Introduction to machine learning

Gianluca Campanella

Contents

Definitions

Differences

Statistics

- Predates computers

$\rightarrow$ **Understand why something happens** in the face of uncertainty

Differences

Statistics

- Predates computers

→ **Understand why something happens** in the face of uncertainty

Machine Learning

- 'Algorithmic modelling' (L. Breiman)

→ Computers can **learn rules** without explicit programming

Two types of Data Science

Analysis-focused

- Maths and Statistics
- Business Intelligence

→ Assist human decision-making

Building-focused

- Machine Learning
- Software Engineering

→ Develop and deploy data-driven products

The five questions

1. How much/many?
2. Is this A or B?
3. How is this organised?
4. Is this weird?
5. What should I do next?

How much/many?

Examples

- How many people will develop cancer in the next 10 years?
- How long will this patient stay in hospital?

$\downarrow$

Regression algorithms

Is this A or B?

Examples

- How likely is this patient to be readmitted in the next year?
- What's the 10-year CVD risk of this patient?

$\downarrow$

Classification algorithms

How is this organised?

Examples

- Which patients develop similar diseases?
- Which diseases frequently occur together?

$\downarrow$

Clustering algorithms

Is this weird?

Examples

- Is the number of cases higher than expected?
- Is this biomarker measurement abnormal?

$\downarrow$

Anomaly detection algorithms

What should I do next?

Examples

- How should warfarin be dosed in this patient?
- How much insulin is needed to stabilise blood glucose?

$\downarrow$

Reinforcement learning algorithms

Supervised vs unsupervised algorithms

Supervised algorithms

- Are trained on existing data
- Can be compared according to some ‘goodness’ metric

Unsupervised algorithms

- Don’t use examples with known outcomes
- Give clues, not ‘right answers’

The five questions… revisited

Family	Class	Question
Supervised	Regression	How much/many?
	Classification	Is this A or B?
Unsupervised	Clustering	How is this organised?
	Anomaly detection	Is this weird?
	Reinforcement learning	What should I do next?

Prediction

Guessing values

- Y = 'length of hospital stay'
- You have some realisations $y_1, y_2, \ldots$ collected over time
- You want to predict the value of Y for a new patient

Guessing values

- Y = 'length of hospital stay'
- You have some realisations $y_1, y_2, \dots$ collected over time
- You want to predict the value of Y for a new patient

How do you do this?

If you prefer, what's the **optimal point forecast** for Y?

Loss functions

Before you can answer, you need a **loss function** that…

- Measures how big an error you're making with your guess g
- Can be minimised to obtain the 'best' g

Loss functions

Before you can answer, you need a **loss function** that...

- Measures how big an error you're making with your guess g
- Can be minimised to obtain the 'best' g

Mean squared error	$\mathrm{MSE}(g) = \mathbb{E}\left[(Y - g)^2\right]$
Mean absolute error	$\mathrm{MAE}(g) = \mathbb{E}[\lvert Y - g \rvert]$

Regression versus classification

Regression

Aim Predict a **continuous** value

Loss How ‘off’ (numerically) our predictions are

Classification

Aim Predict a **class**

Loss How ‘inaccurate’ the predicted classes are

Towards prediction…

Usually we have at least another variable X that we believe to be related to Y…

Towards prediction…

Usually we have at least another variable X that we believe to be related to Y…

Idea

Using some function f of X, we should be able to predict Y '**better**' (i.e. reduce the mean error) than by ignoring it

$$g \rightsquigarrow f(X) \quad \text{and thus} \quad \mathrm{MSE}(f) = \mathbb{E}\left[(Y - f(X))^2\right]$$

What should f be?

Consider the decomposition

$$Y \mid X = f^{\star}(X) + \epsilon$$

- $f^{\star}$ is the optimal prediction (conditional on knowing X)
- ϵ is a random variable (since $f^{\star}$ is not)
- $\mathbb{E}[\epsilon] = 0$ without loss of generality

What should f be?

For the MSE, it can be shown that

$$f^\star(x) = \mathbb{E}[Y \mid X = x]$$

$f^\star$ is what we'd like to know when we want to predict Y given X

...but can we?

Bias-variance trade-off

Bias-variance trade-off

Suppose that…

- The 'true' regression function is $f^\star$
- We have to make do with some suboptimal f

Let's start by expanding the MSE…

$$
\begin{aligned}
(Y - f)^2 &= (Y - f^\star + f^\star - f)^2 \\
&= [(Y - f^\star) + (f^\star - f)]^2 \\
&= (Y - f^\star)^2 + 2(Y - f^\star)(f^\star - f) + (f^\star - f)^2
\end{aligned}
$$

Bias-variance trade-off

Now take the expectation...

$$\mathbb{E}\left[(Y - f^{\star})^2 + 2(Y - f^{\star})(f^{\star} - f) + (f^{\star} - f)^2\right]$$

Since $Y - f^{\star} = \epsilon$ and $\mathbb{E}[\epsilon] = 0$, we have...

$$\begin{aligned}
\mathbb{E}\left[(Y - f^{\star})^2\right] &= \mathbb{V}[\epsilon] \\
\mathbb{E}[Y - f^{\star}] &= \mathbb{E}[\epsilon] = 0 \\
\mathbb{E}\left[(f^{\star} - f)^2\right] &= (f^{\star} - f)^2
\end{aligned}$$

Bias-variance trade-off

$$\mathbb{E}[\mathrm{MSE}(f)] = \mathbb{V}[\epsilon] + (f^\star - f)^2$$

Variance $\mathbb{V}[\epsilon]$

- Doesn't depend on f, just on 'how hard' it is to predict $Y|X = x$

$\rightarrow$ Unpredictable, irreducible fluctuation

Bias-variance trade-off

$$\mathbb{E}[\mathrm{MSE}(f)] = \mathbb{V}[\epsilon] + (f^\star - f)^2$$

Bias $(f^\star - f)^2$

- 'Extra error' we get from not knowing $f^\star$

$\rightarrow$ Amount by which we are systematically off

Bias-variance trade-off

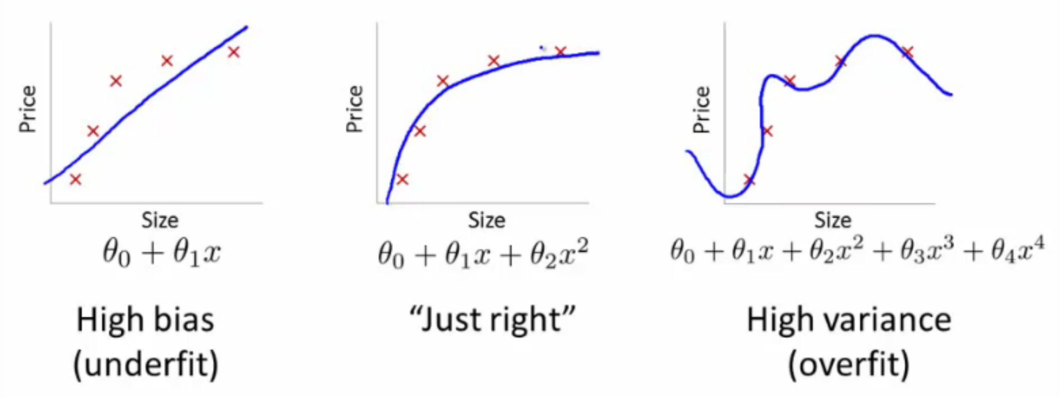
Since f is itself estimated from a sample (it's actually $\hat{f}$), we have...

- The **irreducible variance** $\mathbb{V}[\epsilon]$
- The **bias** in approximating $f^\star$ using f
- The additional **(estimation) variance** of $\hat{f}$

Consistent methods

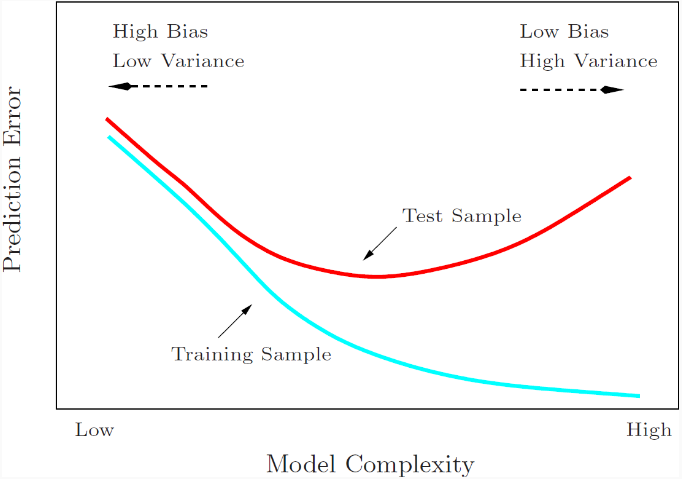
- Bias and estimation variance $\to 0$ as the sample size increases
- Different consistent methods may converge at different rates

Bias-variance trade-off



From Andrew Ng's *Machine Learning* course

Bias-variance trade-off and generalisability



From *The Elements of Statistical Learning*

Cross-validation

General idea

- Fit several models on subsets of the data
- Measure performance of each
- Compute the mean performance

k-fold cross-validation

- Split the data into *k* groups (a.k.a. ‘folds’)
- Repeat for each fold:
    - Fit the model using all but the selected fold
    - Measure performance on the selected fold
- Compute the mean performance across folds

Regularisation

- Penalise ‘large’ coefficients by shrinking them
- Helps avoid overfitting
- Requires **tuning** of an additional parameter α representing the ‘weight’ of the penalty (relative to the prediction error)

L_1	LASSO	$\sum_j \lvert\beta_j\rvert$
L_2	Tikhonov or ridge	$\sum_j \beta_j^2$