

Lecture 4b. Logistic Regression.


COMP90051 Statistical Machine Learning

Semester 2, 2020
Lecturer: Ben Rubinstein



THE UNIVERSITY OF
MELBOURNE

This lecture

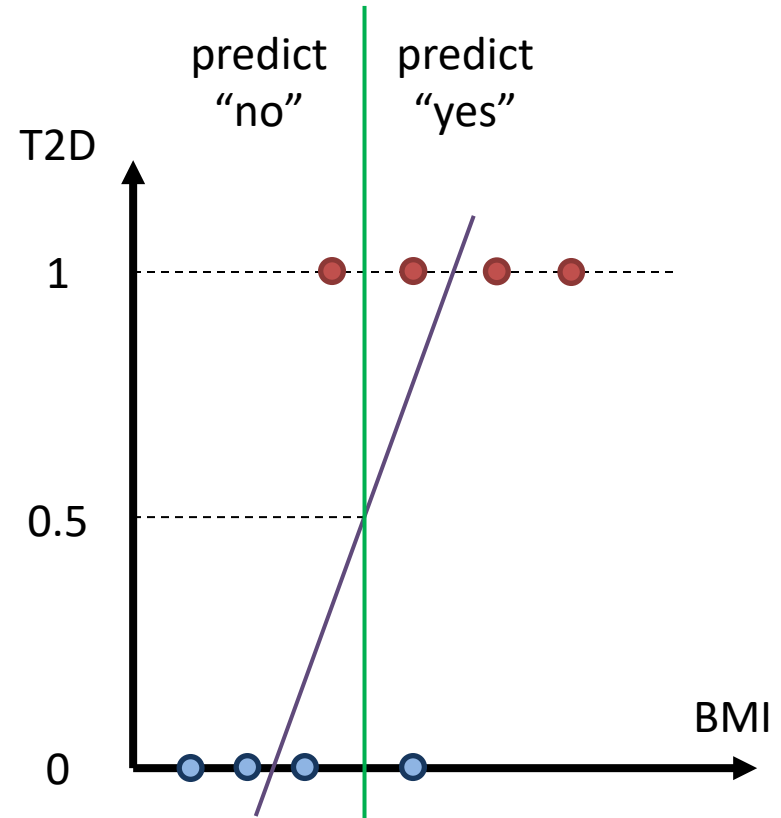
- Iterative optimisation for extremum estimators
 - * First-order method: Gradient descent
 - * Second-order: Newton-Raphson method
 -  Later: Lagrangian duality
- **Logistic regression**: workhorse linear classifier
 - * Possibly familiar derivation: **frequentist**
 - * **Decision-theoretic** derivation
 - * Training with Newton-Raphson looks like repeated, weighted linear regression

Logistic Regression Model

A workhorse linear, binary classifier;
(A review for some of you; new to some.)

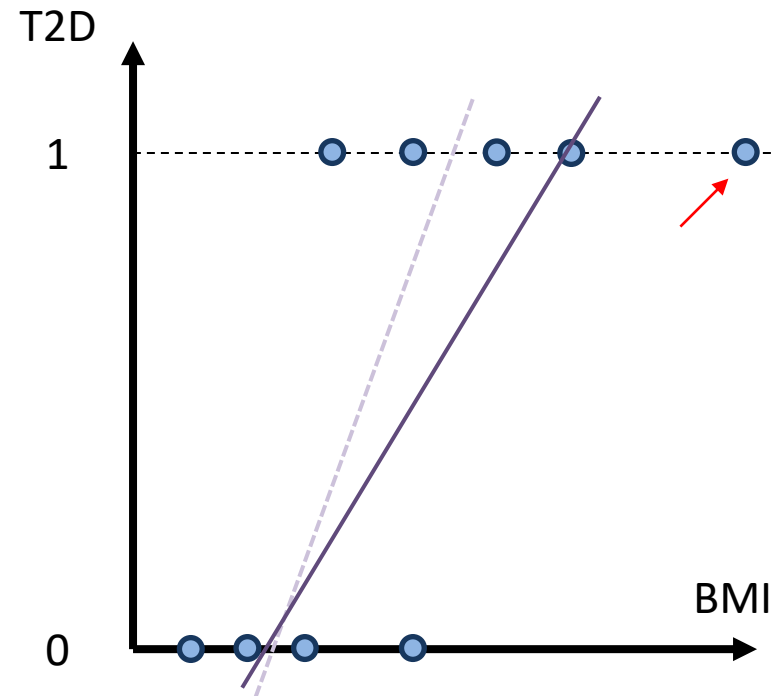
Binary classification: Example

- Example: given body mass index (BMI) does a patient have type 2 diabetes (T2D)?
- This is (supervised) **binary classification**
- One *could* use linear regression
 - * Fit a line/hyperplane to data (find weights \mathbf{w})
 - * Denote $s \equiv \mathbf{x}'\mathbf{w}$
 - * Predict "Yes" if $s \geq 0.5$
 - * Predict "No" if $s < 0.5$



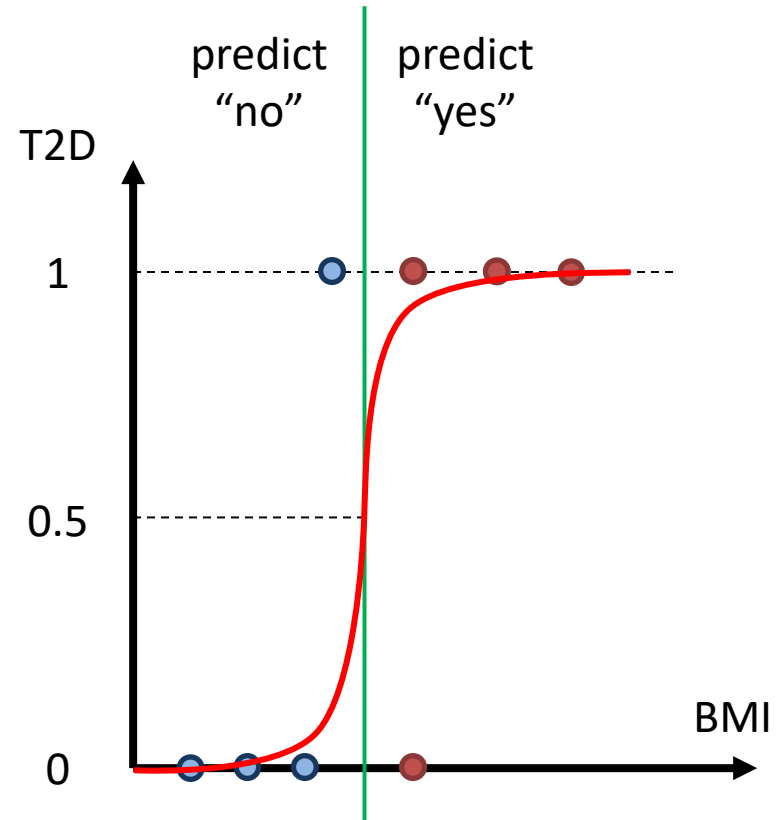
Why not linear regression

- Due to the square loss, points far from boundary have loss squared – even if they’re confidently correct!
- Such “outliers” will “pull at” the linear regression
- Overall, the least-squares criterion looks unnatural in this setting



Logistic regression model

- Probabilistic approach to classification
 - * $P(Y = 1|\mathbf{x}) = f(\mathbf{x}) = ?$
 - * Use a linear function? E.g., $s(\mathbf{x}) = \mathbf{x}'\mathbf{w}$
- Problem: the probability needs to be between 0 and 1.
- **Logistic** function $f(s) = \frac{1}{1+\exp(-s)}$
- **Logistic regression model**
$$P(Y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{x}'\mathbf{w})}$$



How is logistic regression *linear*?

- Logistic regression model:

$$P(Y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{x}'\mathbf{w})}$$

- Classification rule:

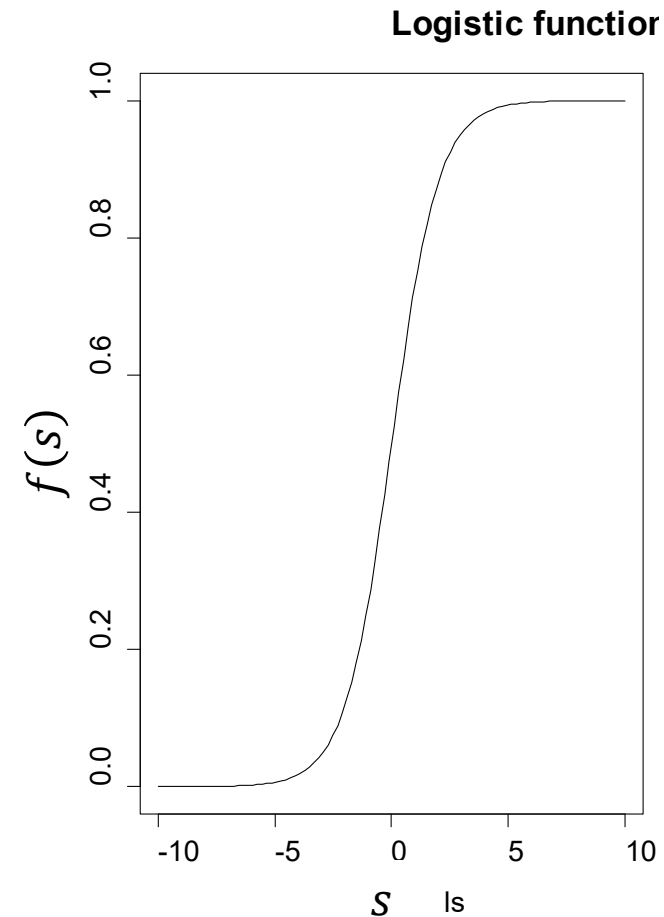
if $(P(Y = 1|\mathbf{x}) > \frac{1}{2})$ then class “1”, else class “0”

- Decision boundary is the set of \mathbf{x} 's such that:

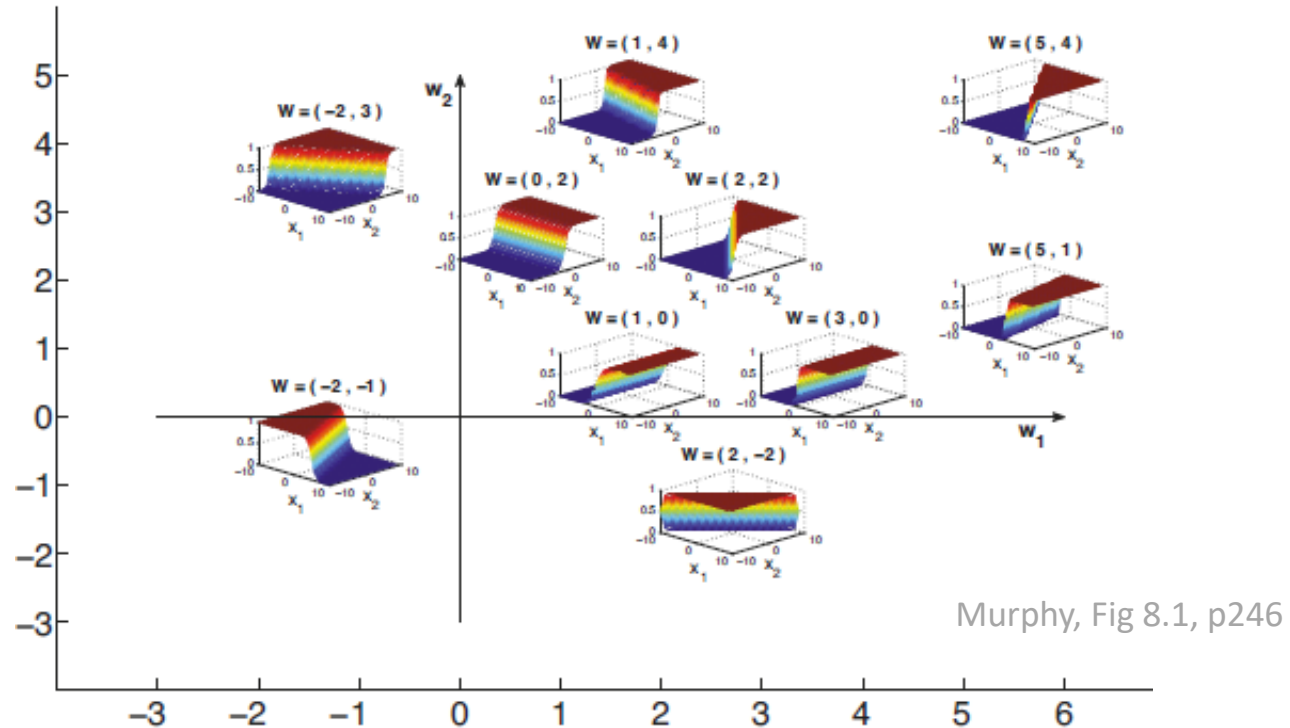
$$\frac{1}{1 + \exp(-\mathbf{x}'\mathbf{w})} = \frac{1}{2}$$

$$\exp(-\mathbf{x}'\mathbf{w}) = 1$$

$$\mathbf{x}'\mathbf{w} = 0$$



Effect of parameter vector (2D problem)



- **Decision boundary** is the line where $P(Y = 1|x) = 0.5$
 - * In higher dimensional problems, the decision boundary is a plane or hyperplane
- Vector \mathbf{w} is perpendicular to the decision boundary (see linear algebra review topic)
 - * That is, \mathbf{w} is a normal to the decision boundary
 - * Note: in this illustration we assume $w_0 = 0$ for simplicity

Linear vs. logistic probabilistic models

- **Linear regression** assumes a Normal distribution with a fixed variance and mean given by linear model

$$p(y|\mathbf{x}) = \text{Normal}(\mathbf{x}'\mathbf{w}, \sigma^2)$$

- **Logistic regression** assumes a Bernoulli distribution with parameter given by logistic transform of linear model

$$p(y|\mathbf{x}) = \text{Bernoulli}(\text{logistic}(\mathbf{x}'\mathbf{w}))$$

- Recall that **Bernoulli distribution** is defined as

$$p(1) = \theta \text{ and } p(0) = 1 - \theta \text{ for } \theta \in [0,1]$$

- Equivalently $p(y) = \theta^y (1 - \theta)^{(1-y)}$ for $y \in \{0,1\}$

Training as Max-Likelihood Estimation

- Assuming independence, probability of data

$$p(y_1, \dots, y_n | \mathbf{x}_1, \dots, \mathbf{x}_n) = \prod_{i=1}^n p(y_i | \mathbf{x}_i)$$

- Assuming Bernoulli distribution we have

$$p(y_i | \mathbf{x}_i) = (\theta(\mathbf{x}_i))^{y_i} (1 - \theta(\mathbf{x}_i))^{1-y_i}$$

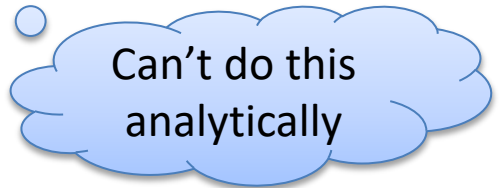
$$\text{where } \theta(\mathbf{x}_i) = \frac{1}{1 + \exp(-\mathbf{x}_i' \mathbf{w})}$$

- Training: maximise this expression wrt weights \mathbf{w}

Apply log trick, simplify

- Instead of maximising likelihood, maximise its logarithm

$$\begin{aligned}\log \left(\prod_{i=1}^n p(y_i | \mathbf{x}_i) \right) &= \sum_{i=1}^n \log p(y_i | \mathbf{x}_i) \\&= \sum_{i=1}^n \log \left((\theta(\mathbf{x}_i))^{y_i} (1 - \theta(\mathbf{x}_i))^{1-y_i} \right) \\&= \sum_{i=1}^n (y_i \log(\theta(\mathbf{x}_i)) + (1 - y_i) \log(1 - \theta(\mathbf{x}_i))) \\&= \sum_{i=1}^n ((y_i - 1)\mathbf{x}_i' \mathbf{w} - \log(1 + \exp(-\mathbf{x}_i' \mathbf{w})))\end{aligned}$$



Can't do this
analytically

Logistic Regression: Decision-Theoretic View

Via cross-entropy loss

Background: Cross entropy

- Cross entropy is an information-theoretic method for comparing two distributions
- Cross entropy is a measure of a **divergence** between reference distribution $g_{ref}(a)$ and estimated distribution $g_{est}(a)$. For discrete distributions:

$$H(g_{ref}, g_{est}) = - \sum_{a \in A} g_{ref}(a) \log g_{est}(a)$$

A is support of the distributions, e.g., $A = \{0,1\}$

Training as cross-entropy minimisation

- Consider log-likelihood for a single data point
$$\log p(y_i | \mathbf{x}_i) = y_i \log(\theta(\mathbf{x}_i)) + (1 - y_i) \log(1 - \theta(\mathbf{x}_i))$$
- Cross entropy $H(g_{ref}, g_{est}) = -\sum_a g_{ref}(a) \log g_{est}(a)$
 - * If reference (true) distribution is

$$g_{ref}(1) = y_i \text{ and } g_{ref}(0) = 1 - y_i$$

- * With logistic regression estimating this distribution as

$$g_{est}(1) = \theta(\mathbf{x}_i) \text{ and } g_{est}(0) = 1 - \theta(\mathbf{x}_i)$$

It finds \mathbf{w} that minimises sum of cross entropies per training point

Summary

- Logistic regression *formulation*
 - * A workhorse linear binary classifier
 - * Frequentist: Bernoulli label with coin bias logistic-linear in \mathbf{x}
 - * Decision theory: Maximising cross entropy with labels

Next time: Training quickly with Newton-Raphson, and how that is repeated (weighted) linear regression under the hood!