Mathematics for Machine Learning

In this session ...

- Calculus
- Linear Algebra
- Probability
- Other methods

There is no ML in this session!

This module presents a number of themes associated with Machine Learning (ML). The lectures in this modules cover several advanced topics in this field.

This lecture only covers the advanced mathematical methods that are/to be employed by other tutors in their Machine Learning lectures. We will not be doing ML or applications of the maths to ML.

Extrema

Vector Calculus

If $f = f(x, y, z)$ is a function of three variables, then the *gradient of* f is

$$\nabla f(x, y, z) = f_x(x, y, z)\,\mathbf{i} + f_y(x, y, z)\,\mathbf{j} + f_z(x, y, z)\,\mathbf{k}.$$

The symbol ∇, called the **del** operator, is a vector differential operator symbolised by

$$\nabla = \mathbf{i}\frac{\partial}{\partial x} + \mathbf{j}\frac{\partial}{\partial y} + \mathbf{k}\frac{\partial}{\partial z}.$$

It has properties similar to the operator d/dx. Standing alone it is meaningless; however, if it operates on $f(x, y, z)$ it produces the three-dimensional vector function given above.

In applications, the gradient $\nabla f(x, y, z)$ is sometimes denoted by **grad** $f(x, y, z)$.

Example 1: Calculate ∇f for $f(x, y, z) = x^2 + yz$.

$$f_x = 2x; \ f_y = z; \ f_z = y,$$

to give

$$\nabla f(x, y, z) = 2x\mathbf{i} + z\mathbf{j} + y\mathbf{k}$$

Example 2: If $f(x,y,z) = yz^3 - 2x^2$; find the gradient of f at the point $P(2,-3,1)$.

$$f_x = -4x;\ f_y = z^3;\ f_z = 3yz^2,$$

to give

$$\nabla f(x,y,z) = -4x\mathbf{i} + z^3\mathbf{j} + 3yz^2\mathbf{k}$$

Hence, at $P(2,-3,1)$

$$\nabla f(x,y,z) = -8\mathbf{i} + \mathbf{j} - 9\mathbf{k}.$$

Lagrange Multipliers

Let $f(x, y, z)$ and $g(x, y, z)$ have continuous first order partial derivatives, and suppose f has an extremum $f(x_0, y_0, z_0)$ when (x, y, z) is subject to the constraint $g(x, y, z) = 0$. If $\nabla g(x_0, y_0, z_0) \neq \mathbf{0}$ then there is a real number λ such that

$$\nabla f(x_0, y_0, z_0) = \lambda \nabla g(x_0, y_0, z_0)$$

λ is called a *Lagrange multiplier*.

Example 3: Find the extrema of $f(x,y) = xy$; if (x,y) is restricted to the ellipse $4x^2 + y^2 = 4$.

Solution: In this example the constraint is $g(x,y) = 4x^2 + y^2 - 4 = 0$. Setting $\nabla f(x,y) = \lambda \nabla g(x,y)$, we obtain

$$y\mathbf{i} + x\mathbf{j} = \lambda(8x\mathbf{i} + 2y\mathbf{j}).$$

Equating coefficients

$$y = 8\lambda x;\ x = 2\lambda y,$$

together with $4x^2 + y^2 - 4 = 0$. A number of ways to solve, here we eliminate y.

$$x - 2\lambda(8\lambda x) = 0.$$

So solve

$$x\left(1 - 16\lambda^2\right) = 0$$

to get the values $x = 0$ or $\lambda = \pm 1/4$.

$x = 0$: using $4x^2 + y^2 - 4 = 0 \to y = \pm 2$.

Possible choices for extrema: $(0, 2)$; $(0, -2)$.

If $\lambda = \pm 1/4$, then $y = 8\lambda x = \pm 2x$. Substitute in $4x^2 + y^2 - 4 = 0$ to get $x = \pm\sqrt{2}/2$.

The corresponding y values are $y = \pm\sqrt{2}$.

This gives the following points

$$\left(\sqrt{2}/2, \pm\sqrt{2}\right)$$
$$\left(-\sqrt{2}/2, \pm\sqrt{2}\right)$$

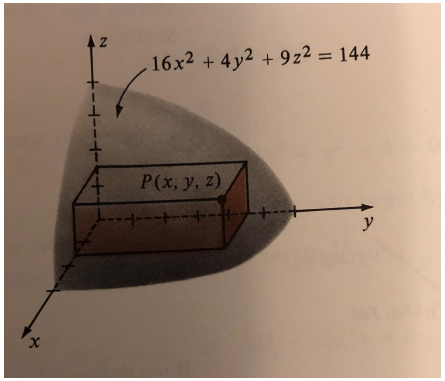
$$\begin{array}{cc} (x,y) & f(x,y) \\ (0,2) & 0 \\ (0,-2) & 0 \\ \left(\sqrt{2}/2,\sqrt{2}\right) & 1 \\ \left(\sqrt{2}/2,-\sqrt{2}\right) & -1 \\ \left(-\sqrt{2}/2,\sqrt{2}\right) & -1 \\ \left(-\sqrt{2}/2,-\sqrt{2}\right) & 1 \end{array}$$

So we see $f(x,y)=1$ is the maximum value at 1 at either $\left(\sqrt{2}/2,\sqrt{2}\right)$ or $\left(-\sqrt{2}/2,-\sqrt{2}\right)$.

A minimum value of $f(x,y)=-1$ is attained at $\left(\sqrt{2}/2,-\sqrt{2}\right)$ or $\left(-\sqrt{2}/2,\sqrt{2}\right)$.

Example 4 (harder): Find the volume of the largest rectangular box with faces parallel to the coordinate planes that can be inscribed in the ellipsoid $16x^2 + 4y^2 + 9z^2 = 144$.

Start by considering a sketch of the problem.



We wish to maximise

$$V = f(x, y, z) = 8xyz.$$

subject to the constraint

$$g(x, y, z) = 16x^2 + 4y^2 + 9z^2 - 144 = 0$$

- Consider

$$\nabla f(x,y,z) = \lambda \nabla g(x,y,z)$$

or

$$8yz\mathbf{i} + 8xz\mathbf{j} + 8xy\mathbf{k} = \lambda\left(32x\mathbf{i} + 8y\mathbf{j} + 18z\mathbf{k}\right).$$

- Together with $g(x,y,z) = 0$ gives the following system of 4 equations:

$$8yz = 32\lambda x,\ 8xz = 8\lambda y,\ 8xy = 18\lambda z,\ 16x^2 + 4y^2 + 9z^2 - 144 = 0.$$

- After some algebra we have $xyz = 12\lambda$

- The first equation gives

$$32\lambda\left(3 - x^2\right) = 0$$

so either $\lambda = 0$ or $x = \sqrt{3}$.

- Similarly, second gives

$$8\lambda\left(12-y^2\right)=0$$

hence $y=2\sqrt{3}$.

- Finally multiplying the third equation $8xy=18\lambda z$ by z

$$8xyz=18\lambda z^2$$

and using the values found earlier

$$8\left(12\lambda\right)=18\lambda z^2$$

which gives $z=4/\sqrt{3}$.

Hence we obtain the desired volume

$$V=8\left(\sqrt{3}\right)\left(2\sqrt{3}\right)\left(4/\sqrt{3}\right)=64\sqrt{3}\text{unit}^3.$$

Essential Linear Algebra

Given $\mathbf{x} \in \mathbb{R}^N; \mathbf{y} \in \mathbb{R}^N$. Define the inner (scalar) product:

$$\begin{aligned} \langle \mathbf{x}, \mathbf{y} \rangle &= \mathbf{x}^\mathsf{T} \mathbf{y} \\ &= \sum_{i=1}^{N} x_i y_i. \end{aligned}$$

Inner products and norms are closely related

$$\|\mathbf{x}\| = \sqrt{\mathbf{x}^\mathsf{T} \mathbf{x}}$$

$$\langle \mathbf{x}, \mathbf{y} \rangle = \|\mathbf{x}\| \, \|\mathbf{y}\| \cos\theta$$

This gives

$$\cos\theta = \frac{\mathbf{x}^\mathsf{T} \mathbf{y}}{\sqrt{\mathbf{x}^\mathsf{T} \mathbf{x} \mathbf{y}^\mathsf{T} \mathbf{y}}}$$

Linear Dependence/Independence

Set of vectors $\underline{v}_1,\ \underline{v}_2,\ \underline{v}_3, ..., \ \underline{v}_m \in V$

1. The set $\{\underline{v}_1,\ \underline{v}_2,\ \underline{v}_3, ..., \ \underline{v}_m\}$ is said to be linearly dependent (l.d) if $\exists$ scalars $\{\lambda_1, ..., \ \lambda_m\} \in \mathbb{F}$ not all zero s.t

$$\lambda_1\underline{v}_1 + \lambda_2\underline{v}_2 + ... + \lambda_m\underline{v}_m = \underline{0}$$

   The left hand side is called a LINEAR COMBINATION of $\underline{v}_1, ..., \ \underline{v}_m$.

2. If $\{\underline{v}_1, ..., \ \underline{v}_m\}$ are not l.d then they are linearly independent (l.i). In this case an equation such as

$$\lambda_1\underline{v}_1 + \lambda_2\underline{v}_2 + ... + \lambda_m\underline{v}_m = \underline{0}$$

$$\Rightarrow \lambda_1 = \lambda_2 = ... = \lambda_m = 0$$

Principal Component Analysis

PCA is a technique that takes a dataset $\mathbf{X}$ (consisting of a set of N tuples in $p-$dimensional space) and finds the directions along which the tuples line up best.

In short, a dimension reduction technique:

1. Taking linear combinations of the p original variables. Say $\mathbf{Z} = \mathbf{X}\dot{\alpha}_j$. Note: α_j is a vector.

2. Each of these linear combinations explains as much as possible of the variance in the data:

$$\max_{\alpha_j} V_j = \mathbb{V}\left(\mathbf{X}\alpha_j\right), \text{ with } V_j \geq V_k \text{ if } j < k.$$

3. Each linear combination is uncorrelated with all the others: $\alpha_j^\mathsf{T}\alpha_k = 0 \; \forall j \neq k$.

4. Each linear combination has unit length: $\alpha_j^\mathsf{T}\alpha_j = 1 \; \forall j$.

PCA Derivation

Denote the variance-covariance matrix of our dataset $\mathbf{X}$ as $\mathbf{\Sigma}$.

Let's focus on the first linear combination with maximum variance. In order to find this, we need to solve the following constrained optimization problem:

Take the first derivative and set to zero:

Note that we want to do maximisation

$$\max_{\alpha_1} \mathbb{V}\left(\mathbf{X}\alpha_1\right) \text{ s.t. } \alpha_1^{\mathsf{T}}\alpha_1 = 1$$

The Lagrangian is

$$\begin{aligned} L &= \left(\mathbf{X}\alpha_1\right)^{\mathsf{T}}\mathbf{X}\alpha_1 - \lambda_1\left(\alpha_1^{\mathsf{T}}\alpha_1 - 1\right) \\ &= \alpha_1^{\mathsf{T}}\mathbf{\Sigma}\alpha_1 - \lambda_1\left(\alpha_1^{\mathsf{T}}\alpha_1 - 1\right) \end{aligned}$$

Take first derivative and set to zero:

$$\frac{\partial L}{\partial \alpha_1} = \mathbf{\Sigma}\alpha_1 - \lambda_1\alpha_1 = 0 \Rightarrow \left(\mathbf{\Sigma} - \lambda_1 I_p\right)\alpha_1 = 0$$

Note we want to maximise $\alpha_1^\mathsf{T}\mathbf{\Sigma}\alpha_1 = \lambda_1\alpha_1^\mathsf{T}\alpha_1 = \lambda_1$.

Solution: $\lambda_1 =$ largest eigenvalue, $\alpha_1 =$ corresponding eigenvector.

To find the second linear combination, we need to take a further constraint into account

$$\max_{\alpha_2} \mathbb{V}\left(\mathbf{X}\alpha_2\right) \quad \text{s.t.} \quad \begin{array}{r} \alpha_2^\mathsf{T}\alpha_2 = 1 \\ \alpha_2^\mathsf{T}\alpha_1 = 0 \end{array}$$

The Lagrangian is

$$L = \alpha_2^\mathsf{T}\mathbf{\Sigma}\alpha_2 - \lambda_2\left(\alpha_2^\mathsf{T}\alpha_2 - 1\right) - \theta_2\alpha_2^\mathsf{T}\alpha_1$$

Optimising yields

$$\frac{\partial L}{\partial \alpha_2} = \Sigma\alpha_2 - \lambda_2\alpha_2 - \theta_2\alpha_1 = 0$$

Left-multiplying $\alpha_1^\top$: $\underbrace{\alpha_1^\top\Sigma\alpha_2}_{=0} - \underbrace{\lambda_2\alpha_1^\top\alpha_2}_{=0} - \theta_2\underbrace{\alpha_1^\top\alpha_1}_{=1} = 0$

Therefore,

$$(\Sigma - \lambda_2 I_p)\,\alpha_2 = 0$$

$\Rightarrow \lambda_2 =$ second largest eigenvalue, $\alpha_2 =$ corresponding eigenvector.

Maximum Likelihood Estimators (MLE)

- Suppose we draw $x_1,\ x_2, ..., x_n$ iid with density $p(\cdot;\theta)$.
- The probability of drawing a number in $[x_1, x_1 + dx]$ is $p(x_1;\theta)\, dx$
- Similarly the probability of drawing a number from x_2 to $x_2 + dx$ is $p(x_2;\theta)\, dx$ and so forth.
- The probability of all these events occurring is

$$p(x_1;\theta) \cdots p(x_n;\theta)\, dx^n = L(\theta; x_1, x_2, ..., x_n)\, dx^n$$

- The *likelihood function* is

$$L(\theta; x_1, x_2, ..., x_n) = L(\theta, \mathbf{x}),$$

which we think of as a function of θ.

- We ignore dx^n as this term is 'constant'.

- Maximum likelihood estimation works on the principle that the 'best' estimate of the parameters θ is given by maximising $L(\theta; \mathbf{x})$.

- As likelihoods are positive, and given independence are of the form

$$L(\theta, \mathbf{x}) = \prod_{i=1}^{n} p(x_i; \theta) = p(x_1;\theta) \cdots p(x_n;\theta)$$

it is often easier working with the log-likelihood

$$\ell(\theta, \mathbf{x}) = \log L(\theta, \mathbf{x}) = \sum_{i=1} \log p(x_i; \theta)$$

- If ℓ is differentiable in θ, then the maximum is when the Score function (derivative of ℓ) is zero,

$$S(\theta, \mathbf{x}) = \frac{d\ell}{d\theta} = 0.$$

- Usually, ℓ cannot have local minima or saddle points, so there is a maximum.

Suppose we pick n i.i.d random variables x_i such that $X_i \sim N\left(\mu, \sigma^2\right)$. Then the probability of drawing a number in the interval

$$x_i \text{ to } x_i + dx = p(x_i; \mu, \sigma) \quad \text{for} \quad i = 1, ..., n$$

where $p\left(x_i; \mu, \sigma^2\right)$ is the pdf for the normal distribution.

$$\begin{aligned} & L(x_1, x_2, ..., x_n; \mu, \sigma) \\ = \; & p(x_1; \mu, \sigma) ..... \times p(x_n; \mu, \sigma) (dx)^n . \end{aligned}$$

Recall, we ignore the term $(dx)^n$ as it is a constant. Hence

$$L\left(x_1, x_2, ..., x_n; \mu, \sigma\right) = \prod_{i=1}^{n} p\left(x_i; \mu, \sigma\right).$$

We now choose the parameters μ & σ in a way that maximises the likelihood of observing this. This is done by setting the **Score function** equal to zero

$$S\left(\mu, \mathbf{x}\right) = \frac{\partial \ell}{\partial \mu} = 0, \quad S\left(\sigma, \mathbf{x}\right) = \frac{\partial \ell}{\partial \sigma} = 0.$$

We have already seen (in the calculus course), in principle that these values can be local maxima/minima or saddle points.

However due to the nature of this problem, i.e. L is a likelihood function ensures that L can only have a maximum.

The problem can be simplified by noting that L attains a maximum iff $\ell = \log L$ has a maximum. In this case ℓ is called the *log-Likelihood function.*

So we have

$$\log L = \ell\left(x_1,\ x_2, ..., x_n;\ \mu,\ \sigma\right) = \sum_{i=1}^{n} \log p\left(x_i;\ \mu,\ \sigma\right)$$

and our problem becomes solving

$$\frac{\partial \ell}{\partial \mu} = 0, \quad \frac{\partial \ell}{\partial \sigma} = 0.$$

Example 1:

Suppose we draw n positive numbers X_i $(i = 1, ..., n)$ from i.i.d samples with distribution

$$f(x) = \begin{cases} 2\lambda^3 \sqrt{\frac{x}{\pi}} e^{-\lambda^2 x} & \text{for } x \geq 0 \\ 0 & \text{for } x < 0 \end{cases}$$

where the unknown parameter $\lambda > 0$.

How can we obtain a maximum Likelihood estimator $\widehat{\lambda}$ for λ?

The Likelihood function is

$$L\left(x_1,\ x_2, ..., x_n;\ \lambda\right) = \prod_{i=1}^{n} f\left(x_i; \lambda\right)$$

$$= \frac{2^n \lambda^{3n}}{\pi^{n/2}} \sqrt{(x_1 x_2 ..... x_n)} e^{-\lambda^2 (x_{1+} x_2 + ... + x_n)},$$

and for $\lambda \to 0$ & ∞, $L \to 0$, so there must be at least one maximum.

We use $\ell = \log L$ (much easier), as ℓ has maximum iff L has a maximum.

$$\begin{aligned} \ell &= \log \frac{2^n}{\pi^{n/2}} + 3n \log \lambda + \tfrac{1}{2} \log \left(x_1 x_2 ..... x_n\right) \\ &\quad -\lambda^2 \left(x_1 + x_2 + ... + x_n\right). \end{aligned}$$

So

$$\begin{aligned}\frac{\partial \ell}{\partial \lambda} &= \frac{3n}{\lambda} - 2\lambda\left(x_1 + x_2 + ... + x_n\right) = 0 \\ &\Rightarrow \; 2\widehat{\lambda}\left(x_1 + x_2 + ... + x_n\right) \\ &= \frac{3n}{\widehat{\lambda}}\end{aligned}$$

Hence

$$\widehat{\lambda}^2 = \frac{3n}{2\left(x_1 + x_2 + ... + x_n\right)}$$

and

$$\widehat{\lambda} = \sqrt{\frac{3n}{2\left(x_1 + x_2 + ... + x_n\right)}}.$$

The Maximum Likelihood estimator method is a popular method due to its simplicity when the form of the probability density function is known.

Example 2: Suppose $X_1, ..., X_n \sim N\left(\mu, \sigma^2\right)$ iid. What is the MLE of $\left(\mu, \sigma^2\right)$?

In this case,

$$p\left(x_i; \mu, \sigma^2\right) = \left(2\pi\sigma^2\right)^{-1/2} \exp\left(-\frac{\left(x_i - \mu\right)^2}{2\sigma^2}\right)$$

so

$$L\left(x_i; \mu, \sigma^2\right) = \left(2\pi\sigma^2\right)^{-n/2} \exp\left(-\sum_i \frac{\left(x_i - \mu\right)^2}{2\sigma^2}\right)$$

$$l\left(x_i; \mu, \sigma^2\right) = -\frac{n}{2} \log\left(2\pi\sigma^2\right) - \sum_i \frac{\left(x_i - \mu\right)^2}{2\sigma^2}$$

Note that for μ, this is just least squares.

Example 3: Suppose $X_1, ..., X_n \sim \exp(-\lambda x_i)$ iid. What is the MLE of λ?

In this case,

$$p(x_i; \lambda) = \lambda \exp(-\lambda x_i)$$

so

$$\begin{aligned} L(x_i; \lambda) &= \lambda^n \exp\left(-\lambda \sum_i x_i\right) \\ l(x_i; \lambda) &= n \log(\lambda) - \lambda \sum_i x_i \end{aligned}$$

hence the score function

$$S(x_i; \lambda) = \frac{n}{\lambda} - \sum_i x_i$$

which gives the MLE

$$\widehat{\lambda} = \frac{n}{\sum_i x_i} = \overline{x}^{-1}$$

Loss Functions

- We have an outcome measurement, which can be quantitative (such as stock price). Let's denote these **outcomes** as $\mathbf{y}$.

- We wish to predict the outcome based on a set of **features**. Let's denote these features as $\mathbf{x}$.

- We have a **training dataset**, where we observe the features and the outcome for a set of objects.

- Using the training data we build a **prediction model**, or **learner**, which we denote as $f(x)$. The model is a (possibly highly non-linear) transformation of features into outcomes:

$$y = f(x) + \varepsilon$$

f(x) – true fn. to approximate

Other types of notation are

$$\begin{aligned} y &= f_\beta(\mathbf{x}) = f(\mathbf{x};\boldsymbol{\beta}) = \boldsymbol{\beta}^{\mathsf{T}}\mathbf{x} \\ &= (\beta_1,\ldots,\beta_n)\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \sum_{i=1}^{N}\beta_i x_i. \end{aligned}$$

input– Calculus $x \in \mathbb{R}$. Vector Calculus: $\mathbf{x} \in \mathbb{R}^D$. ML: can also work for discrete inputs, strings, tree, graphs, ...

output– Classification: $y \in \{0,1\}$. Regression: $y \in \mathbb{R}$

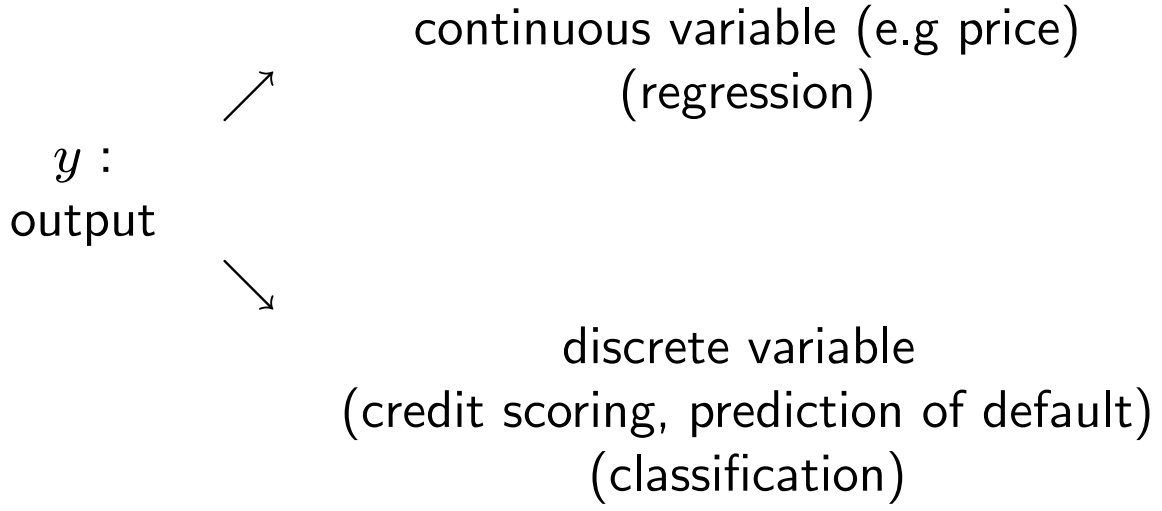
methods– Linear classifiers, neural networks, decision trees, ensemble models,..

probabilistic classifiers, . . .

β– parameters/weights: $\beta \in \mathbb{R}$. $\boldsymbol{\beta} \in \mathbb{R}^D$

To set the scene: Here's a typical scenario of interest

Outcome y (this is the output) which is what we wish to predict



$\mathbf{x}$ (set of features): input. So $\mathbf{x}$ can be a scalar or vector.

p : number of features - size of vector, i.e. $\mathbf{x} \in \mathbb{R}^p$.

Set of input and output pairs $\mathcal{S} = \left\{ \left(x^{(i)}, y^{(i)} \right) : i = 1, \ldots N \right\}$;

data set for training

$x^{(i)} \in \mathbb{R}$, $y^{(i)} \in \mathbb{R}$ we want to estimate the parameter (weights) vector $\beta = (\beta_1, \ldots, \beta_n)$ of unknown parameters.

$$\left(x^{(i)}, y^{(i)}\right) : i = 1, \ldots N \text{ pairs/2-tuples}$$

GOAL: Predict the relationship between the input and output.

So how is $x \longrightarrow y$ achieved?

So output is some (unknown) function of the input, together with some noise. Let's write

$$y = f(x) + \varepsilon$$

The estimate for f, which is sometimes denoted as $\widehat{f}$.

$\mathbb{E}[\varepsilon] = 0 \quad \mathbb{V}[\varepsilon] = \sigma^2$

$\mathbb{E}[y] = \mathbb{E}[f + \varepsilon] = \mathbb{E}[f] + \mathbb{E}[\varepsilon] = f$

The Loss Function L

Here we define the *loss function*

$$L\left(y, f\left(x\right)\right)$$

as a measure of how far our prediction is from the outcome. A popular definition is

$$L\left(y, f\left(x\right)\right) = \left(y - f\left(x\right)\right)^2$$

$y-$ actual outcome. $f\left(x\right) -$ predicted outcome.

We want to minimise L so that the distance between the two outcomes is the smallest value possible.

Loss functions are used in ML to estimate how badly models are performing - a measure of how wrong the model is in terms of its ability to estimate the relationship between input and output.

Look for $f(x)$ which minimises this 'error function' L, which can be written as

$$\underset{f(x)}{\arg\min} \left\{ \mathbb{E}\left[L\left(y, f\left(x \right) \right) \right] \right\}$$

or in component form

$$\underset{f(x)}{\arg\min} \left\{ \frac{1}{N} \sum_{i=1}^{N} \left(y^{(i)} - f\left(x^{(i)} \right) \right)^2 \right\}$$

MSE for the training set is

$$\frac{1}{N} \sum_{i=1}^{N} \left(y^{(i)} - f\left(x^{(i)} \right) \right)^2$$

In OR a classic optimisation problem is to maximise/minimise an objective function subject to a system of constraints The function we want to minimize or maximize is called the objective function. However when we are minimizing it, we may also call it the **cost function**, **loss function**, or error function.

• suppose that X is a $p-$dimensional real-valued random input vector and Y is a real valued random output variable.

• The (unknown) joint distribution of X and Y is $P(X, Y)$.

• We seek to find a function $f(x)$ for predicting Y given X.

• To find the "optimal" $f(x)$, we need an objective (or loss) function, which we assume to be quadratic here

$$L(Y, f(X)) = (Y - f(X))^2$$

The expected loss function

$$\mathbb{E}[L] = \int_{\mathbb{R}} \int_{\mathbb{R}} (y - f(x))^2 P(x, y)\, dxdy$$

$$\begin{aligned}(y - f(x))^2 &= \{y - \mathbb{E}[y|x] + \mathbb{E}[y|x] - f(x)\}^2 \\ &= \{y - \mathbb{E}[y|x]\}^2 + 2\{y - \mathbb{E}[y|x]\}\{\mathbb{E}[y|x] - f\} + \{y - f\}^2\end{aligned}$$

① ② $= 0$ ③

$\mathbb{E}[①] = \mathbb{E}[(y - \mu_y)^2] = \mathbb{V}[y]$

$\mathbb{E}[②] = 0 \quad \because \mathbb{E}[y - \mathbb{E}[y]] = \mathbb{E}[y] - \mathbb{E}[\mu_y] = \mu_y - \mu_y = 0$

$\mathbb{E}[③] = \mathbb{E}[\underbrace{(\mathbb{E}[y] - \hat{f})^2}_{B(y,\hat{f})}] = \text{bias}^2$

$\left(B(\theta) = \mathbb{E}[\hat{\theta}] - \theta \right)$

MSE of estimator = var. of est. + bias^2

Bias-Variance decomposition

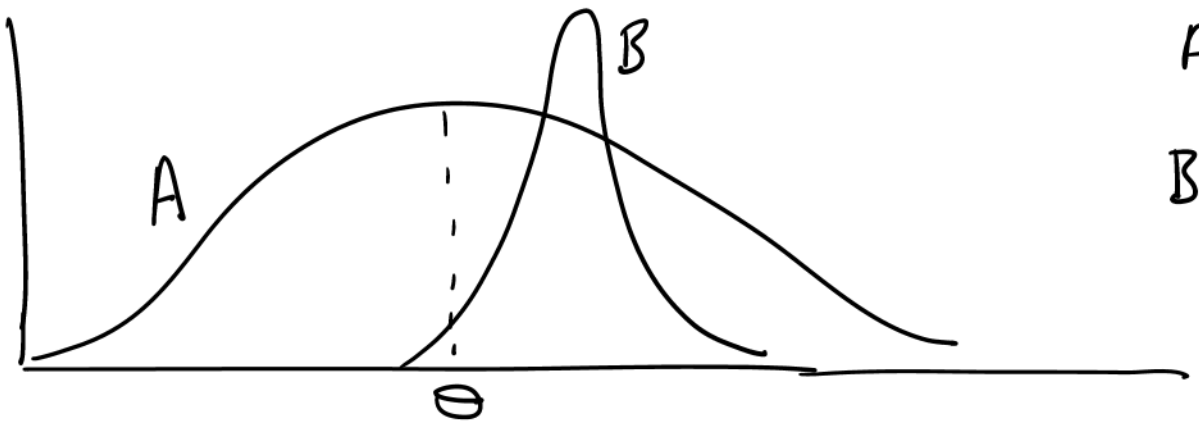
There is a trade-off between bias and variance, with very flexible models having low bias and high variance, and relatively rigid models having high bias and low variance. The model with the optimal predictive capability is the one that leads to the best balance between bias and variance.

$y = f + \varepsilon$

Writing $y - \hat{f} = f + \varepsilon - \hat{f}$

$\mathbb{E}[(f + \varepsilon - \hat{f})] \longrightarrow \mathbb{E}[\varepsilon^2] = \sigma^2$ i.e $\mathbb{V}[\text{noise}]$

$\mathbb{E}[L] = \mathbb{V}(\hat{f}) + \sigma^2 + \text{Bias}^2[\hat{f}]$



A: unbiased but larger var.

B: smaller var