Foundations: Machine Learning

* Introduction to Machine Learning

  Outline:
  - Introduction to Machine Learning
  - Applications of Machine Learning
  - Useful pre-requisites

* Introduction

* Introduction Part II

* Prerequisites

  - Programming experience (esp. Python)
  - Statistics
  -

* ML is the ROX

* Definition of Machine Learning

  Building computational artifacts that learn over time based on experience. Not just building but the mathematics, science, engineering and computing behind the artifacts

* Supervised Learning

  Using information from labelled datasets to label new datasets.

  Function approximation or function induction

* Induction and Deduction

You assume you have a well behaved function which is consistent with your data and you use this to generalise.

Problem: Inductive bias

Induction:- Example → General Rule
Deduction:- General Rule → Example

* Unsupervised Learning

Derive structure from the data.

Pixels → $\boxed{\text{Description}}$ → Summaries → $\boxed{\text{Function Approximation}}$ → labels

Unsupervised Learning — Supervised Learning

* Reinforcement Learning

Learning from delayed reward

* Comparison of These Parts of Machine Learning

All these problems can be formulated as some kind of optimisation problem

supervised learning → labels data well
reinforcement learning → behaviour scores well
unsupervised learning → cluster scores well

DATA IS CENTRAL!

* Machine Learning Basics

Outline :
- How modern companies use Machine Learning
- Stanley at the DARPA Grand Challenge
- Machine Learning taxonomy
- Overfitting : Occam's Razor
- Example problem : Spam Detection
- Linear methods for supervised learning

* What is Machine Learning

* Gibbs Sampling

... using Markov Chain Monte Carlo (MCMC)

Here we sample one variable at a time

+c	+s	-r	-w
+c	-s	-r	-w
+c	-s	+r	-w

this is consistent

Unit 5: Machine Learning I (Supervised Learning)

→ Bayes networks = reason with known models
→ Machine learning = learn models from data

Taxonomy of machine learning:

What? parameters, structure, hidden concepts
What from? supervised, unsupervised, reinforcement
What for? prediction, iagnostics, summarisation ...
How? passive, active (the learning agent is...), offline, online (while/not while the data is being generated)
Outputs? classification, regression
Details? generative (generate a model), discriminative (distinguish between samples)

Supervised Learning

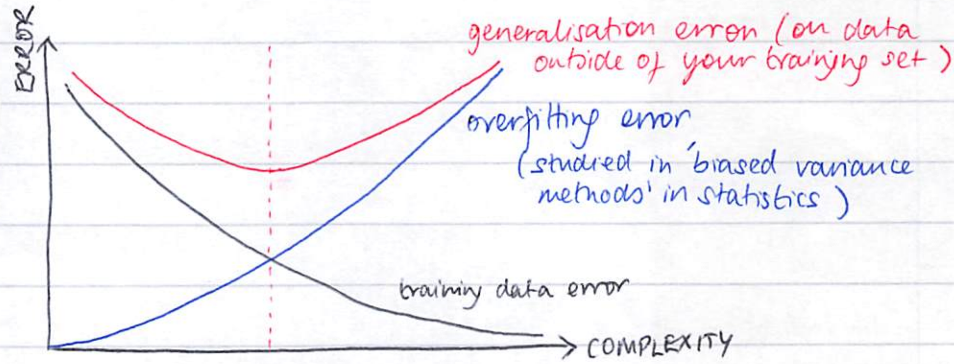
$$\underbrace{\{x_1, x_2, \ldots, x_n\}}_{\text{features of the data}} \to \underbrace{Y}_{\text{feature we want to predict}}$$

$$\begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1n} & \rightarrow & Y_1 \\ X_{21} & X_{22} & \cdots & X_{2n} & \rightarrow & Y_2 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ \underbrace{X_{m1} \quad X_{m2} \quad \cdots \quad X_{mn}}_{X_m} & & & & \rightarrow & Y_m \end{bmatrix} \quad \text{data}$$

Want to find the function $f(X_m) = Y_m$ so it predicts well Y for X not in our training set

Occam's / Okham's Razor
=> Everything else being equal, choose the less complex hypothesis
There tends to be a trade off between fit (to the training data) and complexity (of the hypothesis)

(GOOD) FIT $\longleftrightarrow$ COMPLEXITY (LOW)



generalisation error = overfitting error + training data error

Spam detection - A classification problem

Email → SPAM (don't show it to the reader)
Email → HAM (worth passing on to the reader)

Y can take one of two values - SPAM / HAM hence a classification problem.

Maximum likelihood

suppose

$$S\ S\ S\ H\ H\ H\ H\ H \qquad (8 \text{ messages})$$

let

$$p(S) = \pi$$

we want to find π which maximises the likelihood of the training set assuming that each email is drawn independently from an identical distribution

$$p(y_i) = \begin{cases} \pi & \text{if } y_i = S = 1 \\ 1-\pi & \text{if } y_i = H = 0 \end{cases}$$

$$p(y_i) = \pi^{y_i} (1-\pi)^{(1-y_i)}$$

$$p(\text{data}) = \prod_{i=1}^{m} p(y_i)$$

$$= \pi^{\text{count}(y_i=1)} (1-\pi)^{\text{count}(y_i=0)}$$

$$\ln(p(\text{data})) = \text{count}(y_i=1) \ln \pi + \text{count}(y_i=0) \ln(1-\pi)$$

$$\frac{d}{d\pi}\left(\ln(p(\text{data}))\right) = \frac{\text{count}(y_i=1)}{\pi} - \frac{\text{count}(y_i=0)}{1-\pi}$$

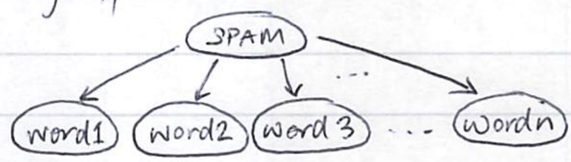
$$\frac{d}{d\pi}\left(\ln(p(\text{data}))\right) = 0 \Rightarrow \frac{\text{count}(y_i=1)}{\pi} = \frac{\text{count}(y_i=0)}{1-\pi}$$

$$\Rightarrow \text{count}(y_i=0)\,\pi = \text{count}(y_i=1)(1-\pi)$$

$$\Rightarrow [\text{count}(y_i=0) + \text{count}(y_i=1)]\,\pi = \text{count}(y_i=1)$$

$$\Rightarrow \pi = \frac{\text{count}(y_i=1)}{\text{count}(y_i=1) + \text{count}(y_i=0)}$$

* Bayes networks examples: Bag of words model
Naive Bayes Network:
Detecting SPAM emails:



SPAM	HAM
OFFER IS SECRET	PLAY SPORTS TODAY
CLICK SECRET LINK	WENT PLAY SPORTS
SECRET SPORTS LINK	SECRET SPORTS EVENT
	SPORTS IS TODAY
	SPORTS COSTS MONEY

	Dictionary	SPAM	HAM
1	OFFER	I	
2	IS	I	I
3	SECRET	III	I
4	CLICK	I	
5	LINK	II	
6	SPORT	I	𝍸
7	PLAY		II
8	TODAY		II
9	WENT		I
10	EVENT		I
11	COSTS		I
12	MONEY		I
	Total	9	15

Using Maximum likelihood...

$P(\text{SPAM}) = 3/8$

$P(\text{"SECRET"} \mid \text{SPAM}) = 1/3$

$P(\text{"SECRET"} \mid \text{HAM}) = 1/15$

$P(\text{"IS"} \mid \text{SPAM}) = 1/9$

$P(\text{"IS"} \mid \text{HAM}) = 1/15$

$P(\text{"TODAY"} \mid \text{SPAM}) = 0$

$P(\text{"TODAY"} \mid \text{HAM}) = 2/15$

Here it's the probability of the word in a message being "..." given ...

Q1 Message M = "SPORTS"

$$P(\text{SPAM} \mid M) = \frac{\frac{3}{8}\cdot\frac{1}{9}}{\frac{3}{8}\cdot\frac{1}{9} + \frac{5}{8}\frac{1}{3}} = \frac{3}{3+15} = \frac{1}{6}$$

Q2 M = "SECRET IS SECRET"

$$P(\text{SPAM} \mid M) = \frac{\frac{3}{8}\frac{1}{3}\frac{1}{9}\frac{1}{3}}{\frac{3}{8}\frac{1}{3}\frac{1}{9}\frac{1}{3} + \frac{5}{8}\frac{1}{15}\frac{1}{15}\frac{1}{15}} = \frac{\cancel{3}\times5\times\cancel{\frac{1}{9}}\times5}{\cancel{3}\times5\times\cancel{\frac{1}{9}}\times5 + \cancel{5}\times1\times\cancel{\frac{1}{5}}\times1}$$

$$= \frac{25}{26}$$

Q3 M = "TODAY IS SECRET"

$$P(\text{SPAM} \mid M) = \frac{\frac{3}{8}\cdot0\cdot\frac{1}{9}\cdot\frac{1}{3}}{\frac{3}{8}\cdot0\cdot\frac{1}{9}\cdot\frac{1}{3} + \frac{5}{8}\cdot\frac{2}{15}\cdot\frac{1}{15}\cdot\frac{1}{15}} = 0$$

The third example is a poor estimate clearly. Here we are overfitting.

* Laplace Smoothing

Technique to deal with overfitting.

Maximum Likelihood : $p(x) = \dfrac{\text{count}(x)}{N}$

Laplace Smoothing : $p(x) = \dfrac{\text{count}(x) + k}{N + k|x|}$

count (x) = # of occurences of this value of x
$|x|$ is the number of values which x can have
k is the smoothing parameter

This is like adding k dummy emails to each class (SPAM and HAM) each containing all the words in the dictionary.

Using Laplace Smoothing for the Bayes networks example ...

$$k=1 \Rightarrow P(\text{SPAM}) = \frac{3+1}{8+2} = \frac{4}{10} = \frac{2}{5}$$

$$\Rightarrow P(\text{HAM}) = \frac{3}{5}$$

$$P(\text{"TODAY"} \mid \text{SPAM}) = \frac{0+1}{9+12} = \frac{1}{21}$$

$$P(\text{"TODAY"} \mid \text{HAM}) = \frac{2+1}{15+12} = \frac{3}{27} = \frac{1}{9}$$

$$P(\text{"IS"} \mid \text{SPAM}) = \frac{1+1}{9+12} = \frac{2}{21}$$

$$P(\text{"IS"} \mid \text{HAM}) = \frac{1+1}{15+12} = \frac{2}{27}$$

$$P(\text{"SECRET"} \mid \text{SPAM}) = \frac{3+1}{9+12} = \frac{4}{21}$$

$$P(\text{"SECRET"} \mid \text{HAM}) = \frac{1+1}{15+12} = \frac{2}{27}$$

$$P(\text{SPAM} \mid \text{"TODAY IS SECRET"})$$

$$= \frac{\frac{\cancel{2}}{\cancel{5}} \cdot \frac{1}{21} \cdot \frac{\cancel{2}}{21} \cdot \frac{4}{21}}{\frac{\cancel{2}}{\cancel{5}} \cdot \frac{1}{21} \cdot \frac{\cancel{2}}{21} \cdot \frac{4}{21} + \frac{3}{\cancel{5}} \cdot \frac{1}{9} \cdot \frac{\cancel{2}}{27} \cdot \frac{\cancel{2}}{27}}$$

$$= \frac{4}{4 + \frac{1}{3} \cdot \frac{1}{27} \cdot \frac{1}{27} \cdot 21.21.21}$$

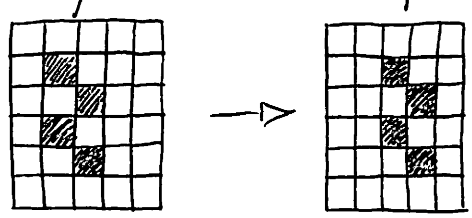
$$= \frac{4}{4 + \frac{1}{9} \cdot \frac{1}{9} \cdot 7.7.7} = \frac{4}{4 + 343/81}$$

Advanced Spam filters
- → known spamming IP?
- → have you emailed the person before?
- → have 1K other people received the message?
- → is the email header and IP consistent?
- → is the email all caps
- → do inline URLs point to where they say?
- → are you addressed by name?

All these can be used in Naive Bayes

* Digital Recognition

Recognising handwritten digits - could try to use Naive-Bayes on 16 x 16 pixels but this doesn't deal



well with shifting. To deal with this one could use Input smoothing in a different way. One could mix pixel counts with those of the neighbouring pixels so if they are shifted we get similar statistics. Here we convolve the input with a Gaussian variable. This may give better results than just using the raw pixel values themselves.

Actually Naive-Bayes is not a great choice for this problem since it turns out that conditional independence of the pixels in this case is too strong an assumption.

* Overfitting Prevention

We talked about
- → Occam's Razor (trade off between fit / prediction)
- → Laplace smoothing } How do we choose the
- → Input smoothing } smoothing parameter?

Cross-validation:

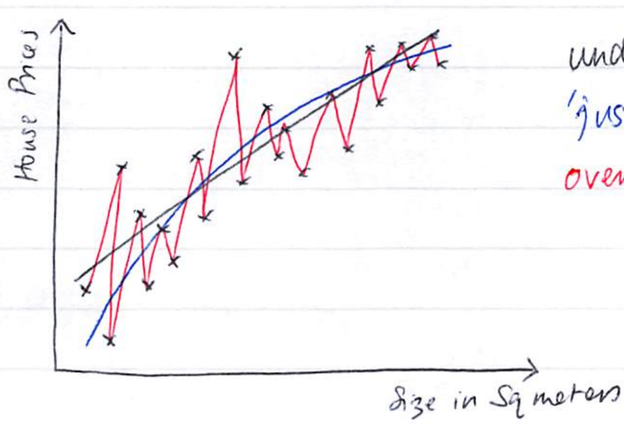
TRAINING DATA		
TRAIN	CROSS-VALIDATE	TEST
80 %	10 %	10 %
↓ parameters	↓ k	↓ verify performance and report

← Typical split of your training data.

find k which gives optimal performance on the predictions for the cross-validation data maybe iteratively

* Classification vs Regression

y is now continuous rather than discrete ⇒ regression problem



underfit
'just right'
overfit

$$\text{DATA} \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1n} & \to & Y_1 \\ X_{21} & X_{22} & \cdots & X_{2n} & \to & Y_2 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ X_{m1} & X_{m2} & \cdots & X_{mn} & \to & Y_m \end{bmatrix}$$

$$f(X) = w_1 X + w_0$$
$$f(X) = wX + w_0$$

We want to find $f \mid f(X) = Y$. We do this by minimising the loss function

$$\text{Loss} = \sum_j' (Y_j - w_1 X_j - w_0)^2$$ sum square erro

solution $w^* = \underset{w}{\arg\min} \{ \text{LOSS} \}$

* Minimising quadratic loss

$$\min_w \sum_{i=1}^{m}' (y_i - w_1 x_i - w_0)^2 = L$$

$$\frac{\partial L}{\partial w_0} = -2 \sum_{i=1}^{m}' (y_i - w_1 x_i - w_0)$$

$$\frac{\partial L}{\partial w_0} = 0 \Rightarrow m w_0 = \sum_{i=1}^{m}' (y_i - w_1 x_i)$$

$$\Rightarrow \boxed{w_0 = \frac{1}{m}\left(\sum_{i=1}^{m}' y_i - w_1 \sum_{i=1}^{m}' x_i \right)}$$

$$\frac{\partial L}{\partial w_1} = -2 \sum_{i=1}^{m}' (y_i - w_1 x_i - w_0) x_i$$

$$\frac{\partial L}{\partial w_1} = 0 \Rightarrow w_1 \sum_{i=1}' x_i^2 = \sum_{i=1}' (y_i - w_0) x_i$$

$$\Rightarrow w_1 \sum_{i=1}^{m}' x_i^2 + w_0 \sum_{i=1}^{m}' x_i = \sum_{i=1}^{m}' x_i y_i$$

$$\Rightarrow w_1 \sum_{i=1}^{m}' x_i^2 + \frac{1}{m} \sum_{i=1}^{m}' (y_i - w_1 x_i) \sum_{i=1}^{m}' x_i = \sum_{i=1}^{m}' x_i y_i$$

$$\Rightarrow w_1 \sum_{i=1}^{m}' x_i^2 + \frac{1}{m} \sum_{i=1}^{m}' y_i \sum_{i=1}^{m}' x_i \quad \frac{w_1}{m}\left(\sum_{i=1}^{m}' x_i\right)^2 = \sum_{i=1}^{m}' x_i y_i$$

$$\Rightarrow \boxed{w_1 = \frac{\sum_{i=1}^{m}' x_i y_i - \frac{1}{m} \sum_{i=1}^{m}' x_i \sum_{i=1}^{m}' y_i}{\sum_{i=1}^{m}' x_i^2 - \frac{1}{m}\left(\sum_{i=1}^{m}' x_i\right)^2}}$$

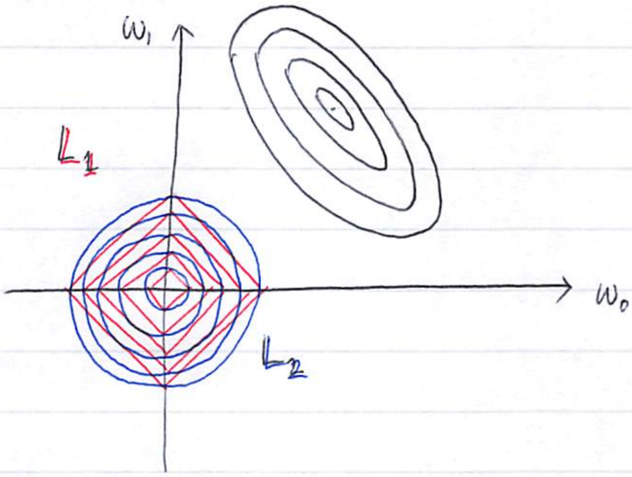
Problems with linear regression
- → non-linear data
- → outliers
- → classification problems

For classification problems we can use logistic regression.

$$\frac{1}{1+e^{-f(x)}}$$

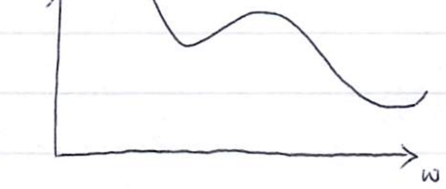
Regularisation is used for complexity control

$$\text{Loss} = \text{Loss (data)} + \text{Loss (parameters)}$$
$$= \sum_j (y_j - w_1 x_j - w_0)^2 + \sum_i |w_i|^p$$



$p = 1$: L_1 regularisation
$p = 2$: L_2 regularisation

* Minimisation of more complicated loss functions



Gradient descent
w^0
$$w^{i+1} \leftarrow w^i - \alpha \frac{\partial L}{\partial w_i}$$

* Gradient descent implementation

$$L = \sum_j (y_j - w_1 x_j - w_0)^2 \longrightarrow \min$$

$$\frac{\partial L}{\partial w_1} = -2 \sum_j (y_j - w_1 x_j - w_0) x_j$$

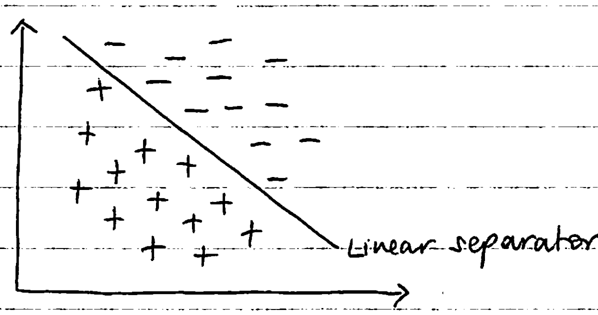
$$\frac{\partial L}{\partial w_0} = -2 \sum_j (y_j - w_1 x_j - w_0)$$

$w_0 = w_0^0$ and $w_1 = w_1^0$

$$w_1^i = w_1^{i-1} - \alpha \frac{\partial L}{\partial w_1}(w_1^{i-1})$$

$$w_0^i = w_0^{i-1} - \alpha \frac{\partial L}{\partial w_0}(w_0^{i-1})$$

* Perceptron algorithm

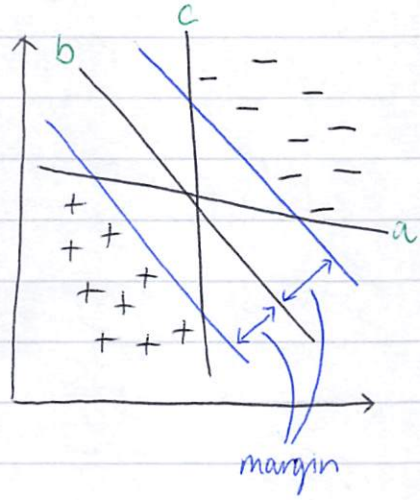


$$f(x) = \begin{cases} 0 & \text{if } w_1 x + w_0 < 0 \\ 1 & \text{if } \underbrace{w_1 x + w_0}_{\text{linear function}} \geq 0 \end{cases}$$

Start with a random guess for w_0 and w_1

$$w_i^k \leftarrow w_i^{k-1} + \alpha (y_j - f(x_j))$$

* Linear separators



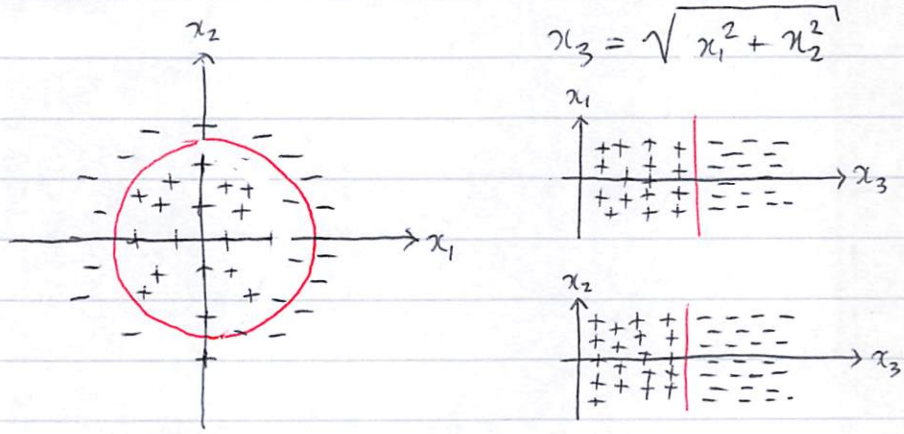
Separator b is preferable to a and c because of the large margin.

Perceptron only finds a linear separator - not the "best" one.

Maximum margin algorithms:
- -> support vector machines
- -> boosting

Support vector machines:
These use a "Kernel Trick" to find features which turn complex non-linear decision boundaries into linear ones

Illustration:



Linear Methods
- Regression vs Classification
- Exact vs Iterative solutions
- Smoothing
- Non-linear problems