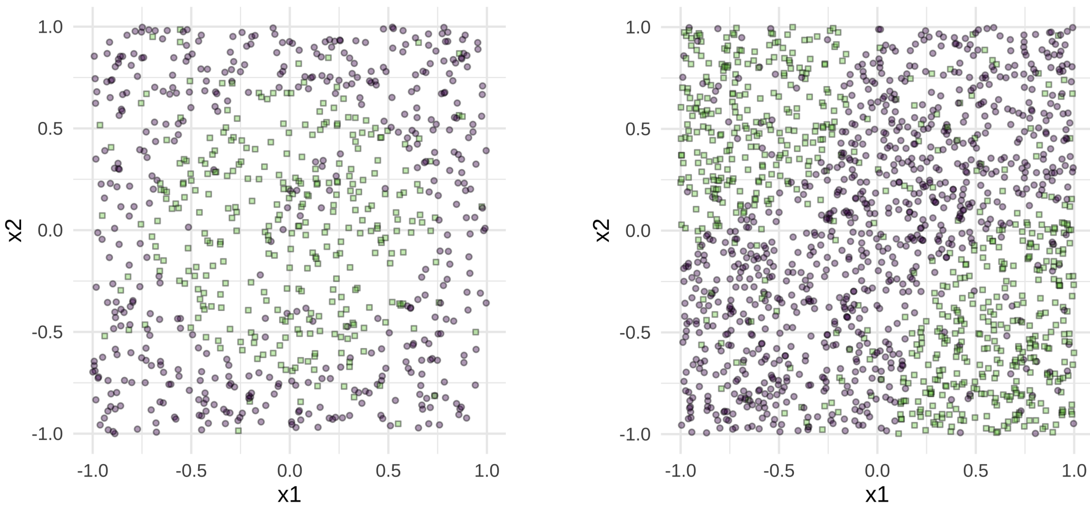
Budget $\uparrow$ # support vectors $\uparrow$

Bias $\uparrow$ Variance $\downarrow$

Budget $\downarrow$ # support vectors $\downarrow$

Bias $\downarrow$ Variance $\uparrow$

Non-linear classification boundaries



What if the data looks like this? Game over 😱 for linear classifiers? (Piece of cake after we learn about **kernel methods**)

"Generative" supervised learning

Some binary thinking...

- **Science vs humanities**. C. P. Snow, *The Two Cultures and the Scientific Revolution* (1959)
- **Intuition vs logic** or **fast vs slow** (Kahneman, 2011), **fox vs hedgehog** (Berlin, 1953 or Gould, 2003)
- **Algorithm vs inference**, *Statistical Modeling: The Two Cultures* (Breiman, 2001) or *Computer Age Statistical Inference* (Efron and Hastie, 2016)
- **Probabilistic** (generative, random, stochastic) vs **physical** (geometric, deterministic)

(Of course, none of these binaries are "real")

Comparison

Probability axioms = **constraints**

Without probability

- Prediction accuracy
- Algorithm efficiency

With probability

- Inference: prediction/confidence intervals, hypothesis tests
- Interpretation: coefficients might be meaningful
- Model diagnostics

History according to Efron and Hastie: algorithm first (possibly unconstrained), then inference gradually catches up

Coming soon: non-linearity

Summary of recent development

Concretely

- More details on logistic regression
- Support Vector Machines

Abstractly

- Optimization algorithms / fitting procedures depending more strongly on observations that are more difficult to classify
- Same optimization problem can be written many different ways, can be more or less amenable to theory/algorithms