Introduction to Machine Learning and Data Mining

COMP9417 Machine Learning and Data Mining

Last revision: 28 Feb 2018

Acknowledgements

Material derived from slides for the book
"Elements of Statistical Learning (2nd Ed.)" by T. Hastie,
R. Tibshirani & J. Friedman. Springer (2009)
http://statweb.stanford.edu/~tibs/ElemStatLearn/

Material derived from slides for the book
"Machine Learning: A Probabilistic Perspective" by P. Murphy
MIT Press (2012)
http://www.cs.ubc.ca/~murphyk/MLbook

Material derived from slides for the book
"Machine Learning" by P. Flach
Cambridge University Press (2012)
http://cs.bris.ac.uk/~flach/mlbook

Material derived from slides for the book
"Bayesian Reasoning and Machine Learning" by D. Barber
Cambridge University Press (2012)
http://www.cs.ucl.ac.uk/staff/d.barber/brml

Material derived from slides for the book
"Machine Learning" by T. Mitchell
McGraw-Hill (1997)
http://www-2.cs.cmu.edu/~tom/mlbook.html

Material derived from slides for the course
"Machine Learning" by A. Srinivasan
BITS Pilani, Goa, India (2016)

Aims

This lecture will introduce you to machine learning, giving an overview of some of the key ideas and approaches we will cover in the course.

Following it you should be able to describe some of the main concepts and outline some of the main techniques that are used in machine learning:

- categories of learning (supervised learning, unsupervised learning, etc.)
- widely-used techniques of machine learning algorithms
- batch vs. online settings
- parametric vs. non-parametric approaches
- generalisation in machine learning
- training, validation and testing phases in applications
- limits on learning

What we will cover

- core algorithms and model types in machine learning
- foundational concepts regarding learning from data
- relevant theory to inform and generalise understanding
- practical applications

What we will NOT cover

- lots of probability and statistics
- lots of neural nets and deep learning
- "big data"
- commercial and business aspects of "analytics"
- ethical aspects of AI and ML

although all of these are interesting and important topics

Some history

> *One can imagine that after the machine had been in operation for some time, the instructions would have been altered out of recognition, but nevertheless still be such that one would have to admit that the machine was still doing very worthwhile calculations. Possibly it might still be getting results of the type desired when the machine was first set up, but in a much more efficient manner. In such a case one would have to admit that the progress of the machine had not been foreseen when its original instructions were put in. It would be like a pupil who had learnt much from his master, but had added much more by his own work.*

From A. M. Turing's lecture to the London Mathematical Society. (1947)

Some definitions

> *The field of machine learning is concerned with the question of how to construct computer programs that automatically improve from experience.*

"Machine Learning". T. Mitchell (1997)

> *Machine learning, then, is about making computers modify or adapt their actions (whether these actions are making predictions, or controlling a robot) so that these actions get more accurate, where accuracy is measured by how well the chosen actions reflect the correct ones.*

"Machine Learning". S. Marsland (2015)

Some definitions

Machine learning is the systematic study of algorithms and systems that improve their knowledge or performance with experience.

Machine Learning”. P. Flach (2012)

The term machine learning refers to the automated detection of meaningful patterns in data.

“Understanding Machine Learning”. S. Shalev-Shwartz and S. Ben-David (2014)

Data mining is the extraction of implicit, previously unknown, and potentially useful information from data.

“Data Mining”. I. Witten *et al.* (2016)

Machine Learning is . . .

Trying to get programs to work in a reasonable way to predict stuff.

R. Kohn (2015)

How is Machine Learning different from ...

Machine learning comes originally from Artificial Intelligence (AI), where the motivation is to build intelligent agents, capable of acting autonomously. Learning is a characteristic of intelligence, so to be successful an agent must ultimately be able to learn, *apply*, *understand* and *communicate* what it has learned.

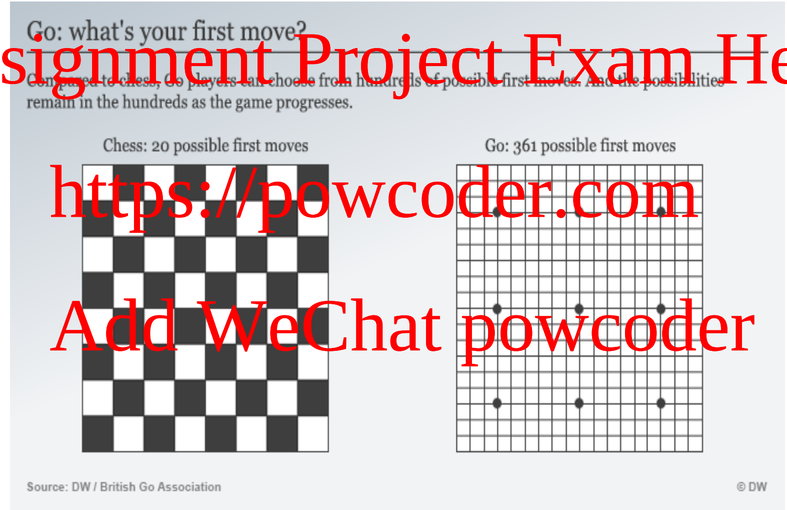
These are not requirements in:

- statistics — the results are typically mathematical models for humans
- data mining — the results are typically models of "insight" for humans

These criteria are often also necessary, but not always sufficient, for machine learning.

Machine Learning for Human-level Artifical Intelligence

ML for human-level AI, right ?



ML for human-level AI, right ? Not so fast . . .

Supervised and unsupervised learning

The most widely used categories of machine learning algorithms are:

- *Supervised learning* – output class (or label) is given
- *Unsupervised learning* – no output class is given

There are also hybrids, such as semi-supervised learning, and alternative strategies to acquire data, such as reinforcement learning and active learning.

Note: output class can be real-valued or discrete, scalar, vector, or other structure . . .

Supervised and unsupervised learning

Supervised learning tends to dominate in applications.

Why ?

Generally, because it is much easier to define the problem and develop an error measure (loss function) to evaluate different algorithms, parameter settings, data transformations, etc. for supervised learning than for unsupervised learning.

Supervised and unsupervised learning

Unfortunately ...

In the real world it is often difficult to obtain good labelled data in sufficient quantities

So in such cases unsupervised learning is really what you want ...

but currently, finding good unsupervised learning algorithms for complex machine learning tasks remains a research challenge.

Machine learning models

Machine learning models can be distinguished according to their main intuition, for example:

- *Geometric* models use intuitions from geometry such as separating (hyper-)planes, linear transformations and distance metrics.
- *Probabilistic* models view learning as a process of reducing uncertainty, modelled by means of probability distributions.
- *Logical* models are defined in terms of easily interpretable logical expressions.

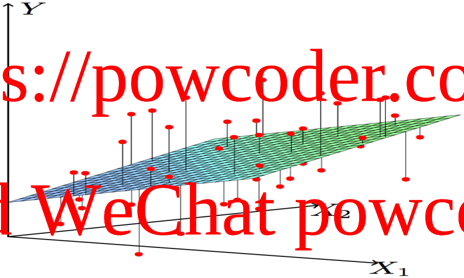
Machine learning models

Alternatively, can be characterised by *algorithmic properties*:

- *Regression models* predict a numeric output
- *Classification models* predict a discrete class value
- *Neural networks* learn based on a biological analogy
- *Local models* predict in the local region of a query instance
- *Tree-based models* partition the data to make predictions
- *Ensembles* learn multiple models and combine their predictions

Linear regression

Given 2 real-valued variables X_1, X_2, labelled with a real-valued variable Y, find "line of best fit" that captures the dependency of Y on X_1, X_2.



Learning here is by minimizing MSE, i.e., the average of the squared vertical distances of values of Y from the learned function $\hat{Y} = \hat{f}(\mathbf{X})$.

Linear regression

A question: is it possible to do better than the line of best fit?

Maybe. Linear regression assume that the $(\mathbf{x}_i, y_i)$ examples in the data are "generated" by the true (but unknown) function $Y = f(\mathbf{X})$.

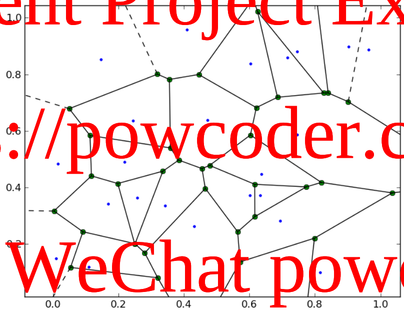
So any training set is a sample from the true distribution $E(Y) = f(\mathbf{X})$.

But what if f is non-linear ?

We may be able to reduce the mean squared error (MSE) value $\sum_i (y_i - \hat{y})^2$ by trying a different function.

Can "decompose" MSE to aid in selecting a better function.

Nearest Neighbour



Nearest Neighbour is a regression or classification algorithm that predicts whatever is the output value of the nearest data point to some query.

Classification

Customer103: (time=t0)	*Customer103:* (time=t1)	...	*Customer103:* (time=tn)
Years of credit: 9	Years of credit: 9		Years of credit: 9
Loan balance: $2,400	Loan balance: $3,250		Loan balance: $4,500
Income: $52k	Income: ?		Income: ?
Own House: Yes	Own House: Yes		Own House: Yes
Other delinquent accts: 2	Other delinquent accts: 2		Other delinquent accts: 3
Max billing cycles late: 3	Max billing cycles late: 4		Max billing cycles late: 6
Profitable customer?: ?	Profitable customer?: ?		**Profitable customer?: No**
...	...		...

If $OtherDelinquentAccounts > 2$, and
$NumberDelinquentBillingCycles > 1$
Then $ProfitableCustomer = No$

If $OtherDelinquentAccounts = 0$, and
$Income > 30k$ OR $YearsOfCredit > 3$
Then $ProfitableCustomer = Yes$

Assassinating spam e-mail

SpamAssassin is a widely used open-source spam filter. It calculates a score for an incoming e-mail, based on a number of built-in rules or 'tests' in SpamAssassin's terminology, and adds a 'junk' flag and a summary report to the e-mail's headers if the score is 5 or more.


```
-0.1 RCVD_IN_MXRATE_WL        RBL: MXRate recommends allowing
                              [123.45.6.789 listed in sub.mxrate.net]
 0.6 HTML_IMAGE_RATIO_02      BODY: HTML has a low ratio of text to image area
 1.2 TVD_FW_GRAPHIC_NAME_MID  BODY: TVD_FW_GRAPHIC_NAME_MID
 0.0 HTML_MESSAGE             BODY: HTML included in message
 0.6 HTML_FONx_FACE_BAD       BODY: HTML font face is not a word
 1.4 SARE_GIF_ATTACH          FULL: Email has a inline gif
 0.1 BOUNCE_MESSAGE           MTA bounce message
 0.1 ANY_BOUNCE_MESSAGE       Message is some kind of bounce message
 1.4 AWL                      AWL: From: address is in the auto white-list
```


From left to right you see the score attached to a particular test, the test identifier, and a short description including a reference to the relevant part of the e-mail. As you see, scores for individual tests can be negative (indicating evidence suggesting the e-mail is ham rather than spam) as well as positive. The overall score of 5.3 suggests the e-mail might be spam.

Linear classification

Suppose we have only two tests and four training e-mails, one of which is spam. Both tests succeed for the spam e-mail; for one ham e-mail neither test succeeds, for another the first test succeeds and the second doesn't, and for the third ham e-mail the first test fails and the second succeeds.

It is easy to see that assigning both tests a weight of 4 correctly 'classifies' these four e-mails into spam and ham. In the mathematical notation introduced above we could describe this classifier as $4x_1 + 4x_2 > 5$ or $(4, 4) \cdot (x_1, x_2) > 5$.

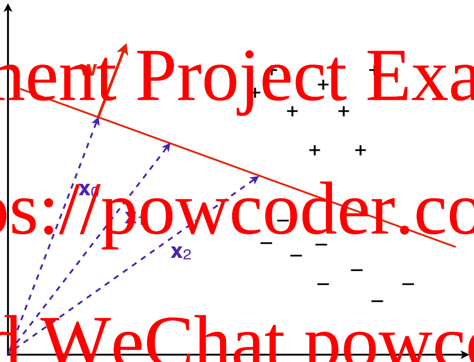
In fact, any weight between 2.5 and 5 will ensure that the threshold of 5 is only exceeded when both tests succeed. We could even consider assigning different weights to the tests – as long as each weight is less than 5 and their sum exceeds 5 – although it is hard to see how this could be justified by the training data.

Spam filtering as a classification task

The columns marked x_1 and x_2 indicate the results of two tests on four different e-mails. The fourth column indicates which of the e-mails are spam. The right-most column demonstrates that by thresholding the function $4x_1 + 4x_2$ at 5, we can separate spam from ham.

E-mail	x_1	x_2	Spam?	$4x_1 + 4x_2$
1	1	1	1	8
2	0	0	0	0
3	1	0	0	4
4	0	1	0	4

Linear classification in two dimensions



- straight line separates positives from negatives
- defined by $\mathbf{w} \cdot \mathbf{x}_i = t$
- $\mathbf{w}$ is perpendicular to decision boundary
- $\mathbf{w}$ points in direction of positives
- t is the decision threshold

Linear classification in two dimensions

Note: $\mathbf{x}_i$ points to a point on the decision boundary. In particular, $\mathbf{x}_0$ points in the same direction as $\mathbf{w}$, from which it follows that $\mathbf{w} \cdot \mathbf{x}_0 = ||\mathbf{w}|| \, ||\mathbf{x}_0|| = t$ (where $||\mathbf{x}||$ denotes the length of the vector $\mathbf{x}$).

Homogeneous coordinates

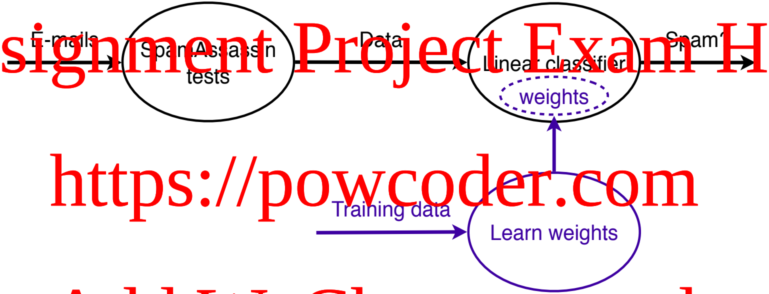
It is sometimes convenient to simplify notation further by introducing an extra constant 'variable' $x_0 = 1$, the weight of which is fixed to $w_0 = -t$.

The extended data point is then $\mathbf{x}^\circ = (1, x_1, \ldots, x_n)$ and the extended weight vector is $\mathbf{w}^\circ = (-t, w_1, \ldots, w_n)$, leading to the decision rule $\mathbf{w}^\circ \cdot \mathbf{x}^\circ > 0$ and the decision boundary $\mathbf{w}^\circ \cdot \mathbf{x}^\circ = 0$.

Thanks to these so-called *homogeneous coordinates* the decision boundary passes through the origin of the extended coordinate system, at the expense of needing an additional dimension.

Note: this doesn't really affect the data, as all data points and the 'real' decision boundary live in the plane $x_0 = 1$.

Machine learning for spam filtering



At the top we see how SpamAssassin approaches the spam e-mail classification task: the text of each e-mail is converted into a data point by means of SpamAssassin's built-in tests, and a *linear classifier* is applied to obtain a 'spam or ham' decision. At the bottom (in blue) we see the bit that is done by machine learning.

A Bayesian classifier I

Bayesian spam filters maintain a *vocabulary* of words and phrases – potential spam or ham indicators – for which statistics are collected from a *training set*.

- For instance, suppose that the word 'Viagra' occurred in four spam e-mails and in one ham e-mail. If we then encounter a new e-mail that contains the word 'Viagra', we might reason that the odds that this e-mail is spam are 4:1, or the probability of it being spam is 0.80 and the probability of it being ham is 0.20.

- The situation is slightly more subtle because we have to take into account the prevalence of spam. Suppose that I receive on average one spam e-mail for every six ham e-mails. This means that I would estimate the odds of an unseen e-mail being spam as 1:6, i.e., non-negligible but not very high either.

A Bayesian classifier II

- If I then learn that the e-mail contains the word 'Viagra', which occurs four times as often in spam as in ham, I need to combine these two odds. As we shall see later, Bayes' rule tells us that we should simply multiply them: 1:6 times 4:1 is 4:6, corresponding to a spam probability of 0.4.

In this way you are combining two independent pieces of evidence, one concerning the prevalence of spam, and the other concerning the occurrence of the word 'Viagra', pulling in opposite directions.

A Bayesian classifier III

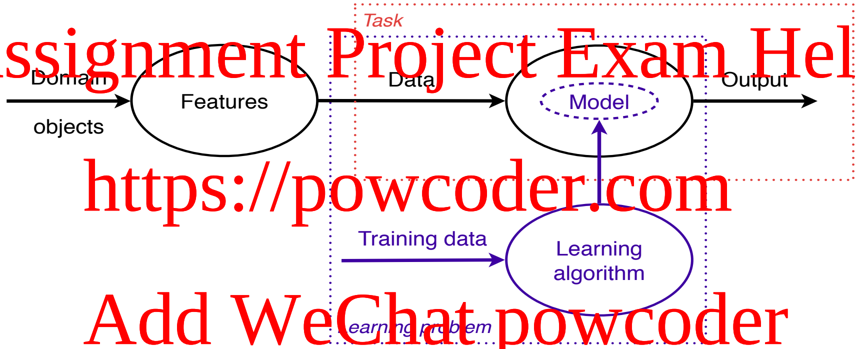
The nice thing about this 'Bayesian' classification scheme is that it can be repeated if you have further evidence. For instance, suppose that the odds in favour of spam associated with the phrase 'blue pill' is estimated at 3:1, and suppose our e-mail contains both 'Viagra' and 'blue pill', then the combined odds are 4:1 times 3:1 is 12:1, which is ample to outweigh the 1:6 odds associated with the low prevalence of spam (total odds are 2:1, or a spam probability of 0.67, up from 0.40 without the 'blue pill').

A rule-based classifier

- if the e-mail contains the word 'Viagra' then estimate the odds of spam as 4:1;
- otherwise, if it contains the phrase 'blue pill' then estimate the odds of spam as 3:1;
- otherwise, estimate the odds of spam as 1:6.

The first rule covers all e-mails containing the word 'Viagra', regardless of whether they contain the phrase 'blue pill', so no overcounting occurs. The second rule *only* covers e-mails containing the phrase 'blue pill' but not the word 'Viagra', by virtue of the 'otherwise' clause. The third rule covers all remaining e-mails: those which neither contain neither 'Viagra' nor 'blue pill'.

How machine learning helps to solve a task



An overview of how machine learning is used to address a given task. A task (red box) requires an appropriate mapping – a model – from data described by features to outputs. Obtaining such a mapping from training data is what constitutes a learning problem (blue box).

Some terminology I

Tasks are addressed by models, whereas learning problems are solved by learning algorithms that produce models.

Some terminology II

Machine learning is concerned with using the right features to build the right models that achieve the right tasks.

Some terminology III

Models lend the machine learning field diversity, but tasks and features give it unity.

Some terminology IV

Does the algorithm require all training data to be present before the start of learning ? If yes, then it is categorised as **batch learning** algorithm.

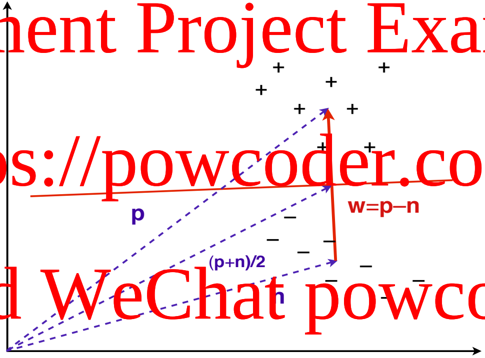
If however, it can continue to learn a new data arrives, it is an **online learning** algorithm.

Some terminology V

If the model has a fixed number of parameters it is categorised as **parametric**.

Otherwise, if the number of parameters grows with the amount of training data it is categorised as **non-parametric**.

Basic linear classifier I

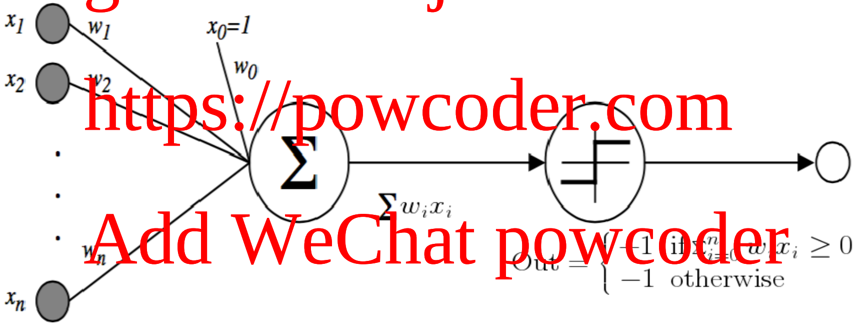


The basic linear classifier constructs a decision boundary by half-way intersecting the line between the positive and negative centres of mass.

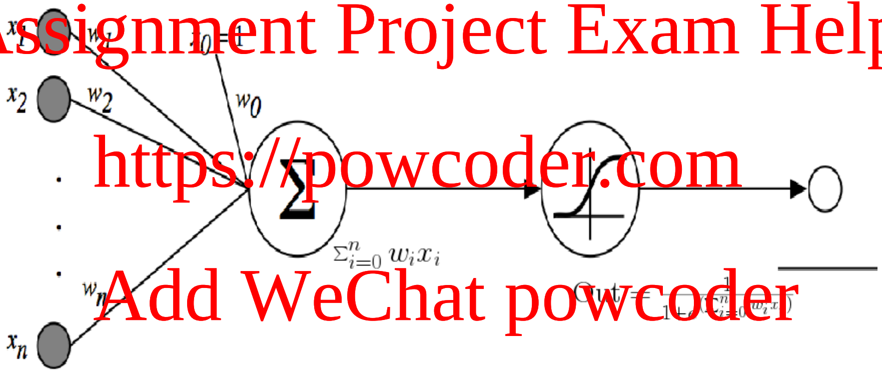
Basic linear classifier II

The basic linear classifier is described by the equation $\mathbf{w} \cdot \mathbf{x} = t$, with $\mathbf{w} = \mathbf{p} - \mathbf{n}$; the decision threshold can be found by noting that $(\mathbf{p} + \mathbf{n})/2$ is on the decision boundary, and hence $t = (\mathbf{p} - \mathbf{n}) \cdot (\mathbf{p} + \mathbf{n})/2 = (||\mathbf{p}||^2 - ||\mathbf{n}||^2)/2$, where $||\mathbf{x}||$ denotes the length of vector $\mathbf{x}$.

Neural Networks I

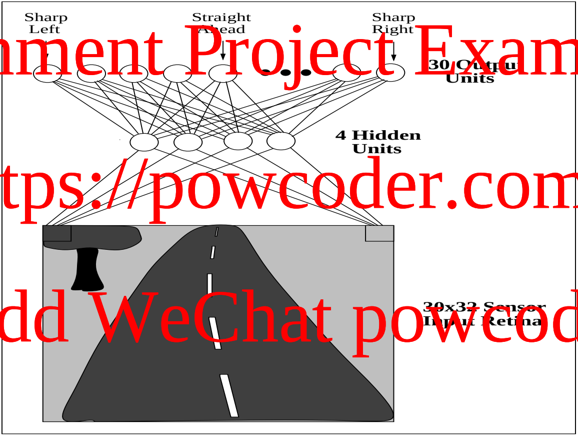


Neural Networks II

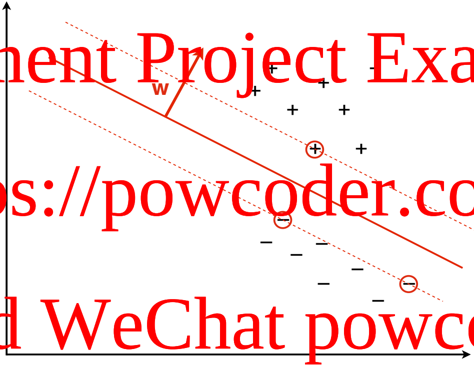


Neural Networks III

Neural Networks IV



Support vector machine



The decision boundary learned by a support vector machine from the linearly separable data from before. The decision boundary maximises the margin, which is indicated by the dotted lines. The circled data points are the support vectors.

A simple probabilistic model

'Viagra' and 'lottery' are two Boolean features; Y is the class variable, with values 'spam' and 'ham'. In each row the most likely class is indicated in bold.

Viagra	lottery	$P(Y = \text{spam}\mid\text{Viagra}, \text{lottery})$	$P(Y = \text{ham}\mid\text{Viagra}, \text{lottery})$
0	0	0.31	**0.69**
0	1	**0.65**	0.35
1	0	**0.80**	0.20
1	1	0.40	**0.60**

Decision rule

Assuming that X and Y are the only variables we know and care about, the posterior distribution $P(Y|X)$ helps us to answer many questions of interest.

- For instance, to classify a new e-mail we determine whether the words 'Viagra' and 'lottery' occur in it, look up the corresponding probability $P(Y = \text{spam}|\text{Viagra}, \text{lottery})$, and predict spam if this probability exceeds 0.5 and ham otherwise.
- Such a recipe to predict a value of Y on the basis of the values of X and the posterior distribution $P(Y|X)$ is called a *decision rule*.

Missing values I

Suppose we skimmed an e-mail and noticed that it contains the word 'lottery' but we haven't looked closely enough to determine whether it uses the word 'Viagra'. This means that we don't know whether to use the second or the fourth row in to make a prediction. This is a problem, as we would predict spam if the e-mail contained the word 'Viagra' (second row) and ham if it didn't (fourth row).

The solution is to average these two rows, using the probability of 'Viagra' occurring in any e-mail (spam or not):

$$\begin{aligned} P(Y|\text{lottery}) = & P(Y|\text{Viagra} = 0, \text{lottery})P(\text{Viagra} = 0) \\ & + P(Y|\text{Viagra} = 1, \text{lottery})P(\text{Viagra} = 1) \end{aligned}$$

Missing values II

For instance, suppose for the sake of argument that one in ten e-mails contain the word 'Viagra', then $P(\text{Viagra} = 1) = 0.10$ and $P(\text{Viagra} = 0) = 0.90$. Using the above formula, we obtain $P(Y = \text{spam}|\text{lottery} = 1) = 0.65 \cdot 0.90 + 0.40 \cdot 0.10 = 0.625$ and $P(Y = \text{ham}|\text{lottery} = 1) = 0.35 \cdot 0.90 + 0.60 \cdot 0.10 = 0.375$. Because the occurrence of 'Viagra' in any e-mail is relatively rare, the resulting distribution deviates only a little from the second row in the data.

Likelihood ratio

As a matter of fact, statisticians work very often with different conditional probabilities, given by the *likelihood function* $P(X|Y)$.

- I like to think of these as thought experiments: if somebody were to send me a spam e-mail, how likely would it be that it contains exactly the words of the e-mail I'm looking at? And how likely if it were a ham e-mail instead?

- What really matters is not the magnitude of these likelihoods, but their ratio: how much more likely is it to observe this combination of words in a spam e-mail than it is in a non-spam e-mail.

- For instance, suppose that for a particular e-mail described by X we have $P(X|Y = \text{spam}) = 3.5 \cdot 10^{-5}$ and $P(X|Y = \text{ham}) = 7.4 \cdot 10^{-6}$, then observing X in a spam e-mail is nearly five times more likely than it is in a ham e-mail.
- This suggests the following decision rule: predict spam if the likelihood ratio is larger than 1 and ham otherwise.

When to use likelihoods

Use likelihoods if you want to ignore the prior distribution or assume it uniform, and posterior probabilities otherwise.

Posterior odds

$$\frac{P(Y = \text{spam}|\text{Viagra} = 0, \text{lottery} = 0)}{P(Y = \text{ham}|\text{Viagra} = 0, \text{lottery} = 0)} = \frac{0.31}{0.69} = 0.45$$

$$\frac{P(Y = \text{spam}|\text{Viagra} = 1, \text{lottery} = 1)}{P(Y = \text{ham}|\text{Viagra} = 1, \text{lottery} = 1)} = \frac{0.40}{0.60} = 0.67$$

$$\frac{P(Y = \text{spam}|\text{Viagra} = 0, \text{lottery} = 1)}{P(Y = \text{ham}|\text{Viagra} = 0, \text{lottery} = 1)} = \frac{0.65}{0.35} = 1.9$$

$$\frac{P(Y = \text{spam}|\text{Viagra} = 1, \text{lottery} = 0)}{P(Y = \text{ham}|\text{Viagra} = 1, \text{lottery} = 0)} = \frac{0.80}{0.20} = 4.0$$

Using a MAP decision rule we predict ham in the top two cases and spam in the bottom two. Given that the full posterior distribution is all there is to know about the domain in a statistical sense, these predictions are the best we can do: they are *Bayes-optimal*.

Example marginal likelihoods

Y	$P(\text{Viagra} = 1 \mid Y)$	$P(\text{Viagra} = 0 \mid Y)$
spam	0.40	0.60
ham	0.12	0.88

Y	$P(\text{lottery} = 1 \mid Y)$	$P(\text{lottery} = 0 \mid Y)$
spam	0.21	0.79
ham	0.13	0.87

Using marginal likelihoods

Using the marginal likelihoods from before, we can approximate the likelihood ratios (the previously calculated odds from the full posterior distribution are shown in brackets):

$$\frac{P(\text{Viagra} = 0|Y = \text{spam})}{P(\text{Viagra} = 0|Y = \text{ham})}\frac{P(\text{lottery} = 0|Y = \text{spam})}{P(\text{lottery} = 0|Y = \text{ham})} = \frac{0.60}{0.88}\frac{0.79}{0.87} = 0.62 \quad (0.45)$$

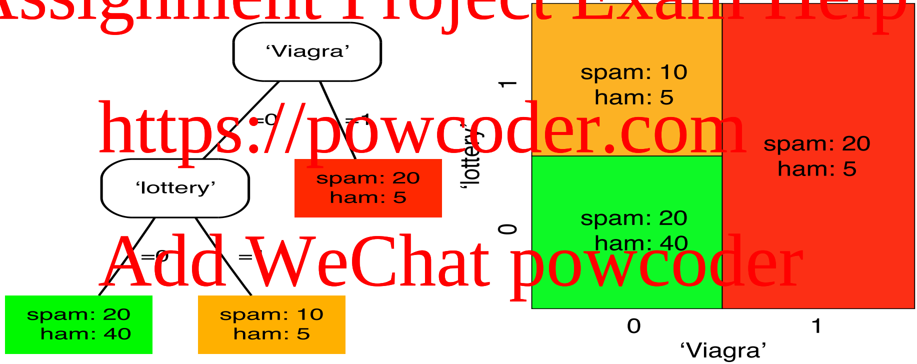
$$\frac{P(\text{Viagra} = 0|Y = \text{spam})}{P(\text{Viagra} = 0|Y = \text{ham})}\frac{P(\text{lottery} = 1|Y = \text{spam})}{P(\text{lottery} = 1|Y = \text{ham})} = \frac{0.60}{0.88}\frac{0.21}{0.13} = 1.1 \quad (1.9)$$

$$\frac{P(\text{Viagra} = 1|Y = \text{spam})}{P(\text{Viagra} = 1|Y = \text{ham})}\frac{P(\text{lottery} = 0|Y = \text{spam})}{P(\text{lottery} = 0|Y = \text{ham})} = \frac{0.40}{0.12}\frac{0.79}{0.87} = 3.0 \quad (4.0)$$

$$\frac{P(\text{Viagra} = 1|Y = \text{spam})}{P(\text{Viagra} = 1|Y = \text{ham})}\frac{P(\text{lottery} = 1|Y = \text{spam})}{P(\text{lottery} = 1|Y = \text{ham})} = \frac{0.40}{0.12}\frac{0.21}{0.13} = 5.4 \quad (0.67)$$

We see that, using a maximum likelihood decision rule, our very simple model arrives at the *Bayes-optimal* prediction in the first three cases, but not in the fourth (‘Viagra’ and ‘lottery’ both present), where the marginal likelihoods are actually very misleading.

A classification tree I



A classification tree combining two Boolean features.

A classification tree II

Each internal node or split is labelled with a feature, and each edge emanating from a split is labelled with a feature value.

Each leaf therefore corresponds to a unique combination of feature values.

Also indicated in each leaf is the class distribution derived from the training set.

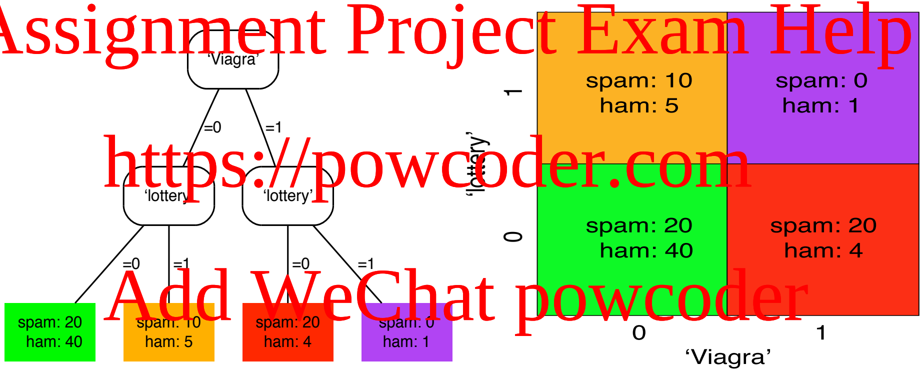
A classification tree partitions the instance space into rectangular regions, one for each leaf. We can clearly see that the majority of ham lives in the lower left-hand corner.

Labelling a classification tree

- The leaves of the classification tree could be labelled, from left to right, as ham – spam – spam, employing a simple decision rule called *majority class*.

- Alternatively, we could label them with the proportion of spam e-mail occurring in each leaf: from left to right, 1/3, 2/3, and 4/5.
- Or, if our task was a regression task, we could label the leaves with predicted real values or even linear functions of some other real-valued features.

A complete classification tree

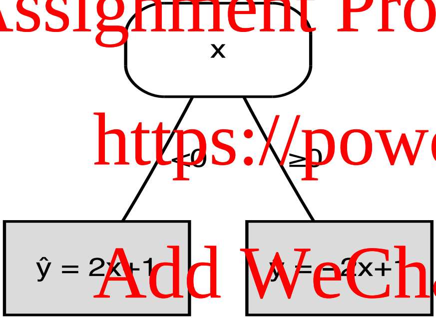


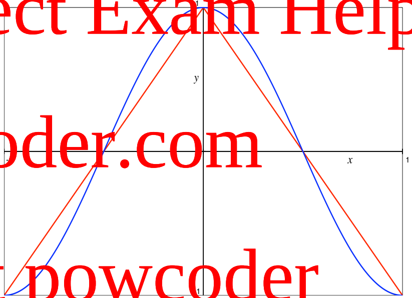
A complete classification tree built from two Boolean features. The corresponding instance space partition is the finest partition that can be achieved with those two features.

Two uses of features

Suppose we want to approximate $y = \cos \pi x$ on the interval $-1 \leq x \leq 1$. A linear approximation is not much use here, since the best fit would be $y = 0$. However, if we split the x-axis in two intervals $-1 \leq x < 0$ and $0 \leq x \leq 1$, we could find reasonable linear approximations on each interval. We can achieve this by using x both as a splitting feature and as a regression variable.

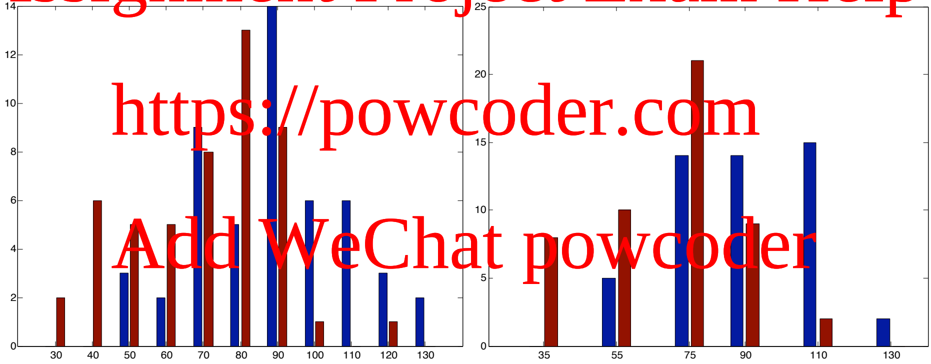
A small regression tree





A regression tree combining a one-split feature tree with linear regression models in the leaves. Note: x used as both a splitting feature and regression variable. At right, function $y = \cos \pi x$ on the interval $-1 \leq x \leq 1$, and piecewise linear approximation by regression tree.

Class-sensitive discretisation I



Class-sensitive discretisation II

Artificial data depicting a histogram of body weight measurements of people with (blue) and without (red) diabetes, with eleven fixed intervals of 10 kilograms width each.

By joining the first and second, third and fourth, fifth and sixth, and the eighth, ninth and tenth intervals, we obtain a discretisation such that the proportion of diabetes cases increases from left to right. This discretisation makes the feature more useful in predicting diabetes.

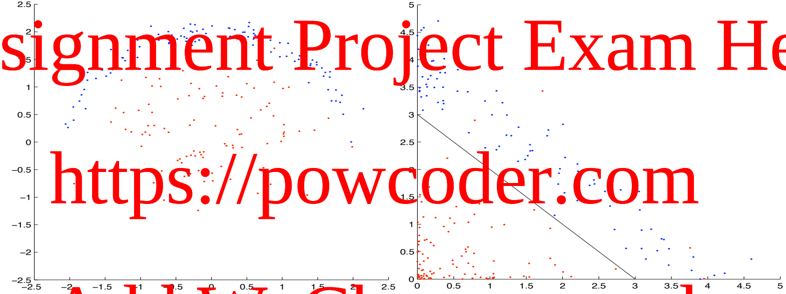
The kernel trick

Let $\mathbf{x}_1 = (x_1, y_1)$ and $\mathbf{x}_2 = (x_2, y_2)$ be two data points, and consider the mapping $(x, y) \mapsto (x^2, y^2, \sqrt{2}xy)$ to a three-dimensional feature space. The points in feature space corresponding to $\mathbf{x}_1$ and $\mathbf{x}_2$ are $\mathbf{x}_1' = (x_1^2, y_1^2, \sqrt{2}x_1y_1)$ and $\mathbf{x}_2' = (x_2^2, y_2^2, \sqrt{2}x_2y_2)$. The dot product of these two feature vectors is

$$\mathbf{x}_1' \cdot \mathbf{x}_2' = x_1^2x_2^2 + y_1^2y_2^2 + 2x_1y_1x_2y_2 = (x_1x_2 + y_1y_2)^2 = (\mathbf{x}_1 \cdot \mathbf{x}_2)^2$$

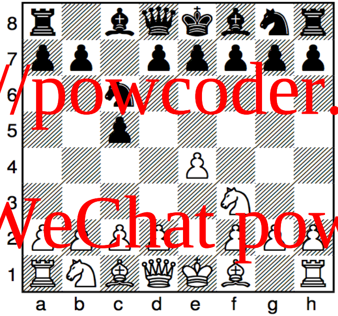
That is, by squaring the dot product in the original space we obtain the dot product in the new space *without actually constructing the feature vectors*! A function that calculates the dot product in feature space directly from the vectors in the original space is called a *kernel* – here the kernel is $\boldsymbol{\kappa}(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1 \cdot \mathbf{x}_2)^2$.

Non-linearly separable data



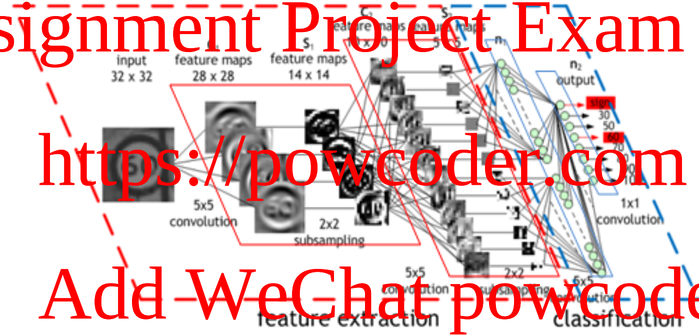
A *linear classifier* would perform poorly on this data. By transforming the original (x, y) data into $(x', y') = (x^2, y^2)$, the data becomes more ‘linear’, and a linear decision boundary $x' + y' = 3$ separates the data fairly well. In the original space this corresponds to a circle with radius $\sqrt{3}$ around the origin.

Where do features come from ?



Feature engineering

Can features be learned ?



Yes, to some extent. For example, in the intermediate layers of a convolutional neural network[1].

[1] Image downloaded 28/2/18 from
http://devblogs.nvidia.com/deep-learning-nutshell-core-concepts/

Where does data come from ?

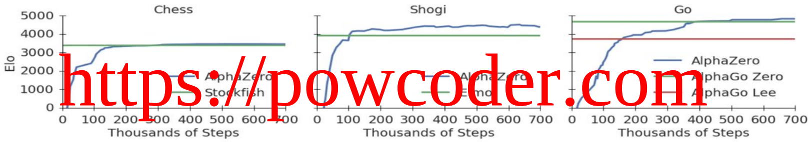


Figure 1: Training *AlphaZero* for 700,000 steps. Elo ratings were computed from evaluation games between different players when given one second per move. **a** Performance of *AlphaZero* in chess, compared to 2016 TCEC world-champion program *Stockfish*. **b** Performance of *AlphaZero* in shogi, compared to 2017 CSA world-champion program *Elmo*. **c** Performance of *AlphaZero* in Go, compared to *AlphaGo Lee* and *AlphaGo Zero* (20 block / 3 day) (*29*).

AlphaZero played around 44 million chess games against an expert (itself).

Tasks for machine learning

The most common machine learning tasks are *predictive*, in the sense that they concern predicting a target variable from features.

- Binary and multi-class classification: categorical target
- Regression: numerical target
- Clustering: hidden target
- Dimensionality reduction: intrinsic structure

Exploratory or *descriptive* tasks are concerned with exploiting underlying structure in the data.

Measuring similarity I

If our e-mails are described by word-occurrence features as in the text classification example, the similarity of e-mails would be measured in terms of the words they have in common. For instance, we could take the number of common words in two e-mails and divide it by the number of words occurring in either e-mail (this measure is called the *Jaccard coefficient*).

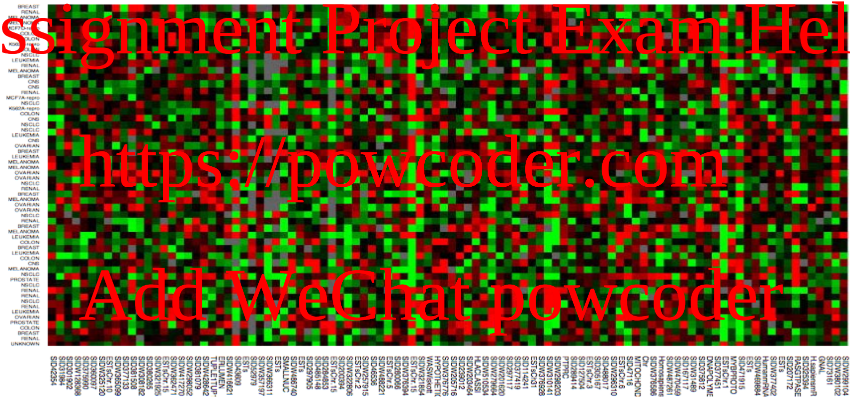
Suppose that one e-mail contains 42 (different) words and another contains 112 words, and the two e-mails have 23 words in common, then their similarity would be $\frac{23}{42+112-23} = \frac{23}{130} = 0.18$.

Measuring similarity II

We can then cluster our e-mails into groups, such that the average similarity of an e-mail to the other e-mails in its group is much larger than the average similarity to e-mails from other groups.

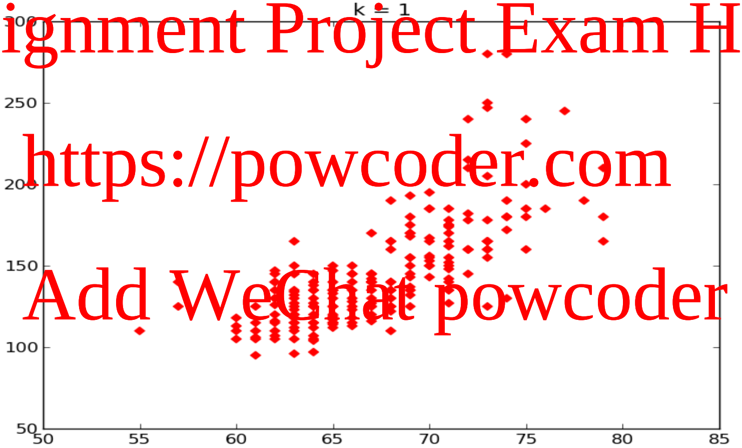
While it wouldn't be realistic to expect that this would result in two nicely separated clusters corresponding to spam and ham – there's no magic here – the clusters may reveal some interesting and useful structure in the data. It may be possible to identify a particular kind of spam in this way, if that subgroup uses a vocabulary, or language, not found in other messages.

Clustering

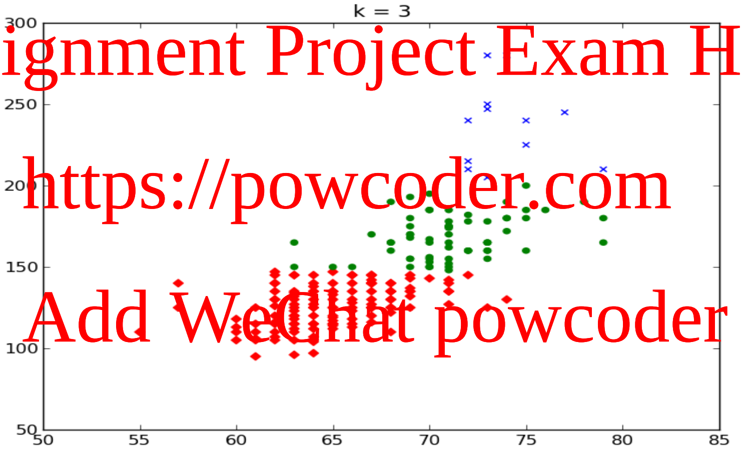


We want to cluster similarly expressed genes in cancer samples.

How many clusters ? I



How many clusters ? II



Looking for structure I

Consider the following matrix:

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 2 & 2 & 2 \\ 0 & 0 & 0 & 1 \\ 1 & 2 & 3 & 2 \\ 1 & 0 & 1 & 1 \\ 0 & 2 & 2 & 3 \end{pmatrix}$$

Imagine these represent ratings by six different people (in rows), on a scale of 0 to 3, of four different films – say *The Shawshank Redemption*, *The Usual Suspects*, *The Godfather*, and *The Big Lebowski*, (in columns, from left to right). *The Godfather* seems to be the most popular of the four with an average rating of 1.5, and *The Shawshank Redemption* is the least appreciated with an average rating of 0.5.

Can you see any structure in this matrix?

Looking for structure II

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 2 & 2 & 2 \\ 0 & 0 & 0 & 1 \\ 1 & 2 & 3 & 2 \\ 1 & 0 & 1 & 1 \\ 0 & 2 & 2 & 3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- The right-most matrix associates films (in columns) with genres (in rows): *The Shawshank Redemption* and *The Usual Suspects* belong to two different genres, say drama and crime, *The Godfather* belongs to both, and *The Big Lebowski* is a crime film and also introduces a new genre (say comedy).
- The tall, 6-by-3 matrix then expresses people's preferences in terms of genres.

Looking for structure III

- Finally, the middle matrix states that the crime genre is twice as important as the other two genres in terms of determining people's preferences.

The philosophical problem

Deduction: derive specific consequences from general theories

Induction: derive general theories from specific observations

Deduction is well-founded (mathematical logic).

Induction is (philosophically) problematic – induction is useful since it often seems to work – an inductive argument !

Generalisation - the key objective of machine learning

What we are really interested in is *generalising* from the sample of data in our training set. This can be stated as:

The inductive learning hypothesis

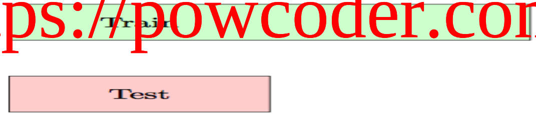
> *Any hypothesis found to approximate the target (true) function well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples.*

A corollary of this is that it is necessary to make some assumptions about the type of target function in a task for an algorithm to go beyond the data, i.e., generalise or learn.

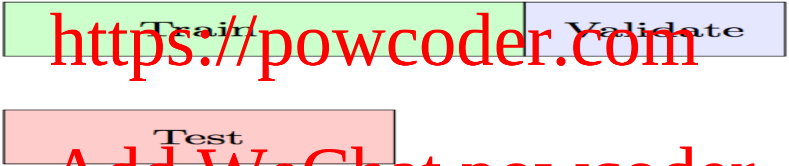
Cross-validation I

There are certain parameters that need to be estimated during learning. We use the data, but NOT the training set, OR the test set. Instead, we use a separate *validation* or *development* set.

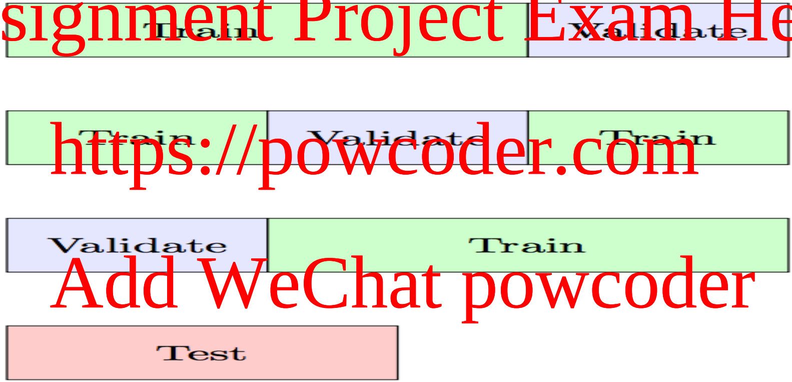
Cross-validation II



Cross-validation III



Cross-validation IV



Contingency table I

Two-class prediction case:

	Predicted Class	
Actual Class	Yes	No
Yes	True Positive (TP)	False Negative (FN)
No	False Positive (FP)	True Negative (TN)

Contingency table II

Classification Accuracy on a sample of labelled pairs $(x, c(x))$ given a learned classification model that predicts, for each instance x, a class value $\hat{c}(x)$:

$$\text{acc} = \frac{1}{|\text{Test}|} \sum_{x \in \text{Test}} I[\hat{c}(x) = c(x)]$$

where Test is a test set and $I[]$ is the indicator function which is 1 iff its argument evaluates to true, and 0 otherwise.

Classification Error is 1-acc.

Overfitting

Imagine you are preparing for your *Machine Learning 101* exam. Helpfully, your professor has made previous exam papers and their worked answers available online. You begin by trying to answer the questions from previous papers and comparing your answers with the model answers provided.

Unfortunately, you get carried away and spend all your time on memorising the model answers to all past questions. Now, if the upcoming exam completely consists of past questions, you are certain to do very well. But if the new exam asks different questions about the same material, you would be ill-prepared and get a much lower mark than with a more traditional preparation.

In this case, one could say that you were *overfitting* the past exam papers and that the knowledge gained didn't *generalise* to future exam questions.

The Bias-Variance Decomposition

Error (particularly MSE) can be shown to have two components:

Bias: error due to mismatch between target function and functions learnable by the algorithm that was applied to the task

Variance: error due to variation between training sets sample from the distribution on data generated by the target function

Inductive Bias

All models are wrong, but some models are useful

Box & Draper (1987)

Inductive Bias

Confusingly, "inductive bias" is *NOT* the same "bias" as in the "bias-variance" decomposition.

"Inductive bias" is the combination of assumptions and restrictions placed on the models and algorithms used to solve a learning problem.

Essentially it means that the algorithm and model combination you are using to solve the learning problem is appropriate for the task.

Success in machine learning requires understanding the inductive bias of algorithms and models, and choosing them appropriately for the task[2].

[2] Even true for "deep learning", but watch Andrew Ng's talk on this at `http://www.youtube.com/watch?v=F1ka6a13S9I&t=48s`.

No Free Lunch

Uniformly averaged over all target functions, the expected off-training-set error for all learning algorithms is the same.

Wolpert (1996)

No Free Lunch

Owing to the "No Free Lunch" Theorem, there is no universally best machine learning algorithm.

Some learning algorithms perform better than others on certain tasks since their assumptions about the type of target function are more appropriate for the learning task in that application.

On some other tasks, those assumptions, and hence the algorithms, may perform much worse than others.

These assumptions form the inductive bias of the learning algorithm.

Ethics of machine learning

Machine learning algorithms are now widely available and increasingly used on “big data” in commercial, medical, legal and scientific applications

These applications bring many benefits, but as they become more widespread they wll start to impact most people as part of daily life

In some of these applications ethical and moral implications of the use of machine learning must be considered

For example, can the use of machine learning be biased or unfair because the training data does not adequately represent all of the data to which the model may be applied ? Here is a recent paper discussing some of these issues and giving some recommendations[3].

[3] Zook et al. (2017) Ten simple rules for responsible big data research. PLoS Comput Biol 13(3): e1005399. `http://doi.org/10.1371/journal.pcbi.1005399`.

Summary

The purpose of this introductory lecture was, from a high-level, to survey the landscape that we will explore in this course.

We have motivated the subject matter, outlined what parts of machine learning will be addressed in this course, and introduced some key ideas and terminology.

Throughout this lecture we have deliberately avoided details like programming languages, machine learning tools and software libraries. These change very rapidly, although the core technical ideas and challenges remain.

Following this lecture you should have a clear idea of the course scope. In the remaining lectures we will expand on the topics we have just introduced and go into more detail. You will also get to work on practical applications of what we have covered.