# Machine Learning and Pattern Recognition

Notes from lectures given by Dr Arno Onken in 2024

Thomas Davies

Last updated: September 21, 2024

## Contents

## Introduction

These are notes that were taken in 2024 for the MLPR course with lectures given by Dr Arno Onken. They were mostly taken live in lectures, in particular, all errors are almost surely mine. The provided lecture notes can be found here. These are not endorsed by the lecturer, and are mostly for my own reference. It should be noted that these notes will be much more maths focused than the ones provided in lectures, this is mainly for my own comfort (I come from a maths background). I may also later add a GitHub link with python (or maybe rust) implementations of the methods provided from scratch (where scratch is defined as starting from some base linear algebra library like numpy).

## Lecture 1

This lecture was primarily describing why one would want to study machine learning, and gave some interesting examples such as face detection. I would recommend reading the provided notes, however I don't think there is any value in me reproducing this text.

## Lecture 2

Very very often, machine learning problems can be reduced in some form to linear regression (see the Kernel Method, Gaussian Process Regression, and more exotically the Kernel Regime for neural networks). So it is important to understand the basics of what we are doing in linear regression. First we need a couple of basic definitions.

**Definition 2.1.** *An **affine function** $f : \mathbb{R}^k \to \mathbb{R}^m$ is one of the form*

$$f_{A,b}(x) = Ax + b$$

*where $x \in \mathbb{R}^k$, $A \in \mathbb{R}^{m\times k}$, and $b \in \mathbb{R}^m$. Note that notationally $m, k$ are often renamed and can change based on the context (although it is normally easy to infer the dimensions).*

In the lectures, $m$ was taken to be 1, and $A$ was denoted as $w^T$, and none of the derivations of the ensuing results were given, however I think there is value in doing them. We now also assume that we have some **training data**.

**Definition 2.2.** ***Supervised training data*** *is a pair of "labelled" data*

$$\mathcal{T} = \{(x^{(n)}, y^{(n)}) \mid n = 1, 2, \ldots, N\}$$

*In the unsupervised case (which we cover later in the course), we are missing $y^{(n)}$ but still hope to be able to recover our underlying function via building structure into our model.*

We assume that it was sampled from a **linear model** i.e. there is some approximate relationship $f(x_n) \approx y_n$ for some affine function $f$ (for a slightly better way to describe this see ordinary least squares). We would like a way to recover a good estimate of $A$ and $b$ from out training data, one way to do this is to minimise the loss in the 2-norm (squared error). We will denote our approximated (affine) function as $\tilde{f}$, and our approximated parameters as $\tilde{A}$, and $\tilde{b}$. The loss function will be denoted as $\mathcal{L}(g)$. Using the 2-norm (with no regularisation).

$$\mathcal{L}(\tilde{f}) = \sum_{n=1}^{N} \left\| y^{(n)} - \tilde{f}(x^{(n)}) \right\|_2^2 = \sum_{n=1}^{N} \sum_{i=1}^{k} (y_i^{(n)} - \tilde{f}_i(x^{(n)}))^2 \tag{1}$$

We seek (as we normally do in machine learning) to minimise this loss function over our parameters $\tilde{A}, \tilde{b}$. Note that as a sanity check, if we perfectly recovered $f$, that the loss would be 0 i.e. $\mathcal{L}(f) = 0$. Now, let's try find a closed form solution (under certain regularity conditions which we will state as they arise).

**Definition 2.3.** *A* ***design matrix*** $X \in \mathbb{R}^{N \times k}$*, and* ***output matrix*** $Y \in \mathbb{R}^{N \times m}$ *are stacked versions of our training data.*

$$X = \begin{bmatrix} - & x^{(1)T} & - \\ - & x^{(2)T} & - \\ & \vdots & \\ - & x^{(N)T} & - \end{bmatrix} \qquad Y = \begin{bmatrix} - & y^{(1)T} & - \\ - & y^{(2)T} & - \\ & \vdots & \\ - & y^{(N)T} & - \end{bmatrix}$$

We can re write the loss in equation 1 by subbing in our parameters, and setting $\tilde{b} = 0$ (this will be expounded on later) as follows.

$$\mathcal{L}(\tilde{f}) = \sum_{n=1}^{N} (y^{(n)} - \tilde{A}x^{(n)})^T (y^{(n)} - \tilde{A}x^{(n)}) \tag{2}$$

Now, differentiating with respect to $\tilde{A}$, we get the following.

$$\frac{\delta \mathcal{L}}{\partial \tilde{A}}(\tilde{f}) = 2 \sum_{n=1}^{N} (y^{(n)} - \tilde{A}x^{(n)}) \tag{3}$$

Setting the derivative to 0 (it is clearly a global minima as when $\tilde{A} \to \infty$ we have $\mathcal{L}(\tilde{f}) \to \infty$ under the regularity condition stated below), and rearranging, we get the closed form solution.

$$\tilde{A} = \tag{4}$$