An Introduction to Data Science

Bahram Mobasher

March 2020

Contents

1 Mathematical Background:

1.1 Vectors

A vector is a finite list of numbers, equally written as vertical arrays surrounded by squares or curved brackets, as in:

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{n-1} \\ v_n \end{pmatrix} \tag{1}$$

They can also be written as numbers separated by commas and surrounded by parenthesis, as in:

$$\mathbf{v} = (v_1, v_2, \ldots, v_{n-1}, v_n) \tag{2}$$

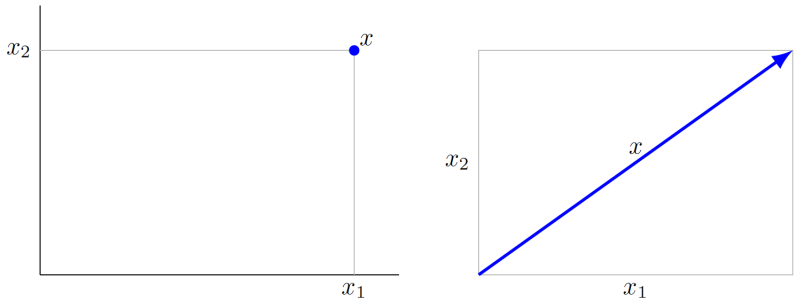


Figure 1: Left. The 2-vector x specifies the position (shown as a dot) with coordinates x_1 and x_2 on a plane. Right. The 2-vector x represents a displacement in the plane (shown as an arrow) by x_1 in the horizontal axis and x_2 on the vertical.

The elements (or entries, coefficients, components) of a vector are the values in the array. The size (dimension or length) of a vector is the number of the elements it contains. A vector of size n is called an n-vector. A 1-vector (vector of size one) is a number. There is no distinction between the 1-vector, eg. [1.3], and the number 1.3.

We often use symbols to define vectors. If we denote an n-vector using the symbol $\mathbf{a}$, the i^{th} element of the vector $\mathbf{a}$ is a_i where the subscript i is an integer index that runs from 1 to n, the size of the vector (Fig 1).

Two vectors $\mathbf{a}$ and $\mathbf{b}$ are equal, which we denote as $\mathbf{a} = \mathbf{b}$, if they have the same size with each of the entries being the same. For example, if $\mathbf{a}$ and $\mathbf{b}$ are n-vectors, then $\mathbf{a} = \mathbf{b}$ means $a_1 = b_1, \ldots, a_n = b_n$.

The numbers or values of the elements in a vector are called scalars. The set of all real numbers is written as R, and the set of all real n-vectors is denoted $\Re^n$. The notation $\mathbf{a} \in \Re^{\mathbf{n}}$ means that $\mathbf{a}$ is an n-vector with real entries. This means that $\mathbf{a}$ is an element of the set $\Re^n$.

One could define vectors by concatenating or stacking two or more vectors, as in:

Where $\mathbf{a}, \mathbf{b}, \mathbf{c}$ and $\mathbf{d}$ are vectors. If $\mathbf{b}$ is an m-vector, $\mathbf{c}$ is an n-vector and $\mathbf{d}$ is a p-vector, this defines the $(m+n+p)$ vector

$$\mathbf{a} = (b_1, \ldots, b_m, c_1, \ldots, c_n, d_1, \ldots, d_p) \tag{3}$$

The stacked vector $\mathbf{a}$ is then written as $\mathbf{a} = (\mathbf{b}, \mathbf{c}, \mathbf{d})$. In this example, one could say that $\mathbf{b}, \mathbf{c}$ and $\mathbf{d}$ are sub-vectors or slices of $\mathbf{a}$, with sizes m, n and p. Colon notations are used to denote sub-vectors. If $\mathbf{a}$ is a vector, then $\mathbf{a_{r:s}}$ is the vector of size $s-r+1$ with entries $a_r, \ldots, a_s$.

$$\mathbf{a_{r:s}} = (a_r, \ldots, a_s) \tag{4}$$

For example, if $\mathbf{z}$ is the 4-vector $(1, -1, 2, 0)$, the slice $\mathbf{z}$ 2:3 is $z_{2:3} = (-1, 2)$.

Unit Vectors: A unit vector is a vector with all its elements equal to zero except one element which is equal to one. The i^{th}

unit vector of size n is the unit vector with its i^{th} element being one and is denoted as e_i. the following are unit vectors of size 3. For $i, j = 1, \ldots, n$. On the left hand side, e_i is an n-vector; $(e_i)_j$ is a number, its j^{th} entry.

$$(e_i)_j = 1 \;\; if \;\; j = i; \;\; (e_i)_j = 0 \;\; if \;\; j \neq i \tag{5}$$

$$e_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{6}$$

$$e_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \tag{7}$$

$$e_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{8}$$

Here e_i denotes the i^{th} unit vector and not the i^{th} element of a vector $\mathbf{e}$.

Sparsity: A vector is said to be sparse if many of its entries are zero. Its sparsity pattern is the set of indices of non-zero entries. Unit vectors are sparse since they have only one non-zero entry.

The zero vector is the sparsest possible vector, since it has no non-zero entries.

Examples:

An n-vector can be used in many applications representing n quantities or values. Some typical examples include:

- **Location and displacement:** A 2-vector can be used to represent a location or position in a 2-dimensional (2-D) space (Figure 1). A 3-vector is used to represent the location or position of some point in a 3-dimensional (3-D) space. The entries of the vector give the coordinates of the position or location.

  A vector can also be used to represent a displacement in a plane or 3-D space. In this case, it is typically drawn as an arrow (Figure 1). A vector can represent the velocity or acceleration at a given time of a given point.

- **Color:** a 3-vector can represent a color, with its entries giving the Red, Green and Blue (RGB) intensity values (often between 0 and 1). The vector (0,0,0) represents black, the vector $(0, 1, 0)$ represents a bright pure green color and the vector $(1, 0.5, 0.5)$ represents shade of pink (Figure 2).

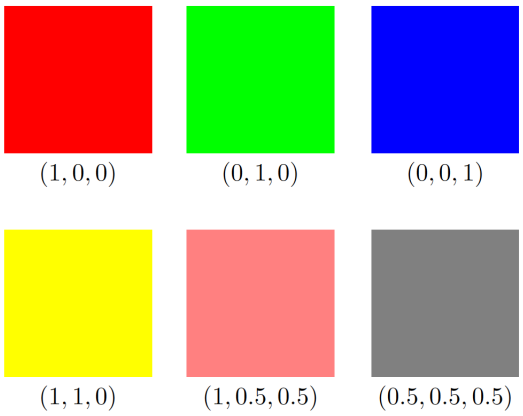


Figure 2: Six colors and their RGB vectors.

- **Portfolio:** an n-vector s can represent a stock portfolio or investment in n different assets, with s_i giving the number of shares of asset i held. The vector (100,50,20) represents a portfolio consisting of 100 shares of asset 1, 50 shares of asset 2, and 20 shares of asset 3. Shares that you owe another party are represented by negative entries in a portfolio vector.

- **Values across a population:** An n-vector can give the values of some quantity across a population of individuals or entries. For example, an n-vector b can give the blood pressure of a collection of n patients, with b_i the blood pressure of patient i, for $i = 1, \ldots, n$.

- **Time series:** An n-vector can represent a time series or signal, that is, the value of some quantity at different times. For example, an audio signal can be represented as a vector

whose entries give the value of acoustic pressure at equally spaced times (typically 48000 or 44100 per second). A vector may give the hourly rainfall (or temperature or pressure) at some location over some time period. When a vector represents a time series, it is similar to plot x_i vs. i with lines connecting consecutive time series values. An example is shown in figure 3 where the 48-vector x give the hourly temperature.

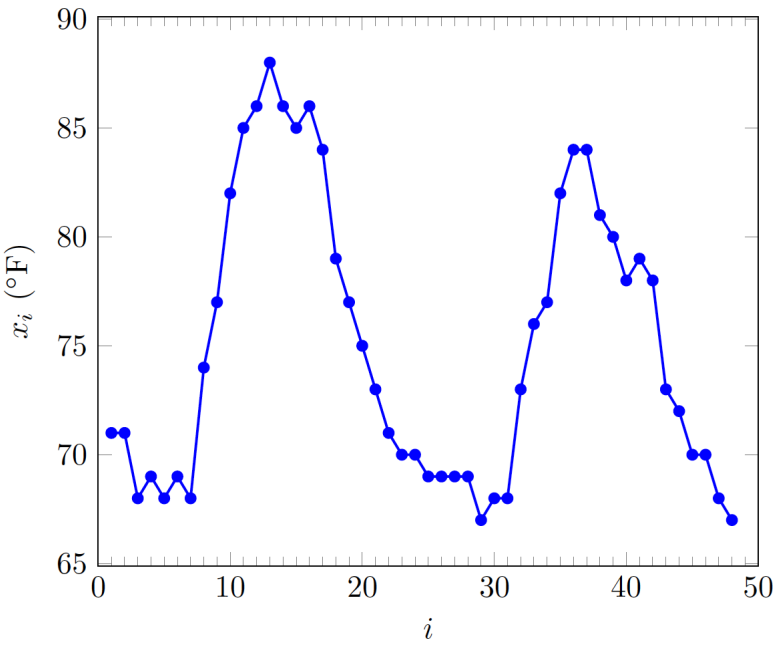


Figure 3: Hourly temperature in downtown Los Angeles on August 5 and 6, 2015 (starting at 12:47AM, ending at 11:47PM).

- **Daily return:** a vector can represent daily return of a stock- its fractional increase or decrease in value. For exam-

ple, the return time series vector $(-0.022, +0.014, +0.004)$ means the stock price went down 2.2% the first day, went up 1.4% the second day and up again by 0.4% the third day. A vector could represent daily or minute-by-minute change in the stock.

- **Images:** A monochrome (black and white) image is an array of $M \times N$ pixels with M rows and N columns (Figure 4). Each of the MN pixels has a grey scale or intensity value with 0 corresponding to black and 1 corresponding to bright white. An image can be represented by a vector of length MN with the elements giving grey scale levels at the pixel locations, ordered column-wise or row-wise.

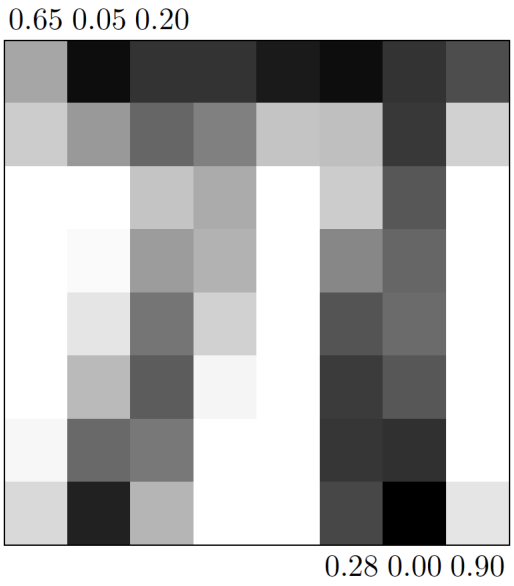


Figure 4: 8×8 image and the grayscale levels at six pixels.

Figure 4 shows an example of an 8×8 image with the vector entries arranged row-wise, the associated 64-vector is

$$X = (0.65, 0.05, 0.20, \ldots, 0.28, 0.00, 0.9)$$

A color $M \times N$ pixel image is described by a vector of length $3MN$, with the entries giving the R, G and B values for each pixel.

1.1.1 Vector addition

Two vectors of the same size could be added together by adding the corresponding elements to form another vector of the same size, called the sum of the vectors.

Properties: The following are properties of vector addition (subtraction). For any vector a, b and c of the same size, we have

- Vector addition is commutative: $a + b = b + a$
- Vector addition is associative: $(a + b) + c = a + (b + c)$
- Adding the zero vector to a vector has no effect: $a + 0 = 0 + a = a$
- Subtracting a vector from itself yields zero: $a–a = 0$

Displacements: When vectors **a** and **b** represent displacements,

the sum $\mathbf{a} + \mathbf{b}$ is the net displacement found by first displacing by $\mathbf{a}$, then displacing by $\mathbf{b}$ (or visa versa), as shown in figure 5. If the vector $\mathbf{p}$ represents a position and vector $\mathbf{a}$ represents a displacement, then $\mathbf{p} + \mathbf{a}$ is the position of the point $\mathbf{p}$, displaced by a (Figure 6).

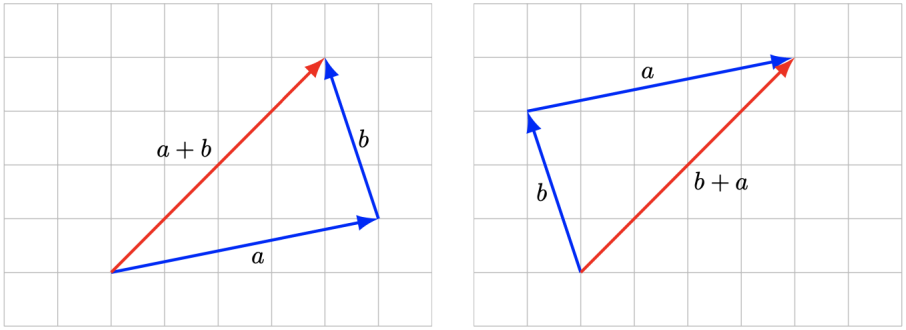


Figure 5: *Left.* The lower blue arrow shows the displacement a; the displacement b, shown as the shorter blue arrow, starts from the head of the displacement a and ends at the sum displacement $a + b$, shown as the red arrow. *Right.* The displacement $b + a$.

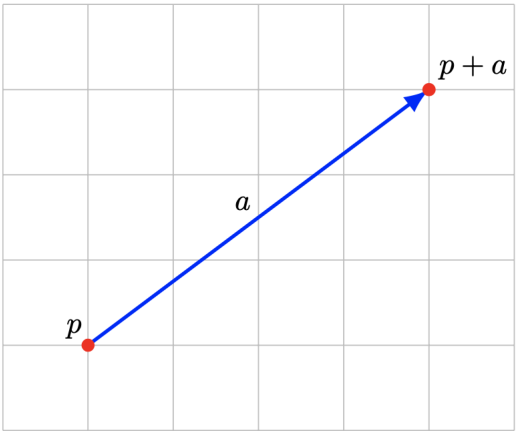


Figure 6: The vector $p + a$ is the position of the point represented by p displaced by the displacement represented by a.

Displacement between two points: If vectors $\mathbf{p}$ and $\mathbf{q}$ represent the positions of two points in 2-D or 3-D space, then $\mathbf{p} - \mathbf{q}$ is the displacement vector from q to p (Figure 7).

Question 1.1: Periodic energy usage

The 168-vector $\mathbf{w}$ gives the hourly electricity consumption of a manufacturing plant, starting on Sunday midnight at 1 AM, over one week, in MWh (megawatt-hours). The consumption pattern is the same each day, i.e., it is 24-periodic, which means that $w_t + 24 = w_t$ for t = 1,...,144. Let d be the 24-vector that gives the energy consumption over one day, starting at midnight.

a Use vector notation to express $\mathbf{w}$ in terms of d.

b Use vector notation to express d in terms of $\mathbf{w}$.

Solution:

a Hourly electricity consumption of a manufacturing plant from midnight 12:00 Sunday a.m. to 1:00 a.m. Monday is given by 24-vector d. It is the same for all other days (12:00 a.m. Monday to 1:00 a.m. Tuesday and ...), so the weekly consumption can be expressed as: $\mathbf{w} = [\mathbf{d}\ \mathbf{d}\ \mathbf{d}\ \mathbf{d}\ \mathbf{d}\ \mathbf{d}\ \mathbf{d}]$.

b For 24 periodic, $d = w_{1:24} = w_{25:48} = ... = w_{145:168}$.

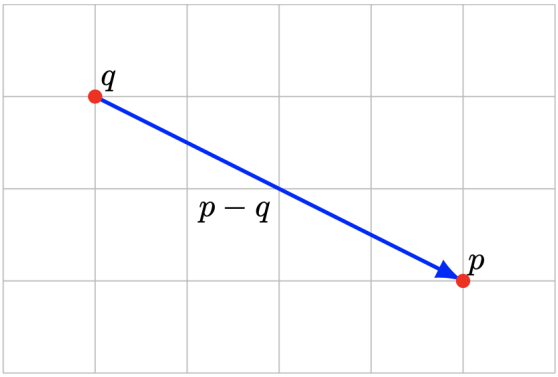


Figure 7: The vector $\mathbf{p} - \mathbf{q}$ represents the displacement from the point represented by q to the point represented by p.

1.1.2 Scalar-vector multiplication

In scalar multiplication a vector is multiplied by a scalar (i.e. a number), which is done by multiplying every element of the vector by the scalar.

Scalar vector multiplication follows the associative law. If $\mathbf{a}$ is a vector and b and c are scalars, we have $(bc)\mathbf{a} = \mathbf{b}(\mathbf{ca})$ on the left side we have a scalar-scalar multiplication (bc) followed by scalar vector multiplication while on the right side we see two vector multiplications. If α is a vector and β and γ are scalars, then $(\beta + \gamma)\alpha = \beta\alpha + \gamma\alpha$ This is the distributive property of scalar-vector multiplication. Like ordinary multiplication, scalar multiplication has a higher precedence in equations than vector addition.

Displacements:
When vector $\mathbf{a}$ represents a displacement, and $\beta > 0$, $\beta\mathbf{a}$ is a displacement in the same direction of $\mathbf{a}$, with its magnitude scaled by β. When $\beta < 0$, $\beta\mathbf{a}$ represents a displacement in the opposite direction of vector a with magnitude scaled by $|\beta|$ (Figure 8).

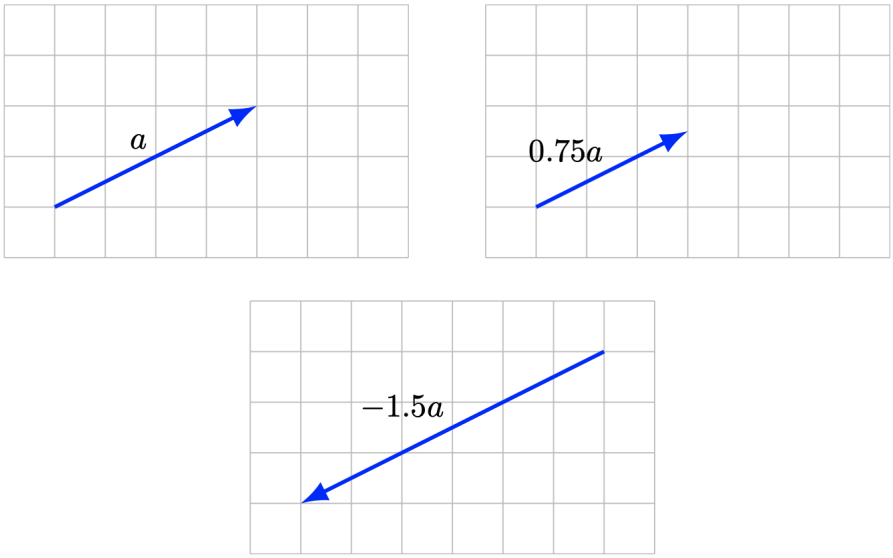


Figure 8: The vector $0.75a$ represents the displacement in the direction of the displacement a, with magnitude scaled by 0.75; $(-1.5)a$ represents the displacement in the opposite direction, with magnitude scaled by 1.5.

Linear combinations:
If $\mathbf{a_1}, \ldots, \mathbf{a_n}$ are n-vectors, and $\beta_1, \ldots, \beta_n$ are scalars, the n-vector $\beta_1 a_1 + \ldots + \beta_n a_n$ is the linear combination of vectors $\mathbf{a_1}, \ldots, \mathbf{a_n}$. The scalars $\beta_1, \ldots, \beta_n$ are called the coefficients of the linear combination.

Linear combination of unit vectors:
We can write any $n-$vector $\mathbf{B}$ as a linear combination of the standard unit vectors as $B = b_1 e_1 + \ldots + b_n e_n$ In this equation b_i is the i^{th} entry of $\mathbf{B}$ (a scalar) and e_i is the i^{th} unit vector.

A specific example is when the vectors represent displacements, a linear combination is the sum of the scaled displacements (Figure 9).

Question 1.2: Linear combinations of linear combinations

Suppose that each of the vectors $\mathbf{b_1}, ..., \mathbf{b_k}$ is a linear combination of the vectors $\mathbf{a_1}, ..., \mathbf{a_m}$ and $\mathbf{c}$ is a linear combination of $\mathbf{b_1}, ..., \mathbf{b_k}$. Then $\mathbf{c}$ is a linear combination of $\mathbf{a_1}, ..., \mathbf{a_m}$. Show this for the case with $m = k = 2$. (Showing it in general is not much more difficult, but the notation gets more complicated.)

Solution:

b_1, b_2 are linear combination of a_1, a_2.

$$b_1 = \alpha_1 a_1 + \alpha_2 a_2$$

$$b_2 = \beta_1 a_1 + \beta_2 a_2$$

where $\alpha_1, \alpha_2, \beta_1$ and β_2 are scalars. Also, c is a linear combination of $b_1, ..., b_m$. So,

$$c = \gamma_1 b_1 + \gamma_2 b_2$$

where γ_1 and γ_2 are scalars.

$$c = \gamma_1(\alpha_1 a_1 + \alpha_2 a_2) + \gamma_2(\beta_1 a_1 + \beta_2 a_2) = \delta_1 a_1 + \delta_2 a_2$$

.

Thus, $\delta_1 = \gamma_1\alpha_1 + \gamma_2\beta_1$ and $\delta_2 = \gamma_1\alpha_2 + \gamma_2\beta_2$. This shows

that c is a linear combination of a_1, a_2.

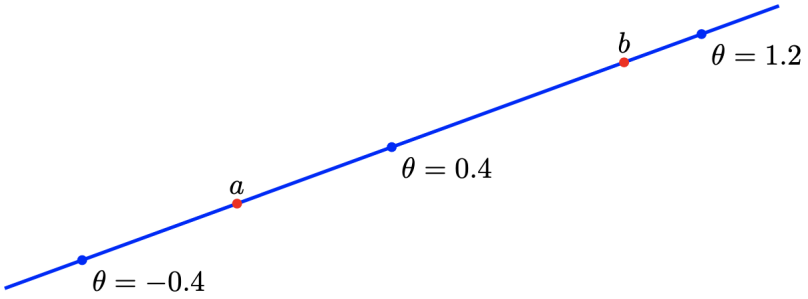


Figure 9: The affine combination $(1-\theta)a+\theta b$ for different values of θ. These points are on the line passing through a and b; for θ between 0 and 1, the points are on the line segment between a and b.

Examples:

Audio mixing: when $\mathbf{a_1}, \ldots, \mathbf{a_m}$ are vectors representing audio signals (over the same period of time, simultaneously recorded), they are called tracks. The linear combination $\beta_1 a_1 + \ldots + \beta_m a_m$ is perceived as a mixture of the studio tracks with relative loudnesses given by $|\beta_1|, \ldots, |\beta_m|$. In a studio the values of $\beta_1, \ldots, \beta_m$ are chosen to give a good balance between the different instruments (eg. vocals and drums).

1.1.3 Inner product:

The inner product (also called dot product) of two $n-$vectors is defined as the scalar

$$a^T b = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n \tag{9}$$

the sum of the products of corresponding entries. The superscript T denotes the transpose of a matrix. Notations of the inner product are any of the following: $\langle a, b \rangle$, $\langle a|b \rangle$, (a, b) and $(a.b)$. As an example of inner product we have:

When $n = 1$, the inner product reduces to the usual product of two scalars (numbers).

Properties: If **a, b** and **c** are vectors of the same size and γ is a scalar, we have:

- **Commutativity:** $a^T b = b^T a$. the order of the two vectors in the inner product does not matter.
- **Associativity with scalar multiplication:** $(\gamma a)^T b = \gamma(a^T b)$. Therefore, we can write both as $\gamma a^T b$.
- **Distributivity with vector addition:** $(a + b)^T c = a^T c + b^T c$. The inner product can be distributed across vector addition.

As another example, we could write

$$(a + b)^T (c + d) = a^T c + a^T d + b^T c + b^T d \tag{10}$$

the inner product of two vectors (in the left) can be expressed as the sum of four inner products (in the right).

Properties:

- **Unit vector:** $e_i^T a_i$. The inner product of a vector with the i^{th} standard unit vector gives the i^{th} element of a (a_i).
- **Sum:** $e_i^T a = a_1 + a_2 + \ldots + a_n$. the inner product of a vector with the vector of ones gives the sum of the elements of the vector.
- **Sum of squares:** $a^T a = a_1^2 + \ldots + a_n^2$. The inner product of a vector with itself gives the sum of the squares of the elements of the vector.
- **Average:** The inner product of an $n-$vector with the vector $1/n$ gives the average of the elements of the vectors. The average of the entries of a vector is denoted by $avg(x)$.

Applications of inner product:

- **Co-occurrence:** If $\mathbf{a}$ and $\mathbf{b}$ are $n-$vectors with each of their elements either 0 or 1 (so-called occurrence), then $a^T b$ gives the total number of indices for which a_i and b_i are both one. If vectors $\mathbf{a}$ and $\mathbf{b}$ are subsets of n objects, then $a^T b$ gives the number of objects in the intersection of two subsets. Figure 10 shows two subsets A and B of 7 objects labeled $1, \ldots, 7$, with corresponding vectors $\mathbf{a} = (\mathbf{0}, \mathbf{1}, \mathbf{1}, \mathbf{1}, \mathbf{1}, \mathbf{1}, \mathbf{1})$ and $\mathbf{b} = (\mathbf{1}, \mathbf{0}, \mathbf{1}, \mathbf{0}, \mathbf{1}, \mathbf{0}, \mathbf{0})$ Here we have $a^T b = 2$ which is

the number of objects in both A and B (objects 3 and 5).

- **Weights and features:** When the vector **f** represents a set of features of an object, and **w** is a vector of the same size (called weight vector), the inner product $w^T f$ is the sum of the feature values, scaled (or weighted) by the weights. For example, if the feature is associated with a loan application (eg. age, income . . .), we interpret $s = w^T f$ as a credit score. $\mathbf{W_i}$ is the weight given to feature i^{th} to form the score.

- **Probability and expected values:** Suppose the n-vector **p** has non-negative entries that sum to one- describing a set of proportions among n items, or a set of probabilities of n outcomes, one of which must occur. Suppose **f** is another n-vector, where f_i is the value of some quantity if outcome i occurs. Then $f^T p$ gives the expected value or the mean of the quantity, with the probabilities given by p.

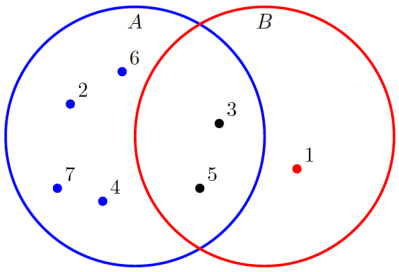


Figure 10: Two sets A and B, containing seven objects.

Question 1.3: Profit and sales vectors

A company sells n different products or items. The $n-$vector

$\mathbf{p}$ gives the profit, in dollars per unit, for each of the n items. (The entries of $\mathbf{p}$ are typically positive, but a few items might have negative entries. These items are called loss leaders, and are used to increase customer engagement in the hope that the customer will make other, profitable purchases. The $n-$vector $\mathbf{s}$ gives the total sales of each of the items, over some period (such as a month), i.e., s_i is the total number of units of item i sold. (These are also typically non-negative, but negative entries can be used to reflect items that were purchased in a previous time period and returned in this one.) Express the total profit in terms of p and s using vector notation.

Solution:

The profit vector is $\mathbf{p} = (\mathbf{p_1}\ \mathbf{p_2}\ ...\ \mathbf{p_n})$, where p_i is the profit per unit of i^{th} item. Also, the sale vector is $\mathbf{s} = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{pmatrix}$, where s_i is the number pf units of sale of i^{th} item. Total profit can be found as $p_1s_1 + p_2s_2 + ... + p_ns_n$. It can be expressed in a vector notation: total profit=$(p_1\ p_2\ ...\ p_n)$.

$$\begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{pmatrix} = p.s$$

.

1.2 Linear Functions

Function notations

The notation $f : \Re^n \to \Re$ means that f is a function that maps real n-vectors to real numbers, i.e. it is a scalar valued function of n-vectors. If $\mathbf{x}$ is an n-vector, then $f(x)$, which is a scalar, denotes the value of the function f at x. In the notation $f(x)$, x is referred as an argument of the function. We can also refer to f as the function of n scalar arguments, the entries of the vector argument, in which case we write

$$f(x) = f(x_1, x_2, \ldots, x_n) \tag{11}$$

here we refer to $x_1, \ldots, x_n$ as the arguments of f. A more specific example is given below where the function is not given as an equation or formula. Suppose $f : \Re^3 \to \Re$ is the function that gives the trajectory of a projectile. As a function of 3-vectors, x_1 is the initial speed, x_2 is the drag force and x_3 is the angle of the projectile with respect to the horizon.

The inner product function: Suppose **a** is an n-vector. We can define a scalar-valued function f of $n-$vectors given by:

$$f(x) = a^T x = a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \tag{12}$$

for any n-vector **x**. This function gives the inner product of its n-vector argument x with some n-vector **a**. In other words, the elements of **a** give the weights used in forming the weighted sum.

Superposition and linearity: The inner product function defined in equation 12 satisfies the property

$$\begin{aligned} f(\alpha x + \beta y) &= a^T(\alpha x + \beta y) = a^T(\alpha x) + a^T(\beta y) \\ &= \alpha(a^T x) + \beta(a^T y) = \alpha f(x) + \beta f(y) \end{aligned} \tag{13}$$

for all n-vectors **x** and **y** and all scalars α and β. This property is called *superposition*. A function that satisfies superposition is called *linear*. The above proves that the inner product with a fixed vector is a linear function.

In the left-hand side of this equation, the term $\alpha x + \beta y$ involves scalar-vector multiplication followed by vector addition. However, the right-hand $\alpha f(x) + \beta f(y)$ involves scalar multiplication and scalar addition.

If a function f is linear, superposition extends to linear combination of any number of vectors and not just linear combinations of two vectors. We have

$$F(\alpha_1 x_1 + \ldots + \alpha_k x_k) = \alpha_1 f(x_1) + \ldots + \alpha_k f(x_k) \tag{14}$$

For any n-vectors $x_1, \ldots, x_k$ and any scalars $\alpha_1, \ldots, \alpha_k$, The function $f : \Re^n \to \Re$ is linear if it satisfies the following properties:

- **Homogeneity:** For any n-vector $\mathbf{x}$ and any scalar α, $f(\alpha x) = \alpha f(x)$
- **Additivity:** For any n-vectors $\mathbf{x}$ and $\mathbf{y}$, $f(x+y) = f(x)+f(y)$

Inner product representation of a linear function: If a function is linear, then it can be expressed as the inner product of its argument with some fixed vector.

Suppose f is a scalar-valued function of $n-$vectors, and is linear. Then there is an $n-$vector $\mathbf{a}$ such that $f(x) = a^T x$ for all $\mathbf{x}$. We call $a^T x$ the inner product representation of f. To show this, we use the unit vector e_i to express an arbitrary $n-$vector $\mathbf{x}$ as $x = x_1 e_1 + \ldots + x_n e_n$. If f is linear, by superposition we have

$$f(x) = f(x_1 e_1 + \ldots + x_n e_n) = x_1 f(e_1) + \ldots + x_n f(e_n) = a^T x$$

with $a = (f(e_1), \ldots, f(e_n))$. This becomes

$$f(x) = x_1 f(e_1) + x_2 f(e_2) + \ldots + x_n f(e_n)$$

which holds for any linear scalar-valued function f.

Question 1.4: Linear Functions?

The function $\phi : \Re^3 \to \Re$ satisfies $\phi(1, 1, 0) = -1$, $\phi(-1, 1, 1) = 1$, and $\phi(1, -1, -1) = 1$. Choose one of the following, and justify your choice: ϕ must be linear; ϕ could be linear; ϕ cannot

be linear.

Solution:

If possible let ϕ be linear, then:

$$\phi(1,-1,-1) = \phi(-1(-1,1,1)$$

$$\phi(1,-1,-1) = -\phi(-1,1,1)$$

which gives $1 = -1$. It is a contradiction. Thus, ϕ cannot be linear.

1.3 Taylor approximation

In many applications, scalar-valued functions of n variables can be approximated as linear functions. Suppose $f : \Re^n \to \Re$ is differentiable, which means that its partial derivatives exist. Let z be an $n-$vector. The first-order Taylor approximation of f near (or at) the point z is the function $\tilde{f}(x)$ of x defined as

$$\tilde{f}(x) = f(z) + \frac{\partial f}{\partial x_1}(z)(x_1 - z_1) + \cdots + \frac{\partial f}{\partial x_n}(z)(x_n - z_n)$$

where $\frac{\partial f}{\partial x_i}(z)$ denotes the partial derivative of f with respect to its i^{th} argument, evaluated at the $n-$vector z. $\tilde{f}(x)$ on the left side depicts that it is an approximation of the function f.

The first-order Taylor approximation $\tilde{f}(x)$ is a good approximation

of $f(x)$ when all x_i, are near the associated z_i. The first-term in the Taylor expansion is a constant. The other terms can be interpreted as the contributions to the change in the function value (from $f(z)$) due to the changes in the components of x from z. $\tilde{f}$ is a linear approximation of f near z. This can be written in the compact notation as

$$\tilde{f}(x) = f(z) + \nabla f(z)^T (x - z)$$

where $\nabla f(z)$ is an n-vector, the gradient of f at the point z

$$\nabla f(z) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(z) \\ \vdots \\ \frac{\partial f}{\partial x_n}(z) \end{pmatrix}$$

The first term in the Taylor expansion is the constant $f(z)$, the value of the function when $x = z$. The second term is the inner product of the gradient of f at z and the deviation or perturbation of x from z - i.e. $x - z$.

Taylor approximation gives us a way to construct an approximation of a function $f : \Re^n \to \Re$, near a given point z, when there is a formula that describes f, and it is differentiable.

An example, for $n = 1$, is shown in figure 11 where over the full x-axis, the Taylor approximation does not give a good approximation of the function f. But for x near z, the Taylor approximation is very good.

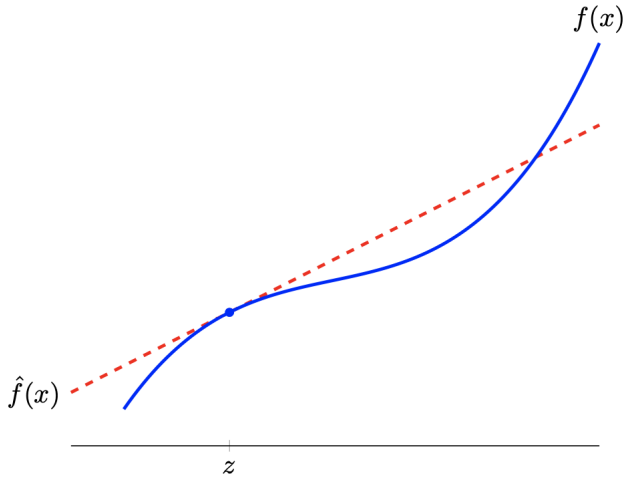


Figure 11: A function f of one variable, and the first-order Taylor approximation $\tilde{f}(x) = f(z) + f'(z)(x - z)$ at z.

Norm and Distance

Norm

The Euclidean norm of an n-vector $\mathbf{x}$ is denoted by $||x||$ and is defined as the square root of the sum of the square of its elements

$$||x|| = \sqrt{x_1^2 + x_2^2 + ... + x_n^2}$$

The Euclidean norm can also be expressed as the square root of the inner product of the vector with itself, i.e. $||x|| = \sqrt{x^T x}$. As a simple examples we have,

$$\left\| \begin{bmatrix} 2 \\ -1 \\ 2 \end{bmatrix} \right\| = \sqrt{9} = 3, \quad \left\| \begin{bmatrix} 0 \\ -1 \end{bmatrix} \right\| = 1$$

when x is a scalar (1-vector), the Euclidean norm is the same as the absolute value of x. Like the absolute value of a number, the norm of a vector is a numerical measure of its magnitude.

Properties of norm:

- Multiplying a vector by a scalar multiplies the norm by the absolute value of the scalar $||\beta\mathbf{x}|| = |\beta|||\mathbf{x}||$
- The Euclidean norm of a sum of two vectors is less than the sum of their norms $||x + y|| \leq ||x|| + ||y||$
- Non-negativity $||x|| \geq 0$
- Definiteness $||x|| = 0$ only if $x = 0$

Norm of a sum: The norm of the sum of two vectors x and y is:

$$||x + y|| = \sqrt{||x||^2 + 2x^T y + ||y||^2}$$

This is derived as follows:

$$||x+y||^2 = (x+y)^T(x+y) = x^T x+x^T y+y^T x+y^T y = ||x||^2+2x^T y+||y||^2$$

Taking the square root of both sides yields the formula for the norm.

Norm of block vectors: The norm-squared of a stacked vector is the sum of the norm-squared values of its sub-vectors. For example, with $d = (a, b, c)$ where $\mathbf{a}, \mathbf{b}$ and $\mathbf{c}$ are vectors, we have

$$||d||^2 = d^T d = a^T a + b^T b + c^T c = ||a||^2 + ||b||^2 + ||c||^2$$

Chebyshev inequality
Suppose that $\mathbf{x}$ is an $n-$vector, and that k of its entries satisfy $|x_i| \geq a$, where $a > 0$. Then k of its entries satisfy $x_i^2 \geq a^2$. It then follows that

$$||x||^2 = x_1^2 + \ldots + x_n^2 \geq ka^2$$

Since k of the numbers in the sum are at least a^2, and the other $n - k$ are positive. We can conclude that $k \leq ||x||^2/a^2$. This is called *Chebyshev inequality*. For $a > ||x||$, the inequality is $k \leq ||x||^2/a^2 < 1$, so we conclude that $k = 0$ since k is an integer. In other words, no entry of a vector can be larger than the norm of the vector.

Distance

Euclidean distance: we can use the norm to define the Euclidean distance between two vectors $\mathbf{a}$ and $\mathbf{b}$ as the norm of their

difference

$$dist(a, b) = ||a - b||$$

As shown in Figure 12, this is the distance between points with coordinates a and b and it could be in two or three dimensions. As an example, consider the 4-vectors

$$u = \begin{pmatrix} 1.8 \\ 2.0 \\ -3.7 \\ 4.7 \end{pmatrix}$$

$$v = \begin{pmatrix} 0.6 \\ 2.1 \\ 1.9 \\ 1.4 \end{pmatrix}$$

$$w = \begin{pmatrix} 2.0 \\ 1.9 \\ -4.0 \\ 4.6 \end{pmatrix}$$

The distance between pairs of them are

$$||u - v|| = 8.368$$
$$||u - w|| = 0.387$$
$$||v - w|| = 8.533$$

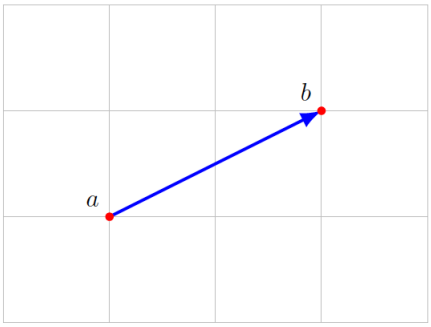


Figure 12: The norm of the displacement $b - a$ is the distance between the points with coordinates a and b.

Examples:

- **Feature distances:** If $\mathbf{x}$ and $\mathbf{y}$ represent vectors of n features of two objects, the quantity $||x-y||$ is called the feature distance and gives a measure of how different the objects are (in terms of their feature values). For example, the feature vectors can be associated with patients in a hospital, with entries such as weight, age, difficulty breathing, test results. We can use feature vector distance to say that one patient case is near another.

- **Nearest neighbor:** suppose $\mathbf{z_1}, \ldots, \mathbf{z_m}$ is a collection of m $n-$ vectors, and that $\mathbf{x}$ is another $n-$vector. We say that z_j is the nearest neighbor of $\mathbf{x}$ (among $\mathbf{z_1}, \ldots, \mathbf{z_m}$) if

$$||x - z_j|| \leq ||x - z_i|| \quad i = 1, \ldots, m$$

in other words $\mathbf{z_j}$ is the closest vector to $\mathbf{x}$ among the vectors $\mathbf{z_1}, \ldots, \mathbf{z_m}$.

Cauchy-Schwartz inequality

This is a relation between norms and inner products and is expressed as

$$|a^T b| \leq ||a||||b||$$

for any $n-$vector $\mathbf{a}$ and $\mathbf{b}$. Written out in terms of entries, this is

$$|a_1 b_1 + \ldots + a_n b_n| \leq (a_1^2 + \ldots + a_n^2)^{1/2}(b_1^2 + \ldots + b_n^2)^{1/2}$$

This clearly holds if $\mathbf{a} = \mathbf{0}$ and $\mathbf{b} = \mathbf{0}$ in which case both sides of the inequality would be zero. Assuming $\mathbf{a}$ or $\mathbf{b}$ to have non-zero values, and $\alpha = ||a||$ and $\beta = ||b||$ we have

$$\begin{aligned} 0 \leq ||\beta a – \alpha b||^2 &= ||\beta a||^2 – 2(\beta a)^T(ab) + ||\alpha b||^2 \\ &= \beta^2||a||^2 – 2\beta\alpha(a^T b) + \alpha^2||b||^2 \\ &= ||b||^2||a||^2 – 2||b||||a||(a^T b) + ||a||^2||b||^2 \\ &= 2||a||^2||b||^2 – 2||a||||b||(a^T b) \end{aligned} \tag{15}$$

Dividing by $2||a||||b||$ gives $a^T b \leq ||a||||b||$. When applying this inequality to $-a$ and b we get $-a^T b \leq ||a||||b||$. Putting these two inequalities we get Cauchy-Schwartz inequality as $|a^T b| \leq ||a||||b||$.

The Cauchy-Schwartz inequality can be used to verify the triangle inequality as follows:

Let **a** and **b** be any vectors. Then

$$||a+b||^2 = ||a||^2+2a^Tb+||b||^2 \leq ||a||^2+2||a||||b||+||b||^2 = (||a||+||b||)^2$$

where we used Cauchy-Schwartz inequality. Taking the square root we get the triangle inequality, $||a+b|| \leq ||a|| + ||b||$.

Angle between vectors:

The angle between two nonzero vectors **a** and **b** is defined as

$$\theta = arcos(a^Tb)/||a||||b||)$$

where *arcos* denotes the inverse cosine, normalized to lie in the interval $[0, \pi]$. We define θ as the unique angle between 0 and π that satisfies

$$a^Tb = ||a||||b||cos\theta$$

For example, the angle between the vectors $\mathbf{a} = (\mathbf{1}, \mathbf{2}, \mathbf{-1})$ and $\mathbf{b} = (\mathbf{2}, \mathbf{0}, \mathbf{-3})$ is

$$arcos(5/\sqrt{6}\sqrt{13}) = arcos(5.661) = 0.969 = 55.52\ deg$$

If the angle is $\pi/2 = 90\ deg$, $a^Tb = 0$ the vectors are said to be *orthogonal*.

If the angle is zero, $a^Tb = ||a||||b||$ the vectors are aligned with each vector being positive multiple of the other.

If the angle is $\pi = 180\ deg$, $a^T b = -||a||||b||$ the vectors are anti-aligned. Each vector is negative multiple of the other

If the angle is $< \pi/2$, the vectors are said to be acute angle which is $a^T b > 0$

If the angle is $> \pi/2$, the vectors are said to be obtuse angle which is $a^T b < 0$.

These are demonstrated in Figure 13.

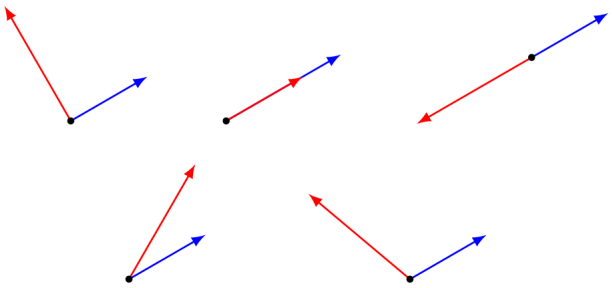


Figure 13: Top row. Examples of orthogonal, aligned, and anti-aligned vectors. Bottom row. Vectors that make an obtuse and an acute angle.

Norm of sum via angles

For vectors $\mathbf{x}$ and $\mathbf{y}$ we have

$$||x+y||^2 = ||x||^2 + 2x^T y + ||y||^2 = ||x||^2 + 2||x||||y||cos\theta + ||y||^2 \tag{16}$$

where θ is the angle between $\mathbf{x}$ and $\mathbf{y}$.

if $\mathbf{x}$ and $\mathbf{y}$ are aligned ($\theta = 0$), we have $||x + y|| = ||x|| + ||y||$

if $\mathbf{x}$ and $\mathbf{y}$ are orthogonal ($= 90$ deg.), we have $||x + y||^2 = ||x||^2 + ||y||^2$. In this case the norm-squared values add and we have $||x + y|| = \sqrt{||x||^2 + ||y||^2}$.

Linear Independence

A collection of $n-$vectors $\mathbf{a_1}, \ldots, \mathbf{a_k}$ (with $k \geq 1$) is called linearly dependent if

$$\beta_1 a_1 + \ldots + \beta_k a_k = 0$$

holds for some $\beta_1, \ldots, \beta_k$ that are not all zero.

When a collection of vectors is linearly dependent, at least one of the vectors can be expressed as the linear combination of other vectors: if $\beta_i \neq 0$ in the above equation, we take $\beta_i a_i$ to one side and divide the equation by β_i gives

$$a_i = (-\beta_1/\beta_i)a_1 + \ldots + (-\beta_{i-1}/\beta_i)a_{i+1} + \ldots + (-\beta_k/\beta_i)a_k$$

Linearly independent vectors

A collection of n-vectors $\mathbf{a_1}, \ldots, \mathbf{a_k}$ (with $k \geq 1$) is called linearly independent if it is not linearly dependent, which means that

$$\beta_1 a_1 + \ldots + \beta_k a_k = 0$$

only holds for $\beta_1 = \ldots = \beta_k = 0$. In other words, the only linear combination of the vectors that equals the zero vector is the linear combination with all coefficients zero.

A list of vectors is linearly dependent only if any one of the vectors is a multiple of another one.

The vectors are linearly dependent if we could express one as a linear combination of the others. The vectors

$$a_1 = \begin{pmatrix} 0.2 \\ -7.0 \\ 8.6 \end{pmatrix} \tag{17}$$

$$a_2 = \begin{pmatrix} -0.1 \\ 2.0 \\ -1.0 \end{pmatrix} \tag{18}$$

$$a_3 = \begin{pmatrix} 0.0 \\ -1.0 \\ 2.2 \end{pmatrix} \tag{19}$$

are linearly dependent since $a_1 + 2a_2 - 3a_3 = 0$. Here we can express any of these vectors as a linear combination of the other two.

The vectors

$$a_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{20}$$

$$a_2 = \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix} \tag{21}$$

$$a_3 = \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix} \tag{22}$$

are linearly independent. To see this, suppose $\beta_1 a_1 + \beta_2 a_2 + \beta_3 a_3 = 0$. This means that

$$\beta_1 - \beta_3 = 0 \;\; -\beta_2 + \beta_3 = 0 \;\; \beta_2 + \beta_3 = 0$$

adding the last two equations we find $2\beta_3 = 0$, so $\beta_3 = 0$. This means $\beta_1 = \beta_2 = 0$.

Basis

A collection of n linearly independent $n-$vectors is called a basis. If the $n-$ vectors $\mathbf{a_1}, \ldots, \mathbf{a_n}$ are a basis, then any n-vector $\mathbf{b}$ can be written as a linear combination of them. To see this consider the collection of $n+1$ $n-$vectors $\mathbf{a_1}, \ldots, \mathbf{a_n}, \mathbf{b}$. by the independent

dimension inequality, these vectors are linearly dependent, so there is $\beta_1, \ldots, \beta_{n+1}$, not all zero, that satisfy

$$\beta_1 a_1 + \ldots + \beta_n a_n + \beta_{n+1} b = 0$$

If $\beta_{n+1} = 0$, then we have

$$\beta_1 a_1 + \ldots + \beta_n a_n = 0$$

which since $a_1, \ldots, a_n$ are linearly independent, implies that $\beta_1 = \ldots = \beta_n = 0$. But then all the β_i are zero- a contradiction. Therefore, we conclude that $\beta_{n+1} \neq 0$. It then follows that

$$b = (-\beta_1/\beta_{n+1})a_1 + \ldots + (-\beta_n/\beta_{n+1})a_n$$

which means $\mathbf{b}$ is a linear combination of $\mathbf{a_1}, \ldots, \mathbf{a_n}$.

Any n-vector $\mathbf{b}$ *can be written in a unique way as a linear combination of a basis* $\mathbf{a_1}, \ldots, \mathbf{a_n}$.

Expansion in a basis

When we express $n-$vector $\mathbf{b}$ as a linear combination of a basis $a_1 \ldots a_n$, we refer to

$$b = \alpha_1 a_1 + \ldots + \alpha_n a_n$$

as the expansion of $\mathbf{b}$ in the $a_1 \ldots a_n$ basis. The coefficients $\alpha_1 \ldots \alpha_n$ are called the coefficients of the basis $\mathbf{a_1} \ldots \mathbf{a_n}$.

Any vector $\mathbf{b}$ can be written as a linear combination of unit $n-$ vectors $\mathbf{e_1}, \ldots, \mathbf{e_n}$ as the basis

$$b = b_1 e_1 + \ldots + b_n e_n$$

Orthonormal Vectors

A collection of vectors $\mathbf{a_1} \ldots \mathbf{a_k}$ is orthogonal if $\mathbf{a_i} \perp \mathbf{a_j}$ for any i, j with $i \neq j$, $i, j = 1, , k$ (Fig 14). A collection of vectors $\mathbf{a_1} \ldots \mathbf{a_k}$ is orthonormal if it is orthogonal and $||a_i|| = 1$ for $i = 1, \ldots, k$. A vector of norm one is called normalized. Dividing a vector by its norm is called "normalizing". In other words, if the inner products of pairs of vectors in the set $\mathbf{a_1}, \ldots, \mathbf{a_k}$ is orthonormal, it means

$$a_i^T a_j = \begin{cases} 1 & \text{for} \quad i = j, \\ 0 & \text{for} \quad i \neq j \end{cases} \tag{23}$$

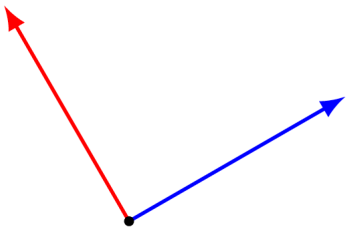


Figure 14: Orthonormal vectors in a plane.

Example: The following 3-vectors are orthonormal as the inner product of any pair of them is zero.

$$\begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} \tag{24}$$

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \tag{25}$$

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} \tag{26}$$

Orthonormal vectors are linearly independent

Suppose $a_1, \ldots, a_k$ are orthonormal, and

$$\beta_1 a_1 + \ldots + \beta_k a_k = 0$$

Taking the inner product of this equality with a_i yields

$$0 = a_i^T(\beta_1 a_1 + \ldots + \beta_k a_k) = \beta_1(a_i^T a_1) + \ldots + \beta_k(a_i^T a_k) = \beta_i$$

since $a_i^T a_j$ for $j \neq i$ and $a_i^T a_i = 1$. Therefore, the only linear combination of $a_1, \ldots, a_k$ that is zero is the one with all coefficients zero.

Example: We express the 3-vector $\mathbf{x} = (\mathbf{1}, \mathbf{2}, \mathbf{3})$ as a linear combination of the orthonormal basis given in equations 24-26. The inner products of $\mathbf{x}$ with these vectors are

$$\begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}^T x = -3 \tag{27}$$

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}^T x = \frac{3}{\sqrt{2}} \tag{28}$$

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}^T x = -\frac{1}{\sqrt{2}} \tag{29}$$

Therefore the expansion of $\mathbf{x}$ in its basis is

$$x = -3\begin{bmatrix}0\\0\\-1\end{bmatrix} + \frac{3}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}\begin{bmatrix}1\\1\\0\end{bmatrix}\right) + \frac{-1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}\begin{bmatrix}1\\-1\\0\end{bmatrix}\right)$$

Question 1.5: Taylor approximation of norm

Find a general formula for the Taylor approximation of the function $f(x) = ||x||$ near a given nonzero vector z. You can express the approximation in the form $\hat{f}(x) = a^T(x - z) + b$.

Solution: let $x \in \Re^n$ and $x = (x_1\ x_2\ ...\ x_n)^T$. Hence, $f(x) = ||x|| = \sqrt{x_1^2 + x_2^2 + ...}$ First derivation of $f(x)$ can be found as $\nabla f(x) = \frac{x}{||x||}$, So, the fist term of Taylor expansion cab be written as:

$$f(x) = f(z) + (\nabla f(z))^T(x - z) = a^T(x - z) + b$$

where $a = \nabla f(z) = \frac{z}{||z||}$ and $b = f(z)$.

1.4 Matrices

A matrix is an array of numbers written between rectangular brackets or large parenthesis.

$$T = \begin{pmatrix} t_{11} & t_{12} & \dots & t_{1n} \\ t_{21} & t_{22} & \dots & t_{2n} \\ & \vdots & & \\ t_{m1} & t_{m2} & \dots & t_{mn} \end{pmatrix} \tag{30}$$

An important attribute of a matrix is its size or dimension which is the number of rows and columns. The above matrix has three rows and four columns. Its size is 3×4. The entries in a matrix are called elements. The i, j element is the value in the i^{th} row and j^{th} column. The i, j element of a matrix A is denoted by A_{ij}. If A is an $m \times n$ matrix, then the row index i runs from 1 to m and the column index j runs from 1 to n.

Consider the following matrix

$$T = \begin{pmatrix} 3 & 5 & 10 & -6 \\ 2 & 4 & 8 & 12 \\ 7 & -5 & 4 & 2 \end{pmatrix} \tag{31}$$

If the above matrix is B, then we have $B_{13} = 10$, $B_{32} = -5$.

Two matrices are equal if they have the same size, and their corresponding entries are all equal. The set of real $m \times n$ matrices is

denoted $\Re^{m \times n}$

An n-vector can be interpreted as an $n \times 1$ matrix. There is no distinction between vectors and matrices with one column. A matrix with only one row $1 \times n$ is called a row vector. A 1×1 matrix is considered to be a scalar.

An $m \times n$ matrix A has n columns given by

$$a_j = \begin{pmatrix} A_{ij} \\ \cdot \\ \cdot \\ A_{mj} \end{pmatrix} \tag{32}$$

for $j = 1, \ldots, n$. The same matrix has m rows, given by $n-$ row vectors

$$b_j = [A_{i1} \ldots A_{in}]$$

for $i = 1, \ldots, m$.

Block Matrices

In some cases the entries to matrices are themselves matrices. These are called Blocked Matrices, as in

$$A = \begin{pmatrix} B & C \\ D & E \end{pmatrix} \tag{33}$$

Where B, C, D and E are matrices. The elements B, C, D and E are called sub-matrices of A. Block matrices must have right dimensions to fit together. Matrices in the same block row must have the same number of rows while matrices in the same block column must have the same number of columns.

Examples of Matrix Application:

- **Images:** A black and white image with $M \times N$ pixels can naturally be expressed with an $M \times N$ matrix. The i gives the vertical position of the pixel and the column index j the horizontal position of the pixel. The i, j entry gives the value of the pixel.

- **Rainfall data:** An $m \times n$ matrix A gives the rainfall at m different locations on n consecutive days. For example A_{42} is the rainfall at location 4 in day 2. The j^{th} column of A, which is an m vector, gives the rainfall at the m locations on day j. The i^{th} row of A, which is an $n-$ row vector is the time series of rainfall at location i.

- **Asset returns:** A $T \times n$ matrix R gives the returns of a collection of n assets over T periods. Therefore, R_{ij} gives the return of asset j in period i. For example $R_{12,7} = -0.03$ means that asset 7 had a 3% loss in period 12. The 4^{th}

column of R is a T vector that is the return time series for asset 4. The 3^{rd} row of R is an n-row vector that gives the returns of all assets in period 3.

- **Prices from multiple suppliers:** An $m \times n$ matrix P gives the prices of n different goods from m different suppliers. P_{ij} is the price that supplier i charges for good j. The j^{th} column of P is the $m-$ vector of supplier prices for good j. The i^{th} row gives the prices for all goods from supplier i.

Zero and identity matrix: A zero matrix is a matrix with all its elements equal to zero. An identity matrix is always square. Its diagonal elements (those with equal rows and columns) are all equal to one, with its off-diagonal elements being all zero. The identity matrix is often denoted by the letter I. An identity matrix can then be expressed as follows:

$$I_{ij} = \begin{cases} 1 & \text{for} \quad i = j, \\ 0 & \text{for} \quad i \neq j \end{cases} \tag{34}$$

For example $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ are 2×2 and 4×4 identity matrices.

Sparse Matrix:

A matrix A is called to be sparse if many of its elements are zero (just a few of its entries are non-zero). Its sparsity pattern is the set of indices (i, j) for which $A_{ij} \neq 0$. The number of non-zeros (nnz) of a sparse matrix A is the number of entries in its sparsity pattern and is denoted by $nnz(A)$. If A is $m \times n$ we have $nnz(A) \leq mn$. Its density is $nnz(A)/(mn)$ which is not more than one. There is no rule as what the number $nnz(A)$ should be to make a matrix sparse.

An $n \times n$ identity matrix is sparse since it has only n non-zeros. So, its density is $n/(n \times n) = 1/n$. A zero matrix is the sparsest possible matrix.

Diagonal matrices:

A square matrix $n \times n$ is diagonal if $A_{ij} \neq 0$ for $i = j$. The entries of a matrix with $i = j$ are diagonal elements. All off-diagonal entries of a diagonal matrix are zero. An examples of a diagonal matrice is:

$$\begin{pmatrix} 3 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 10 \end{pmatrix}$$

The notation $diag(a_1, \ldots, a_n)$ is used to denote the $n \times n$ diagonal

matrix A with diagonal entries $A_{11} = a_1, \ldots, A_{nn} = a_n$. By definition $I = diag(1)$.

Transpose Matrices:

If A is an $m \times n$ matrix, its transpose denoted by A^T is the $n \times m$ matrix expressed by $(A^T)_{ij} = A_{ji}$. In other words the rows and columns of A are transposed in A^T. For example

$$\begin{pmatrix} 3 & 5 \\ 7 & 15 \\ 4 & 10 \end{pmatrix}^T = \begin{pmatrix} 3 & 7 & 4 \\ 5 & 15 & 10 \end{pmatrix}$$

If we transpose a matrix twice, we get back to its original form $(A^T)^T = A$. The transcript T in transpose is the same one used to denote the inner product of two n-vectors.

Symmetric Matrices:

A square matrix A is symmetric if $A = A^T$- i.e. $A_{ij} = A_{ji}$ for all i, j. One example of the usage of symmetric matrices is the friend relation on a set of n people, where $(i, j) \in R$ means that person i and person j are friends (person j and person i would also be friends).

Properties of transpose matrices

$$(A^T)^T = A$$
$$(AB)^T = B^T A^T$$
$$(A + B)^T = A^T + B^T$$

Addition, Subtraction and Norm

Addition of two matrices. Two matrices of the same size can be added together. The result is another matrix of same size obtained by summing the corresponding elements of the two matrices. For example,

$$\begin{pmatrix} 3 & 5 \\ 7 & 15 \\ 4 & 10 \end{pmatrix} + \begin{pmatrix} 2 & 6 \\ 12 & 15 \\ 14 & 10 \end{pmatrix} = \begin{pmatrix} 5 & 30 \\ 19 & 15 \\ 18 & 20 \end{pmatrix}$$

Matrix subtraction is similar

Properties of matrix addition

We assume that A, B and C are matrices of the same size. The following properties can be derived from the matrix definition:

- **Commutativity:** $A + B = B + A$
- **Associativity:** $(A + B) + C = A + (B + C)$
- **Addition with zero matrix:** $A + 0 = 0 + A = A$

- **Transpose of sum:** $(A + B)^T = A^T + B^T$

Scalar-matrix multiplication:

The scalar-matrix multiplication is defined the same way as for the vectors and is defined by multiplying every element of a matrix by the scalar. For example

$$(-2)\begin{pmatrix} 3 & 5 \\ 7 & 15 \\ 4 & 10 \end{pmatrix} = \begin{pmatrix} -6 & -10 \\ -14 & -30 \\ -8 & -20 \end{pmatrix}$$

If β and γ are scalars and A is a matrix, we have

$$(\beta A)^T = \beta A^T$$

and

$$(\beta + \gamma)A = \beta A + \gamma A$$

In matrix algebra, matrix multiplication has a higher precedence than matrix addition which means that we can carry out multiplication before addition (when there are no parenthesis to fix the order).

Matrix norm

The norm of an $M \times n$ matrix A, denoted by $||A||$, is the square root of the sum of the squares of its entries

$$||A|| = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2}$$

This is the same as the definition of our vector norm, when A is a vector and $n = 1$. The norm of an $m \times n$ matrix is the norm of an mn- vector formed from the entries of the matrix. Like the vector norm, the matrix norm is a quantitative measure of the magnitude of a matrix. In some applications, it is more natural to use the RMS values of matrix entries, $||A||/\sqrt{mn}$ as a measure of matrix size.

Properties of matrix norm:

For any $m \times n$ matrix A, we have $||A|| \geq 0$ and $||A|| = 0$ only if $A = 0$. The matrix norm is non-negative homogeneous. For any scalar γ and $m \times n$ matrix A, we have $||\gamma A|| = |\gamma|||A||$. Also, for any $m \times n$ matrices A and B, we have the triangle inequality

$$||A + B|| \leq ||A|| + ||B||$$

Distance between two matrices is calculated as $||A–B||$. For matrix norm, we have $||A|| = ||A^T||$. The norm of a matrix is the same as the norm of its transpose. Also,

$$||A||^2 = ||a_1||^2 + \ldots + ||a_n||^2$$

where $a_1, \ldots, a_n$ are the columns of A. In another word, the square norm of a matrix is the sum of the squared norms of its columns.

The Trace

Trace of a square matrix is the sum of all its diagonal elements of that matrix

$$Tr(A) = \sum_{i=1}^{n} A_{ii}$$

For matrix A and B and scalar t, properties of trace matrix include:

$Tr(A) = Tr(A^T)$
$Tr(A + B) = Tr(A) + Tr(B)$
$Tr(tA) = tTr(A)$
For A and B such that AB is square, $TrAB = TrBA$

1.5 Matrix-vector multiplication

If A is an $m \times n$ matrix and $\mathbf{x}$ an $n-$vector, then the matrix-vector product $y = Ax$ is the $m-$vector $\mathbf{y}$ with elements

$$y_i = \sum_{k=1}^{n} A_{ik}x_k = A_{i1}x_1 + \ldots + A_{in}x_n \quad (\text{for } i = 1, \ldots, m)$$

as an example, we multiply the following matrix with a vector

$$\begin{pmatrix} 0 & 2 & -1 \\ -2 & 1 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix} = \begin{pmatrix} (0)(2)+(2)(1)+(-1)(-1) \\ (-2)(2)+(1)(1)+(1)(-1) \end{pmatrix} = \begin{pmatrix} 3 \\ -4 \end{pmatrix}$$

Row and column interpretation:

We can express the vector-matrix product in terms of the rows and columns of a matrix. In the above relation, y is the inner product of x with the i^{th} row of matrix A

$$y_i = b_i^T x \quad (\text{for } i = 1, \ldots, m)$$

where b_i^T is the i^{th} row of matrix A. The matrix-vector product can also be interpreted in terms of the columns of A. If a_k is the k^{th} column of matrix A, then $y = Ax$ can be written

$$y = x_1 a_1 + \ldots + x_n a_n$$

This shows that $y = Ax$ is a linear combination of the columns of A with the coefficients in the linear combination being the elements of x.

Examples:

- **Total price from multiple suppliers**: Suppose the $m \times n$ matrix P gives the prices of n goods from m suppliers. If q is an $n-$vector of quantities of the n goods then $c = Pq$

is an $N-$vector that gives the total cost of the goods from each of the N suppliers.

- **Portfolio return time series**: Suppose that R is a $T \times n$ asset return matrix, that gives the returns of n assets over T periods. A common trading strategy maintains constant investment weights given by the $n-$vector w over the T periods. For example, $w_4 = 0.15$ means that 15% of the total portfolio value is held in asset 4. Then Rw, which is a $T-$vector, is the time series of the portfolio returns over the period $1, \ldots, T$.

- **Document scoring**: Let A be an $N \times n$ document term matrix., which gives the word counts of a corpus of N documents using a dictionary of n words. Therefore, the rows of A are the word count vectors for the documents. Suppose that $\mathbf{w}$ is an n-vector that gives a set of weights for the words in the dictionary. Then $s = Aw$ is an N-vector that gives the scores of the documents, using the weights and the word counts. A search engine could then choose w based on the search query so that the scores are predictions of relevance of the document to the search.

Inner product

When $\mathbf{a}$ and $\mathbf{b}$ are $n-$vectors, $a^T b$ is the inner product of $\mathbf{a}$ and $\mathbf{b}$, obtained from transposing matrices and forming a matrix-vector product. Starting with the column $n-$vector $\mathbf{a}$, consider it as an

$n \times 1$ matrix and transpose it to obtain the $n-$row vector $\mathbf{a^T}$. Now we multiply this $1 \times n$ matrix by the $n-$vector $\mathbf{b}$, to obtain the 1-vector $a^T\ b$, which we also consider a scalar. The notation $a^T\ b$ for the inner product is just a special case of matrix-vector multiplication.

1.6 Application of Matrices

Here we describe some of the applications of matrices.

1.6.1 Geometric Transformation

Several geometric transformations or mapping from point-to-point can be expressed as matrix-vector products $y = Ax$, with A a 2×2 (or 3×3) matrix. In the following we consider mapping from x to y in the 2-D case.

- **Scaling:** Scaling is the mapping $y = ax$, where a is a scalar. This can be expressed as $y = Ax$ with $A = a\ I$. This mapping stretches a vector by the factor $|a|$ (or shrinks it when $|a| < 1$ and it flips the direction of a vector if $a < 0$.
- **Dilation:** Dilation is the mapping $y = D\ x$, where D is a diagonal matrix, $D = diag(d_1, d_2)$. This mapping stretches the vector x by different factors along the two different axes (or shrinks if $|d_i| < 1$, and flips if $d_i < 0$).
- **Rotation:** Suppose that y is the vector obtained by rotating x by θ radians counterclockwise. This matrix is called

the rotation matrix

$$\begin{pmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{pmatrix}$$

- **Reflection:** Suppose that $\mathbf{y}$ is the vector obtained by reflecting $\mathbf{x}$ through the line that passes through the origin, inclined θ radians with respect to horizontal. Then we have

$$\begin{pmatrix} cos(2\theta) & sin(2\theta) \\ sin(2\theta) & -cos(2\theta) \end{pmatrix}$$

- **Projection onto a line:** The projection of the point x onto a set is the point in the set that is closest to x. Suppose y is the projection of x onto the line that passes through the origin, inclined θ radians with respect to horizontal. We have

$$y = \begin{pmatrix} (1/2)(1 + cos(2\theta)) & 1/2sin(2\theta) \\ 1/2sin(2\theta) & (1/2)(1 - cos(2\theta)) \end{pmatrix} x$$

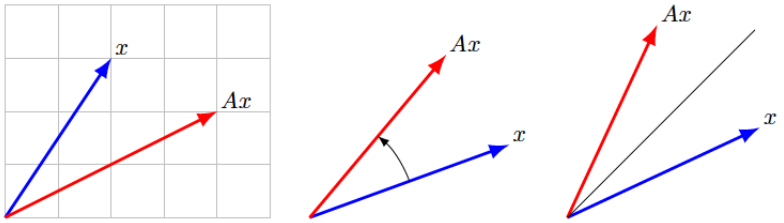


Figure 15: From left to right: A dilation with A = diag(2, 2/3), a counterclockwise rotation by $\pi = 6$ radians, and a reflection through a line that makes an angle of $\pi/4$ radians with the horizontal line.

Question 1.6: 3-D rotation

Let x and y be 3-vectors representing positions in 3-D. Suppose that the vector y is obtained by rotating the vector x about the vertical axis (i.e., e3) by 45° (counterclockwise, i.e., from e1 toward e_2). Find the 3×3 matrix A for which $y = Ax$. Hint: Determine the three columns of A by finding the result of the transformation on the unit vectors e_1, e_2, e_3.

Solution:

For a given angle θ $\begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

For $\theta = 45°$:

$$y = Ax = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{pmatrix} x$$

1.6.2 Convolution

The convolution of f and g is written as $f * g$ and is defined as the integral of the product of the two functions after one is reversed and shifted

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau$$

with an equivalent definition being

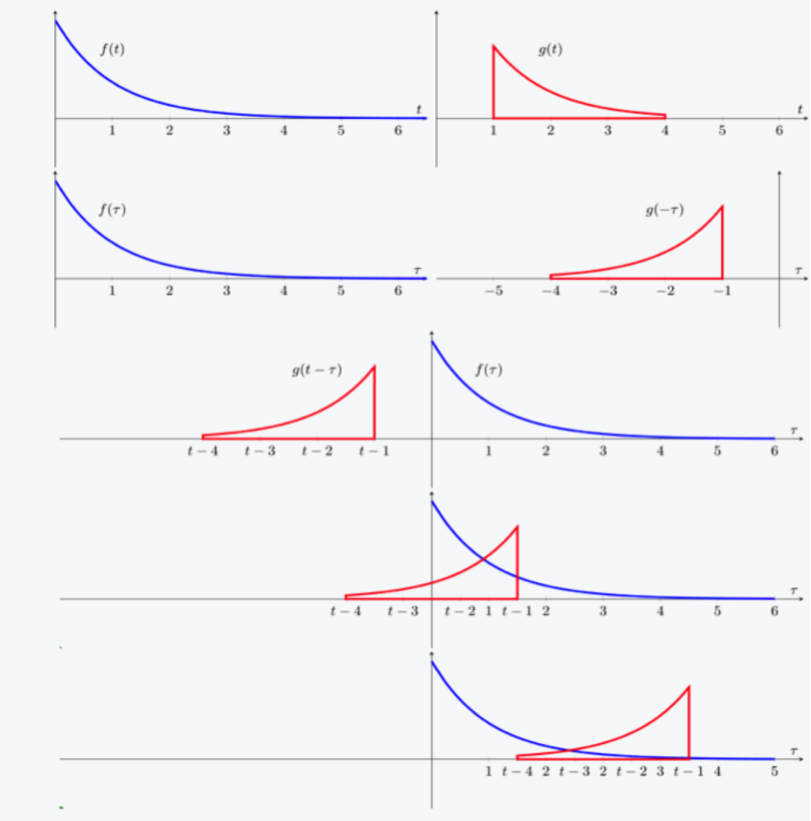
$$(f * g)(t) = \int_{-\infty}^{\infty} f(t - \tau)g(\tau)d\tau$$

Since convolution is commutative. In principle, the convolution can be described as the weighted average of the function $f(\tau)$ at the moment t where the weighting is given by $g(-\tau)$ shifted by t. As t changes, the weighting function covers different parts of the input function.

Example:

To find convolution of two functions g and f, one needs to take the following steps:

- express each function in terms of a dummy variable τ.
- Reflect one of the functions $g(\tau)g(-\tau)$.
- Add a time offset, t, which allows $g(t-\tau)$ to slide along the $\tau-$axis.
- Start at $-\infty$ and slide it to $+\infty$. Wherever the two functions intersect, find the integral of their product. This is equivalent to a weighted sum of function $f(\tau)$, where the weighting function is $g(-\tau)$.
- The resulting function is the convolution of functions f and g.



1.6.3 Convolution of two vectors:

The convolution of an n-vector $\mathbf{a}$ and an m-vector $\mathbf{b}$ is the $(n + m - 1)$-vector denoted by $\mathbf{c} = \mathbf{a} * \mathbf{b}$, with entries

$$c_k = \sum_{i+j=k+1} a_i \, b_j, \quad k = 1, \ldots, n + m - 1$$

where we sum over all values of i and j in their index ranges $1, \ldots, n$ and $1, \ldots, m$ for which the sum $i + j$ is $k + 1$.

For example, with $n = 4$, $m = 3$, we have

$$\begin{aligned} c_1 &= a_1\ b_1 \\ c_2 &= a_1\ b_2 + a_2\ b_1 \\ c_3 &= a_1\ b_3 + a_2\ b_2 + a_3\ b_1 \\ c_4 &= a_2\ b_3 + a_3\ b_2 + a_4\ b_1 \\ c_5 &= a_3\ b_3 + a_4\ b_2 \\ c_6 &= a_4\ b_3 \end{aligned}$$

Convolution reduces to ordinary multiplication when $n = m = 1$ and to scalar-vector multiplication when either n or m is one.

2-D Convolution:

Convolution can be extended to multiple dimensions. Suppose that A is an $m \times n$ matrix and B is a $p \times q$ matrix. Their convolution is the $(m + p - 1) \times (n + q - 1)$ matrix

$$C_{rs} = \sum_{i+k=r+1, j+l=s+1} A_{ij}\, B_{kl} \quad r = 1, \ldots, m+p-1\ s = 1, \ldots, n+q-1$$

where the indices are restricted to their ranges (i.e. A_{ij} and B_{kl} are zero when the indices are out of range).

Properties of Convolution

- Commutative $A * B = B * A$
- Associative $(A * B) * C = A * (B * C)$
- For a fixed B, $A * B$ is a linear function of A.

Examples of Convolution

Image blurring: If the $m \times n$ matrix X represents an image, $Y = X * B$ represents the effect of blurring the image by a Point Spread Function (PSF) given by the entries of the matrix B. Consider the matrix

$$\begin{pmatrix} 1/4 & 1/4 \\ 1/4 & 1/4 \end{pmatrix}$$

$Y = X * B$ is an image where each pixel value is the average of a 2×2 block of 4 adjacent pixels in X (Figure 15). The image Y would be perceived as image X with some blurring of the fine details. For a 8×9 matrix this is expressed as

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

And its convolution with B matrix is

$$\begin{pmatrix} 1/4 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/4 \\ 1/2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1/2 \\ 1/2 & 1 & 3/4 & 1/2 & 1/2 & 1/2 & 1/2 & 3/4 & 1 & 1/2 \\ 1/2 & 1 & 3/4 & 1/4 & 1/4 & 1/2 & 1/4 & 1/2 & 1 & 1/2 \\ 1/2 & 1 & 1 & 1/2 & 1/2 & 1 & 1/2 & 1/2 & 1 & 1/2 \\ 1/2 & 1 & 1 & 1/2 & 1/2 & 1 & 1/2 & 1/2 & 1 & 1/2 \\ 1/2 & 1 & 1 & 3/4 & 3/4 & 1 & 3/4 & 3/4 & 1 & 1/2 \\ 1/2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1/2 \\ 1/4 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1/4 \end{pmatrix}$$

Withe the PSF $D^{hor} = [1 \quad -1]$, the pixel values in the image $Y = X * D^{hor}$ are the horizontal first order differences of those in X:

$$Y_{ij} = X_{ij} - X_{i,j-1} \quad i = 1, ..., m \;\; j = 2, ..., n$$

and $Y_{i1} = X_{i1}, X_{i,n+1} = -X_{in}$ for $i = 1, ..., m$. With a PSF

$$D^{ver} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

the pixel values in the image $Y = X * D^{ver}$ are the vertical first order differences of those in X:

$$Y_{ij} = X_{ij} - X_{i-1,j} \quad i = 2, ..., m \;\; j = 1, ..., n$$

and $Y_{1j} = X_{1j}, X_{m+1,j} = -X_{mj}$ for $j = 1, ..., n$.

Convolution of two Functions:

Convolution is a mathematical operation on two functions (f and g) that produces a third function expressing how the shape of one is modified by the other. It is defined as the integral of the product of the two functions after one is reversed and shifted. Some features of convolution are similar to cross-correlation: for real-valued functions, of a continuous or discrete variable, it differs from cross-correlation only in that either $f(x)$ or $g(x)$ is reflected about the y-axis; thus it is a cross-correlation of $f(x)$ and $g(-x)$, or $f(-x)$ and $g(x)$. For continuous functions, the cross-correlation

operator is the adjoint of the convolution operator. Convolution has applications in statistics, probability, computer vision, image processing, language processing and signal processing. The convolution of f and g is written $f * g$, denoting the operator with the symbol $*$. It is defined as the integral of the products of the two functions after one is reversed and shifted. In the integral form, it is defined as

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau)g(t - \tau)d\tau$$

an equivalent definition is

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(t - \tau)g(\tau)d\tau$$

the convolution formula can be described as a weighted average of the function $f(\tau)$ at the moment t where the weighting is given by $g(–\tau)$ simply shifted by amount t. As t changes, the weighting function emphasizes different parts of the input function.

- Express each function in terms of a dummy variable τ.
- Reflect one of the functions: $g(\tau) - - > g(-\tau)$.
- Add a time-offset, t, which allows $g(t-\tau)$ to slide along the τ axis.

- Start t at $-\infty$ and slide it all the way to $+\infty$. Whenever the two functions intersect, find the integral of their product. Compute a sliding, weighted sum of function $f(\tau)$, where the weighting function is $g(-\tau)$. The result is the convolution of the functions f and g.

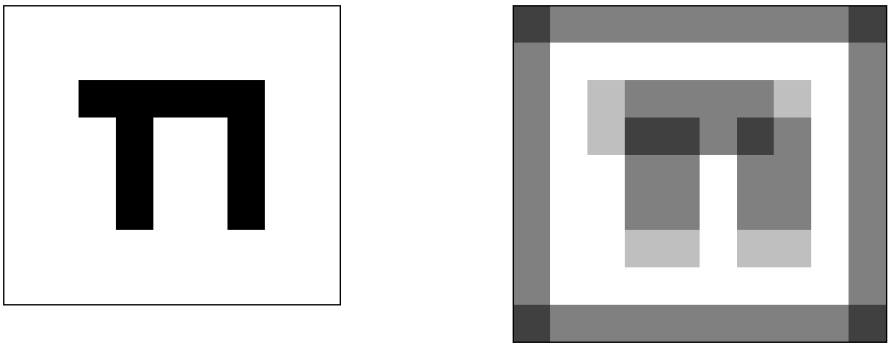


Figure 16: An 8×9 image and its convolution with the point spread function

Taylor Approximation

Suppose the function f is differentiable (i.e. has partial derivatives) and z is an $n-$vector. The first order Taylor approximation of f near z is given by

$$\tilde{f}(x) = f_i(z) + \frac{\partial f_i}{\partial x_1}(z)(x_1 - z_1) + \cdots + \frac{\partial f_i}{\partial x_n}(z)(x_n - z_n)$$

For $i = 1, \ldots, m$. This is the first order Taylor approximation of each of the scalar valued functions f_i, described before. For x near

z, $\tilde{f}(x)$ is a good approximation of $f(x)$. In matrix-vector form, this can be written in a compact for

$$\tilde{f}(x) = f(z) + Df(z)(x - z)$$

Where $m \times n$ matrix $Df(z)$ is the derivative or *Jacobian* matrix of f at z with its components being the partial derivatives of f

$$Df(z)_{ij} = \frac{\partial f_i}{\partial x_j}(z) \;\; i = 1, \ldots, m \;\; j = 1, \ldots, n$$

Evaluated at point z. The rows of the Jacobian are $\nabla f_i(z)^T$, for $i = 1, \ldots, m$.

Matrix-Matrix Multiplication

One can multiply two matrices A and B provided their dimensions are compatible. This means the number of columns of A equals the number of rows of B. Suppose A and B are compatible eg. A has size $m \times p$ and B has size $p \times n$. Then the product matrix $C = AB$ is the $m \times n$ matrix with elements

$$C_{ij} = \sum_{k=1}^{p} A_{ik}B_{kj} = A_{i1}B_{1j} + \ldots + A_{ip}B_{pj} \;\; i = 1, \ldots m, j = 1, \ldots, n$$

The summation above is interpreted as moving left to right along the i^{th} row of A while moving top to bottom down in the j^{th}

column of B, keeping a sunning sum of the product of elements.

As an example, consider

$$\begin{pmatrix} -1.5 & 3 & 2 \\ 1 & -1 & 0 \end{pmatrix} \begin{pmatrix} -1 & -1 \\ 0 & -2 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 3.5 & -4.5 \\ -1 & 1 \end{pmatrix}$$

To find the (1,2) entry of the right hand matrix, we move along the first row of the left-hand matrix, and down the second column of the middle matrix, to get $(-1.5)(-1) + (3)(-2) + (2)(0) = -4.5$.

Inner product

A special case of matrix-matrix multiplication is the multiplication of a row vector with a column vector. If a and b are n-vectors then the inner product

$$a^T\ b = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n$$

can be interpreted as the matrix-matrix product of the $1 \times n$ matrix a^T and the $n \times 1$ matrix b. The result is a 1×1 matrix, which is a scalar.

Outer product

The outer product of an $m-$vector $\mathbf{a}$ and an $n-$vector $\mathbf{b}$ is given by $\mathbf{a^T b}$, which is an $m \times n$ matrix

$$a\ b^T = \begin{pmatrix} a_1b_1 & a_1b_2 & \dots & a_1b_n \\ a_2b_1 & a_2b_2 & \dots & a_2b_n \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ a_mb_1 & a_mb_2 & \dots & a_mb_n \end{pmatrix}$$

whose entries are all products of the entries of a and b.

Product of block matrices: Suppose A is a block matrix with $m \times p$ entries A_{ij} and B a block matrix with $p \times n$ block entries B_{ij}. We have matrix products $A_{ik}B_{kj}$ for $k = 1, \dots, p$. In general

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} E & F \\ G & H \end{pmatrix} = \begin{pmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{pmatrix}$$

For any matrices A, B, C and D whose products make sense.

Matrix multiplication order

Matrix multiplication is not commutative- we do not have $AB = BA$. In fact BA may not make sense and even if it does, may have a different size than AB. For example, if A is 2×3 and B is 3×4, then AB makes sense (the dimensions are compatible)

but BA does not make sense. Even when both AB and BA make sense (both matrices are squares), we do not have $AB = BA$.

Properties of matrix multiplication

Assuming A, B and C are matrices and γ a scalar,

- Associativity: $(AB)C = A(BC)$ – the product can therefore be written as ABC.
- Associativity with scalar multiplication – $\gamma(AB) = (\gamma A)B = A(\gamma B)$
- Distributivity: Matrix multiplication distributes across matrix addition: $A(B + C) = AB + AC$
- Transpose of the products: The transpose of a product is the product of the transposes but in the opposite order $(AB)^T = B^T A^T$

Question 1.7: Rainfall and river height

The T-vector r gives the daily rainfall in some regions over a period of T days. The vector h gives the daily height of a river in the region (above its normal height). By careful modeling of water ow, or by fitting a model to past data, it is found that these vectors are (approximately) related by convolution: $h = g*r$, where $g = (0.1, 0.4, 0.5, 0.2)$: Give a short story in English (with no mathematical terms) to approximately describe this relation. For example, you might mention how many days after

a one day heavy rainfall the river height is most affected. Or how many days it takes for the river height to return to the normal height once the rain stops.

Solution:

Consider the first few entries of h, we know that the convolution is $h = g * r$, then:

$$h_1 = 0.1r_1$$

$$h_2 = 0.1r_2 + 0.4r_1$$

$$h_3 = 0.1r_3 + 0.4r_2 + 0.5r_1$$

$$h_4 = 0.1r_4 + 0.4r_3 + 0.5r_2 + 0.2r_1$$

$$h_5 = 0.1r_5 + 0.4r_4 + 0.5r_3 + 0.2r_2$$

$$h_6 = 0.1r_6 + 0.4r_5 + 0.5r_4 + 0.2r_3$$

A one day precipitation has just a little impact on the river height on that day. Its fundamental impact goes ahead the after quite a while and the day after that. Its impact on the following day is lower, and following 4 days it has no impact.

1.6.4 Systems of Linear Equations

Consider a set of m linear equations in n variables or unknowns $x_1, \ldots, x_n$

$$\begin{cases} A_{11}x_1 + A_{12}x_2 + \ldots + A_{1n}x_n = b_1 \\ A_{21}x_1 + A_{22}x_2 + \ldots + A_{2n}x_n = b_2 \\ \vdots \\ A_{n1}x_1 + A_{n2}x_2 + \ldots + A_{mn}x_n = b_m \end{cases}$$

Where the numbers A_{ij} are called the coefficients in the linear equations.These equations can be written in matrix notation

$$Ax = b$$

where A in an $m \times n$ coefficient matrix and b is an m-vector. An n-vector x is the solution of the linear equations if $Ax = b$ holds.

Rank of a Matrix

The rank of a matrix A corresponds to the maximum number of linearly independent columns of A. This is a measure of non-degeneracy of the system of linear equations and linear transformations.

Example: the following matrix has rank 2. The first two rows are linearly independent, making the row rank 2. However, all the three rows are linearly dependent (subtracting the second row from the first gives the third row).

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

The column rank of a matrix is the size of the largest subset of columns of A that constitutes a linearly independent set.

Orthogonal Matrices

Two vectors $\mathbf{x}, \mathbf{y}$ are orthogonal if $\mathbf{x^T y} = \mathbf{0}$. A square matrix U $R^{n \times n}$ is orthogonal if all its columns are orthogonal to each other.

Derminants

The determinant is a scalar value that can be computed from the elements of a square matrix and encodes certain properties of the linear transformation described by the matrix. The determinant of a matrix A is denoted by $det(A)$ or $|A|$. A square 2×2 matrix has a determinant as

$$Det(A) = Det(\begin{pmatrix} a & b \\ c & d \end{pmatrix}) = ad - bc$$

and a 3×3 matrix has a determinant:

$$Det(A) = Det(\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & k \end{pmatrix}) = a(ek - fh) + (-1)b(dk - fg) + c(dh - eg)$$

Geometrically, it can be expressed as the volume scaling factor of the linear transformation described by the matrix. This is also the volume of the n-dimensional parallelepiped spanned by the column and row vectors. We could show a 2-dimensional determinant as a parallelogram. The value of the determinant corresponds to the area of the parallelogram.

A general determinant of matrix A can be written as

$$det(A) = \sum_{i=1}^{k} a_{ij} C_{ij}$$

with no summation over j and where C_{ij} is the cofactor of a_{ij} defined by

$$C_{ij} = (-1)^{i+j} M_{ij}$$

where M_{ij} is the minor of matrix A formed by eliminating row i and column j.from A. Determinant is distributive, meaning that

$$|AB| = |A||B|$$

This means that the determinant of a matrix inverse is

$$|I| = |AA^{-1}| = |A||A^{-1}| = 1$$

where I is the identity matrix. Therefore

$$|A| = I/|A^{-1}|$$

Matrix Inverse

The inverse of a square matrix is denoted as A^{-1} and is a unique matrix such that $A^{-1}A = I$ where I is the unitary matrix. Non-square matrices do not have an inverse. A square matrix only has an inverse if its determinant is not zero.

Question 1.8: Inverse of a matrix

Given matrix A:

$$\text{A}= \begin{bmatrix} 3.4 & -1.2 & 2.7 \\ 2.3 & -2.4 & 1.1 \\ 3.1 & -1.8 & 2.5 \end{bmatrix}$$

1) Find determinant of matrix A.
2) Find inverse of matrix A.

Solution:

(1) det(A)=$\begin{vmatrix} 3.4 & -1.2 & 2.7 \\ 2.3 & -2.4 & 1.1 \\ 3.1 & -1.8 & 2.5 \end{vmatrix}$=3.4[(-2.4)(2.5)-(1.1)(-1.8)]+

1.2[(2.3)(2.5)-(1.1)(3.1)]+2.7[(2.3)(-1.8)-(-2.4)(3.1)]=-1.95

2) $A^{-1} = \frac{adjA}{|A|}$

Where $|A|$ is calculated from part 1, and $adjA$ is adjoint of matrix A.

$$adjA = \begin{bmatrix} -4.02 & -1.86 & 5.16 \\ -2.34 & 0.13 & 2.47 \\ 3.3 & 2.4 & -5.4 \end{bmatrix}$$

$$A^{-1} = \frac{1}{-1.95} \begin{bmatrix} -4.02 & -1.86 & 5.16 \\ -2.34 & 0.13 & 2.47 \\ 3.3 & 2.4 & -5.4 \end{bmatrix}.$$

1.6.5 Eigenvalue and Eigenvectors

An eigenvector of a linear transformation is a non-zero vector that changes by a scalar factor when that linear transformation is applied to it. The corresponding eigenvalue is the factor by which the eigenvector is scaled. Geometrically, an eigenvector corresponding to a real non-zero eigenvalue, points in a direction in which it is stretched by the transformation and the eigenvalue is the factor by which it is stretched. If the eigenvalue is negative, the direction is reversed. The prefix "eigen" is the German word for "proper" or "characteristic".

Given the square matrix $A \in R^{n \times n}$, $\lambda \in C$ is an eigenvalue of A and $x \in C^n$ is the corresponding eigenvector if

$$Ax = \lambda x \quad x \neq 0$$

This means multiplying A by a vector x results in a new vector that has the same direction as x but scales by a factor λ. The matrix A is a square matrix and x is a column vector. The mapping

is a result of matrix multiplication in the left and scaling of the column vector in the right side of the equation. An eigenvector corresponding to a real non-zero eigenvalue points in a direction that is stretched by the transformation and the eigenvalue is the factor by which it is stretched.

The linear transformations could take many forms. For example, it could be a differential operator d/dx in which case the eigenvectors are called eigenfunctions that are scaled by the differential operator

$$\frac{d}{dx}(e^{\lambda x}) = \lambda\ e^{\lambda x}$$

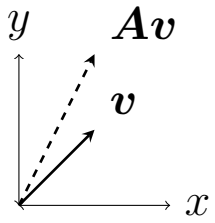
Alternatively, the differential operator could take the form of a square matrix, as shown above.

1.6.6 Properties of eigenvalues and eigenvectors

Consider matrix A= $\begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$ and vector $v = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.

$$\mathrm{A}v = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 6 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

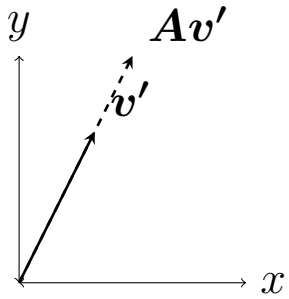
The resulting vector is a rotation of the original vector, scaled by 3. This is therefore not the eigenvector of matrix A.



Now consider vector $v' = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$.

$$\mathrm{A}v' = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 5 \\ 10 \end{bmatrix} = 5 \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Now, the resulting vector is the initial vector shifted by scale 5. This is the eigenvector of matrix A.



Question 1.9: Eigenvalues and Eigenvectors of a matrix

Find the eigenvalues and eigenvectors of A given below.

$$A = \begin{pmatrix} 2 & -2 \\ 1 & 4 \end{pmatrix}$$

Solution:

To find the eigenvalues solve $det(A - \lambda I) = 0$:

$$\begin{vmatrix} 2-\lambda & -2 \\ 1 & 4-\lambda \end{vmatrix} = 0$$

$$(2\text{-}\lambda)(4\text{-}\lambda)+2=0$$

$$\lambda^2 - 6\lambda + 10 = 0$$

$$\lambda_1 = 3 + i \ , \ \lambda_2 = 3 - i$$

To find eigenvector corresponding to the eigenvalue λ, solve $(A - \lambda I)X = 0$.

When $\lambda_1 = 3 + i$:

$$A = \begin{pmatrix} 2 - (3+i) & -2 \\ 1 & 4 - (3+i) \end{pmatrix} X = 0$$

Considering X=$\begin{pmatrix} x \\ y \end{pmatrix}$:

$$x = (i - 1)y$$

So, corresponding eigenvector is $\begin{pmatrix} i-1 \\ 1 \end{pmatrix}$. Similarly, for $\lambda_1 = 3 - i$, we get $\begin{pmatrix} -i-1 \\ 1 \end{pmatrix}$.

1.6.7 Markov chains

Markov chains are type of random processes undergoing a set of discrete and memoryless transitions within a given state space. A Markov chain can be specified by all possible states (S) and a transition matrix (P). The elements of transition matrix, P_{ij}, represent the probability of transition from state S_i to state S_j after one time step. let the initial probability vector, $I = (I_1, I_2, \ldots, I_n)$ denote the probabilities of being in states $S = (S_1, S_2, \ldots, S_n)$ at any given step ($\sum_n I_n = 1$). Then the probability vector for the state after one step is $I^{(1)} = PI$. Similarly, the probability vector after m steps will be $I^{(m)} = P^m I$. In the large number of steps, the probability of being in each state approaches a limit value given by the corresponding probability vector of the steady state w. The steady state w satisfies $Pw = w$. Intuitively, w is the probability vector such that applying transition matrix on that (Pw) returns the same probability vector (w). It is an eigenvector of P with eigenvalue 1. In the following we will solve two real-world problems which can be modeled with Markov chain.

1.6.8 PageRank from Matrix Perspective

PageRank algorithm is invented by Larry Page and Sergei Brin, founders of Google, to rank importance of web pages in search engine results. Imagine that we are surfing the web randomly. Suppose that at any given time, we are on a certain webpage, and we click a random link on that page to another page. If we

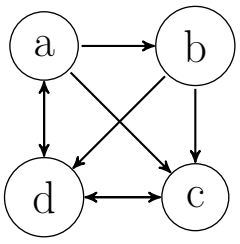
continue in this manner, then our surfing habits can be modeled by a Markov chain. Indeed, the probability of our traveling from P_i to P_j is just $\frac{1}{|P_i|}$, where $|P_i|$ is the number of links on webpage P_i. Hence, if we let $H = (p_{ij})_{i,j}^n$ be the matrix where

$$p_{ij} = \begin{cases} \frac{1}{|P_i|} & \text{if } P_i \text{ links to } P_j, \\ 0 & \text{otherwise,} \end{cases}$$

then the Markov chain with transition matrix H will model our surfing behavior.

The most important pages will be the one which we spend the most time on. H has a unique steady state vector $w = (w_1, w_2, \ldots, w_n)$, and we will spend w_i of our time on page P_i. Therefore, we can let w_i be the rank $r(P_i)$ of page P_i.

As a simple example, consider the following links between webpages: $a \to b, a \to c, a \to d, b \to c, b \to d, c \to d, d \to a, d \to c$, as depicted in the graph below.



So, the transition matrix is:

$$\mathrm{H} = \begin{bmatrix} 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \end{bmatrix}$$

Elements in each row of matrix H show all the possible links between web pages and all (on the same row) add to 1. Solving $(Hw = w)$,we find the eigenvector of H for a eigenvalue of 1 is $w = (\frac{6}{29}, \frac{2}{29}, \frac{9}{29}, \frac{12}{29})$. Therefore, the most important page is page d, followed by c, a, and b.

1.6.9 Market share of technology companies

Financial analysis of rise and decline of three technology companies show that the monthly market shares of three companies A, B and C can be estimated by a transformation matrix P:

$$\mathrm{P} = \begin{bmatrix} 0.8 & 0.1 & 0.1 \\ 0.03 & 0.95 & 0.02 \\ 0.2 & 0.05 & 0.75 \end{bmatrix}$$

The first row of matrix P represents the share of Company A that will pass to Company A, Company B and Company C respectively. The second row represents the share of Company B that will pass to Company A, Company B and Company C respectively, while the third row represents the share of Company C that will pass to Company A, Company B and Company C respectively. Notice each row adds to 1. Let's find the ultimate market share of the 3 companies A, B and C. As we discussed in 1.6.7 the ultimate

market share can be calculated with the eigenvector of P matrix for a given eigenvalue of 1. The eigenvector of P can be find by eigenvector of its transpose ($Pw = w \rightarrow w = P^T w$). Solving equation for P gives us the trivial solutions since each row of P adds to 1, however solving equation $P^T w = w$ will provide the ultimate share.

$$\begin{bmatrix} 0.8 & 0.03 & 0.2 \\ 0.1 & 0.95 & 0.05 \\ 0.1 & 0.02 & 0.75 \end{bmatrix} \begin{bmatrix} \mathrm{w}_1 \\ \mathrm{w}_2 \\ \mathrm{w}_3 \end{bmatrix} = \begin{bmatrix} \mathrm{w}_1 \\ \mathrm{w}_2 \\ \mathrm{w}_3 \end{bmatrix}$$

which gives a set of equations as follows:

$$-0.2w_1 + 0.03w_2 + 0.2w_3 = 0$$

$$0.1w_1 - 0.05w_2 + 0.05w_3 = 0$$

$$0.1w_1 + 0.02w_2 - 0.25w_3 = 0$$

So, the corresponding eignvector is:

$$w = \begin{bmatrix} 23.7116 & 61.8564 & 14.4332 \end{bmatrix}$$

The ultimate share are $23.7\%, 61.9\%$ and 14.4% for company A, B and C, respectively.

1.7 Hessian Matrix

The Hessian matrix of a multivariate function $f(x_1, x_2, x_3, \ldots, x_n)$ is defined as a square matrix with its elements being second derivatives of the function

$$\text{Hf(x)} = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}$$

The Hessian matrix helps to find the maximum or minimum of a multivariate function. For example, if $f(x_1, x_2, x_3, \ldots, x_n)$ function represents distance, then Hessian's i^{th} row and j^{th} column entry tells us about the rate of change of velocity's i^{th} component (ie. acceleration) along the j^{th} direction. The Hessian matrix is also used for feature detection as it provides the gradient of an image in different directions. For a stable feature the curvature across a feature point should be high in more than one direction. A feature with change in only one direction is unstable as it's impossible to determine where along an edge it is located. It is used to find if the change in gradient is large in more than one direction.

2 Statistical Background

2.1 Populations and Samples

In statistics, we often rely on a sample — that is, a small subset of a larger set of data — to draw inferences about the larger set. The larger set is known as the population from which the sample is drawn. A sample is typically a small subset of the population.

2.1.1 Simple Random Sampling:

Such sampling requires every member of the population to have an equal chance of being selected into the sample. In addition, the selection of one member must be independent of the selection of every other member. That is, picking one member from the population must not increase or decrease the probability of picking any other member (relative to the others). In this sense, we can say that simple random sampling chooses a sample by pure chance. Random samples, especially if the sample size is small, are not necessarily representative of the entire population.

Random Assignments: This is random division of the sample into two groups. Random assignment is critical for the validity of an experiment.

Stratified Random Sampling: In stratified random sampling the population is divided into a number of subgroups (or strata). Random samples are then taken from each subgroup with sample sizes proportional to the size of the subgroup in the population. For instance, if a population contained equal numbers of men and women, and the variable of interest is suspected to vary by gender, one might conduct stratified random sampling to insure a representative sample

2.1.2 Sample Size:

Recall that the definition of a random sample is a sample in which every member of the population has an equal chance of being selected. This means that the sampling procedure rather than the results of the procedure define what it means for a sample to be random. Random samples, especially if the sample size is small, are not necessarily representative of the entire population.

Variables: There are dependent and independent variables. For example, if you want to correlate the number of hours students study as a function of the success rate of the students. The number of hours is the independent variable and the success in the exam is the dependent variable.

2.2 Discrete or Continuous variables:

values such as the number of children in kindergarten are discrete variables- e.g. there are 3, 4 or 5 children but not 3.2 children. The change in temperature is a continuous variable e.g. 24.4 or 23.2 deg C.

We employ probabilistic models for the behavior of our sample data, and infer from the data accordingly- hence the name, statistical inference. The most powerful use of statistics is prediction. These days it is known as machine learning.

2.2.1 Expected values

Consider a repeatable experiment with random variable, X. The expected value of X is the long-run average value of X as we repeat the experiment infinitely.

Example:

- Take a 20 questions multiple choice test with A, B, C, D as the answers. If you answer all "A", you expect to get 25% right (5 out of 20). This is calculated as follows: The probability, P, of getting a question right if you guess is 0.25
  The number of questions in the test, n, is 20
  $P \times n = 0.25 \times 20 = 5$
  This is an expected value for a binomial random variable. It is binomial because there are only two outcomes: you get the answer right or you get the answer wrong. The basic expected value $E(x)$ is the probability of an event $P(x)$ multiplied by the number of times that event happens, X. That is

$$E(x) = P(x) * X$$

  .

  For example, if you toss a coin 10 times, the probability of getting a head in each trial is 0.5. Therefore, the expected value (the number of heads you could expect to get in 10 coin tosses) is: $E(x) = P(x) * X = 0.5 * 10 = 5$.

Question 2.1: Breaking a rod at random

A rod of length $2l$ is broken into two parts at a point whose position is random, in the sense that the point is equally likely to be anywhere on the rod. Let X be the length of the smaller part. Write down the probability density function (PDF) of X, and find the expectation of X.

Solution:

The break-point is uniformly distributed along the rod, so that the length X of the smaller part is itself uniformly distributed on the interval $[0,\, l]$. Its probability density function is thus:

$$f_X(x) = \begin{cases} 1/l, & 0 \leq x \leq l, \\ 0, & otherwise. \end{cases}$$

The expectation of X can be written down directly as $\frac{1}{2}l$, because of the symmetry of the PDF $f_X(x)$. Alternatively, it can be derived from

$$E(X) = \int_0^l x f_X(x) dx = \left[\frac{x^2}{2l}\right]_0^l = \frac{1}{2}l$$

2.2.2 Random Variables

Random variables quantify the outcome of a random process. Suppose we roll two dice, with X and Y referring to the number of dots we get on the blue and yellow dice respectively. Then X and Y are random variables as they are numerical outcomes of the experiment. Moreover, $X + Y$, $2XY$, $\sin(XY)$ are also random variables.

2.2.3 Discrete Random Variables

Consider the dice example. The random variable X could take on six values in the set $1, 2, 3, 4, 5, 6$. It is said that the *support of* X is $1, 2, 3, 4, 5, 6$, meaning the list of the values the random variables can take on. Now consider we toss a coin until we get a head. Let N be the number of tosses needed. Then the support of N is the set $[1, 2, 3 \dots]$. This is an infinite set. Now consider throwing a dart at the interval $(0, 1)$ and the place that it hits, R, can take on any of the values between 0 and 1. Here the support is an uncountable infinite set. We say that X, X_1, X_2 and N are discrete random variables while R is continuous.

2.2.4 Independent Random Variables

Two random variables are independent if events corresponding to them are independent. In the dice example, it is clear that the random variables X and Y do not affect each other (if I know $X = 6$, that knowledge would not help me to guess Y). The probability

of $Y = 2$ when knowing X is still 1/6. This is mathematically written as

$$P(Y = 2|X = 6) = P(Y = 2)$$

or

$$P(Y = 2 \text{ and } X = 6) = P(Y = 2)P(X = 6)$$

In other words, the events $X = 6$ and $Y = 2$ are independent and similarly $X = i$ and $Y = j$ are independent for any i and j values.

2.2.5 Expected Value for Multiple Events

The formula for calculating the expected value when there are multiple probabilities is

$$E(X) = \sum XP(X)$$

.

The equation is the same as before but adds the sum of the probabilities. Let K_{in} be the number of times the value i occurs among $X_1, X_2, \ldots, X_n$; $i = 1, \ldots, 10$ and $n = 1, 2, 3, \ldots$. For instance, $K_{4,20}$ is the number of times we get 4 heads in the first 20 repetitions of our experiment. Then

$$E(X) = \lim_{n\to\infty} \frac{X_1 + \cdots + X_n}{n}$$

$$= \lim_{n\to\infty} \frac{0 \times K_{0n} + 1 \times K_{1n} + 2 \times K_{2n} + \cdots + 10 \times K_{10n}}{n}$$

$$= \sum_{i=0}^{10} i \times \lim_{n\to\infty} \frac{K_{in}}{n}$$

Properties of the Expectation values

1. The expected value of a discrete random variable X which has support A is

   $$E(X) = \sum_{c\in A} c\ P(X = c)$$

   $E(X)$ amounts to the weighted sum of the values in the support of X, with the weights being the probabilities of those values.

2. For any random variables U and V, the expected value of a new random variable $D = U + V$ is the sum of the expected values of U and V:

   $$E(U + V) = E(U) + E(V)$$

3. For any random variable U and constant a, then

   $$E(aU) = aE(U)$$

Here aU is a new random variable defined in terms of an old one.

4. For random variables X and Y - not necessarily independent- and constants a and b, we have

$$E(aX + bY) = a\ E(X) + b\ E(Y)$$

By induction, for constants $a_1, \ldots, a_k$ and random variables $X_1, \ldots, X_k$, form the new random variables $a_1X_1 + \cdots + a_kX_k$, then

$$E(a_1X_1 + \cdots + a_kX_k) = a_1E(X_1) + \cdots + a_kE(X_k)$$

For any constant, b, we have

$$E(b) = b$$

5. The expected value of the function $g(X)$ is

$$E[g(X)] = \sum_{c \in A} g(c).P(X = c)$$

where the sum is over all values c that can be taken on by X.

6. If U and V are independent, then

$$E(UV) = E(U)E(V)$$

2.3 Sampling Distributions

Random Samples A random sample is defined as a set of objects that are chosen randomly. In other words, this is a sequence of independent and identically distributed (IID) random variables. In other words, the terms random sample and IID are basically the same . In statistics we often refer to this "random sample" but in probability it is more commonly referred to as IID. Therefore,

- Identically distributed means that there are no overall trends-the distribution does not fluctuate and all items in the sample are taken from the same probability distribution.
- Independent means that the sample items are all independent events. In other words, they are not connected to each other in any way.

A more technical definition of an IID statistics is that random variables $X_1, X_2, \ldots, X_n$ are IID if they share the same probability distribution and are independent events. Sharing the same probability distribution means that if you plot all the variables together, they would resemble some kind of distribution: a uniform, normal, or any other kind of distributions. Consider a sample of n random variables $X_1, X_2, \ldots X_n$. Since they are IID, each variable X_i has the same mean and variance. Random variables that are identically distributed do not necessarily have to have the same probability. As an example, a flipped coin can be modeled by a binomial distribution and generally has a 50% chance of a head or

a tail. Now, let's say the coin is weighted so that the probability of heads was 49.5% and tails was 50.5%. Although the coin flips are IID, they do not have equal probabilities.

The variables $X_1, X_2, \ldots, X_n$ form a random sample of size n from a population if the X_i are IID and their common distribution is that of the population.

2.3.1 The Sample Mean

The mean of a sample of random variables $X_1, X_2, \ldots, X_n$, denoted by $\bar{X}$, is defined as

$$\bar{X} = \frac{X_1 + X_2 + \cdots + X_n}{n}$$

$\bar{X}$ itself is a random variable.

2.3.2 Expected Value and Variance of $\bar{X}$

Let μ be the population mean. Since X_i is distributed as in the population, the expected value of X_i, $E(X_i)$, is the same as the mean: $E(X_i) = \mu$.

$$E(\bar{X}) = E(\frac{1}{n}\sum_{i=1}^{n} X_i) = \frac{1}{n}E(\sum_{i=1}^{n} X_i) = \frac{1}{n}(\sum_{i=1}^{n} E\ X_i) = \frac{1}{n}\ n\ \mu$$

The variance of $\bar{X}$ is $1/n$ times the population variance:

$$Var\bar{X} = Var[\frac{1}{n}\sum_{i=1}^{n} X_i] = \frac{1}{n^2}Var[\sum_{i=1}^{n} X_i] = \frac{1}{n^2}\, n\sigma^2 = \frac{1}{n}\sigma^2$$

where we used $Var(cU) = c^2 Var(U)$ and the additive property of variance for independent random variables ($Var(U+V) = Var(U) + Var(V)$).

For large values of n, $\bar{X}$ is pretty accurate and is close to the population mean, μ. In statistics we often ask the question, if the variance of our estimator is small enough.

Estimation of σ^2

The sample analog of μ is $\bar{X}$. What about the sample analog of the Expectation values? Since $E()$ averages over the whole population of Xs, the sample analog is to average over the sample. So, our sample analog is

$$s^2 = \frac{1}{n}\sum (X_i - \bar{X})^2$$

which is the population variance. Square root of the population variance is the standard deviation.

2.3.3 Variance

The expected value indicates the average value a random variable takes on. The dispersion around that expected value provides an

estimate of how well the expected value is representative of the random variables.

For a random variable U, for which the expected value is $E(U)$, the variance, $Var(U)$, is defined as

$$Var(U) = E[(U–EU)^2]$$

The square root of the variance is called the *standard deviation*. The variance gives a measure of the dispersion. In the above expression if the values of U are mostly clustered near its mean, then $(U - EU)^2$ will be small and hence, the variance of U would be small too. If there is a wide variation in U, the variance would be large.

Question 2.2: Mean and variance of a random variable

Find the mean and variance of the continuous random variable X with probability density function given by:

$$f_X(x) = \begin{cases} 3x^{-4}, & x \geq l, \\ 0, & otherwise. \end{cases}$$

Solution:

By definition,

$$E(X) = \int_{-\infty}^{\infty} x f_X(x) dx = 3 \int_1^{\infty} x^{-3} dx = \frac{3}{2}$$

and

$$Var(X) = E(X^2) - \{E(X)\}^2 = 3\int_1^\infty x^{-2}dx - \frac{9}{4} = \frac{3}{4}$$

Properties of Variance

1. For any random variable U and constant c,

$$Var(cU) = c^2 Var(U)$$

This makes sense. Imagine we multiply any random variable by 5. Then, its average squared distance to its mean should increase by a factor of 25. Now let's prove the above relation. Define $V = cU$, then

$$Var(V) = E[(V - EV)^2]$$

$$= E[cU - E(cU)]^2 = E[cU - cE(U)]^2 = c^2 E[U - E(U)]^2 = c^2 Var(U)$$

2. For any constant d,

$$Var(U + d) = Var(U)$$

This is because the variance of a constant is always zero (i.e. a constant never varies)

Chebychev's Inequality

For a random variable X with mean μ and variance σ^2

$$P(|X - \mu| \geq c\sigma) \leq \frac{1}{c^2}$$

In other words, X strays more than, for example, 3 standard deviations from its mean at most only 1/9 of the time.

Example:

The professor mentions that anyone scoring more than 1.5 standard deviations above mean earns an "A" grade, while those with scores under 2.1 standard deviations below the mean get an "F". Out of 200 students in the class, how many got either "A" or "F"? Take $c = 2.1$ in Chebychev. It indicates that at most $1/2.1^2 = 0.23$ of the students were in the "Fail" category. This means about 46 of them.

Coefficient of Variation

Any measure of the size of $Var(X)$ should relate to the size of $E(X)$. This leads to the definition of the *coefficient of variation* which is defined as the ratio of the standard deviation to the mean:

$$coef.\ \ of\ var = \frac{\sqrt{Var(X)}}{EX}$$

This is a scale-free measure and indicates whether a variance is large or not.

2.3.4 Covariance

A measure of the degree to which U and V vary together is their covariance

$$Cov(U, V) = E[(U - E(U))(V - E(V))]$$

Suppose, for instance, U is large relative to its expected value and at the same time V is small relative to its expected value. For example, consider the price of an item. Stores charging a higher price U, will sell fewer of them than V and vice versa.

Suppose we have U as the human height and V as weight. These are usually large together or small together. Therefore, the covariance here is positive. Often those with height greater than the expected value, have weights greater than the expected value. This means that both terms $E(U-E(U))$ and $E(V-E(V))$ have the same signs and therefore the covariance is positive. Covariance can also be referred to as "correlation".

Properties of Covariance:

1. $Cov(U + V) = E(UV)$–$EU.EV$
2. $Var(U + V) = Var(U) + Var(V) + 2Cov(U, V)$
3. $Var(aU + bV) = a^2Var(U) + b^2Var(V) + 2abCov(U, V)$ for any constants a and b.

4. If U and V are independent, then $Cov(U, V) = 0$. In this case $Var(U + V) = Var(U) + Var(V)$

5. $Var(a_1X_1+\cdots+a_kX_k) = \sum_{i=1}^{k} a_i^2 Var(X_i)+2\sum a_i a_j Cov(X_i, X_j)$. If X_i are independent, then we have $Var(a_1X_1 + \cdots + a_kX_k) = \sum_{i=1}^{k} a_i^2 Var(X_i)$.

Probability Distribution Function (PDF) Characteristics:

PDF provides the maximum information on the behavior of a random variable or the physical quantities it represents. It is often required to express these in terms of a set of chosen properties or functions. Several characterizations are needed to express the data and their distribution as discussed in the following.

Median: The median is the value separating the higher half from the lower half of the data or a population. For a dataset, it is the middle value. The advantage of the median over mean is that it is not skewed by a small proportion of extremely large or extremely small data. For example, median income is a more accurate representation of the income of a community.

To measure the median, we first set the data in the increasing or decreasing order. If the number of data points (observations), n, is odd, median is the value at position $(n+1)/2$. If the number of observations, n, is even, we first find the value at position $(n/2)$. Then find the value at position $(n + 1)/2$. The average of these

two values is the median.

Examples: For numbers 1,3,3,6,7,8,9 the median is 6. For numbers 1,2,3,4,5,6,8,9 the median is $(4 + 5)/2 = 4.5$

Mode: The mode of a data distribution is the value that appears most often. If X is a discrete random variable, the mode is the value x at which the probability mass distribution takes its maximum value. Mode is the value that is most likely to be sampled. If all the numbers occur the same number of times, there is no mode on the distribution. There is more than one mode if more than one number is the most frequent.

Example: For numbers 6,3,9,6,6,5,9,3 the first mode is 6. The second mode is 3 and 9.

Skewness: Skewness measures the deviation from symmetry in a normal distribution. It measures the extent to which a distribution varies from a normal distribution. A normal distribution has zero skewness. Pearson's first and second coefficients of skewness are the most commonly used measures of the skewness.

- Pearson mode skewness: $(\mu$–mode$)/\sigma$

- Pearson's 1st skewness coefficient: $(3\mu$–mode$)/\sigma$

- Pearson's 2nd skewness coefficient: $(3\mu\text{–median})/\sigma$

Kurtosis: While skewness measures extreme values in one tail of the distribution with respect to the other, the kurtosis measures extreme values in either tail. Distributions with large kurtosis mean data exceeding the tails of the normal distribution (five or more standard deviations from the mean).

Question 2.3: Distribution of examination marks

The following table shows the number of candidates who scored $0, 1, \ldots, 10$ marks for a particular question in an examination.

Mark	0	1	2	3	4	5	6	7	8	9	10
No. of candidates	8	10	49	112	98	86	54	37	28	12	6

Calculate the mean, median and mode of the distribution of marks. What feature of the distribution is suggested by the fact that the mean is greater than the median?

Solution:

The mean is the straightforward average of the $8 + 10 + \cdots + 12 + 6 = 500$ marks. Hence the mean is

$$\frac{(8\times10)+(10\times1)+\cdots+(6\times10)}{500} = \frac{2241}{500} = 4.48$$

The median of n values, put in order of magnitude, is the middle value. It is the value ranked in position $(n+1)/2$ if n is odd and is the average of the two middle observations, i.e. those ranked $(n/2)$ and $(n/2+1)$, when n is even. In this problem, $n = 500$ and so the median is the average of the 250th and 251st in order of magnitude. These are both 4, so the median is also 4.
The mode is that value which occurs with the greatest frequency, so is clearly 3 here.
The mean, 4.48, is greater than the median, 4, which suggests that the distribution is positively skewed, that is, the values to the right of the median are more spread out than are those on the left. This cannot affect the median, clearly, but the extra spread on the right causes the mean to be greater.

Statistical Bias

Suppose we wish to estimate some population quantity θ, using an estimator $\hat{\theta}$ computed from our sample data. $\hat{\theta}$ is called unbiased if

$$E\hat{\theta} = \theta$$

Otherwise it is biased with the amount of the bias

$$E\hat{\theta} - \theta$$

It can be shown that

$$E(s^2) = \frac{n-1}{n}\sigma^2$$

If we were to take many samples, the average of all of our s^2 values would be slightly smaller than σ^2. As a result, statisticians decided to divide by $n-1$ to make the sample variance an unbiased estimator of σ^2. The definition of s^2 became

$$s^2 = \frac{1}{(n-1)} \Sigma_{i=1}^{n} (X_i - \bar{X})^2$$

Parametric Distributions

The idea of the distribution of a random variable is central in statistics and probability. Let U be a discrete random variable. The distribution of U is the support of U, together with the associated probabilities.

Examples:

- Let X denote the number of dots one gets when rolling a die. The value of X are then $1, 2, 3, 4, 5, 6$, each with probability $1/6$. Therefore

$$\text{distribution of } X = (1, 1/6), (2, 1/6), (3, 1/6), \\ (4, 1/6), (5, 1/6), 6, 1/6)$$

- Let X taking values 1 and 2 with probabilities 0.48 and 0.52 respectively. Therefore,

$$\text{distribution of } X = (1, 0.48), (2, 0.52)$$

The probability mass function of a discrete random variable V, denoted as p_v, is: $p_v(k) = P(V = k)$, for any value k in the support of V.

Distributions based on Bernoulli Trials

A sequence $B_1, B_2, \ldots$ of independent indicator variables with $P(B_i = 1) = p$ for all i is called a sequence of *Bernoulli trials*. The event $B_i = 1$ is called success, with 0 being termed failure.

Example: Number of trials needed to obtain the first success. Imagine in tossing a coin, we want to estimate the number of trials needed to get the first head, with N being the number of tosses needed. In order for this to take k tosses, we need $k-1$ tails and then a head. Thus

$$p_N(k) = (1 - \frac{1}{2})^{k-1}.\frac{1}{2}, \quad k = 1, 2, \ldots$$

We say that N has geometric distribution with $p = 1/2$. Here we define, for example, head a "success" and tail a "failure". This of course does not mean anything and only is for the purpose of demonstration.

Now, define M to be the number of rolls of a die needed until number 5 shows up. Then

$$P_M(k) = (1–\frac{1}{6})^{k-1}.\frac{1}{6} \quad k = 1, 2 \ldots$$

reflecting the fact that event $M = K$ occurs if we get $k-1$ non-5s and then a 5. Here "success" is getting a 5. We say that N has a geometric distribution with $p = 1/6$.

In general, suppose the random variable w is defined to be the number of trials needed to get a success in a sequence of Bernoulli trials. Then

$$p_w(k) = (1–p)^{k-1}.p \quad k = 1, 2 \ldots$$

There is a different distribution for each value of p. This is therefore called a *parametric family of distributions* indexed by the parameter p.

The Binomial Family of Distributions

We discussed Bernoulli trials with parameter p, with a variable number of trials (N) but a fixed number of successes (1). A binomial distribution arises when we have the opposite situation- a fixed number of Bernoulli trials (n) but a variable number of successes (i.e. X).

The binomial distribution with parameters n and p is the discrete probability distribution of the number of successes in a sequence of n independent experiments, each having a yes/no question, and each with its own success (with probability p) and failure (probability $q = 1 - p$). A single success/failure experiment (when $n = 1$) is called a Bernoulli trial. The binomial distribution is

frequently used to model the number of successes in a sample of size n drawn with replacement from a population of size N. If the sampling is done without replacement, the draws are not independent and so, the resulting distribution is a hypergeometric distribution, not a binomial one. For N much larger than n, the binomial distribution would be a good approximation.

Question 2.4: Sampling incoming batches

A company taking delivery of a large batch of manufactured articles accepts the batch if either (a) a random sample of 6 articles from the batch contains not more than one defective article, or (b) a random sample of 6 contains two defective articles, and a second random sample of 6 is taken, and found to contain no defectives. If 20% of the articles in the batch are actually defective, what is the probability that the company will accept the delivered batch?

Solution:

We denote the numbers of defectives in the first and second samples by X_1 and X_2 respectively. These random variables are independent, and both have the binomial distribution with index n= 6 and parameter p= 0.2, where p is the probability that an item is defective. A batch will be accepted if $X_1 \leq 1$ or if $X_1 = 2$ and $X_2 = 0$. Thus,

$$Pr(\text{batch is accepted}) = \Pr\{(\mathrm{X}_1 \leq 1) \cup (\mathrm{X}_1 = 2 \ \cap \ \mathrm{X}_2 = 0)\}$$

$$= Pr(X_1 \leq 1) + Pr(X_1 = 2 \ \cap \ X_2 = 0)$$

By the addition law of probability for mutually exclusive events. Now,

$$Pr(X_1 \leq 1) = Pr(X_1 = 0) + Pr(X_1 = 1)$$
$$= (0.8)^6 + \{6 \times (0.2) \times (0.8)^5\} = 0.6554$$

We also obtain, by independence of X_1 and X_2,

$$Pr(X_1 = 2 \ \cap \ X_2 = 0) = [\binom{6}{2} \times (0.2)^2 \times (0.8)^4] \times (0.8)^6$$

$$Pr(X_1 = 2 \ \cap \ X_2 = 0) = 0.0644$$

Therefore,

$$Pr(\text{batch is accepted}) = 0.6553 + 0.0644 = 0.720$$

Probability Mass Function

If the random variable X follows the binomial distribution with parameters $n \in N$ and $p \in [0, 1]$, we write $X \sim B(n, p)$. The probability of getting exactly k successes in n independent Bernoulli trials is given by the probability mass function. The formulation can be understood as follows: k successes occur with probability p^k and $n - k$ failures occur with probability $(1 - p)^{n-k}$. However, the k successes can occur anywhere among the n trials and there

are different ways of distributing k successes in a sequence of n trials given by:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Where $n!$ is n factorial corresponding to $n! = n \times (n-1) \times (n-2) \cdots \times 2 \times 1$. The probability mass distribution is then

$$f(k, n, p) = P_f(k; n, p) = P_f(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

where $k = 0, 1, 2, \ldots$.

Example: We toss a coin five times and let X be the number of heads we get. We say that X is binomially distributed with parameters $n = 5$ and $p = 1/2$. Let's find $P(X = 2)$. There are many orders that this could happen such as: HHTTT, TTHHTT, HTTHT etc. Each order has probability $0.5^2(1-0.5)^3$ and there are $\binom{2}{5}$ orders. Thus

$$P(X = 2) = \binom{2}{5} 0.5^2 (1-0.5)^3 = 52/32 = 5/16$$

The Poisson Family of Distributions

This expresses the probability of a given number of events occurring in a fixed interval of time or space if these events occur with a known constant mean rate and independently of the time since

the last event. The Poisson family is often used to model count data.

Examples:

If you go to a certain bank every day and count the number of customers who arrive between 11:00 and 11:15am, you will probably find that approximated by a Poisson distribution.

Suppose an individual keeping track of the amount of mail they receive each day may notice that they receive an average number of 4 letters per day. If receiving any particular piece of mail does not affect the arrival times of future pieces of mail, i.e., if pieces of mail from a wide range of sources arrive independently of one another, then a reasonable assumption is that the number of pieces of mail received in a day obeys a Poisson distribution

2.3.5 Probability Mass Function

A discrete random variable X is said to have a Poisson distribution with parameter $\lambda > 0$, if, for $k = 0, 1, 2, \ldots$, the probability mass function of X is given by

$$f(k; \lambda) = Pr(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

where $k!$ is the factorial of k taking values $k = 0, 1, \ldots$. The positive real number λ is equal to the expected value of X indicating

the average number of the events per interval, with its variance corresponding to

$$\lambda = E(X) = Var(X)$$

Question 2.5: The telephone exchange

A telephone exchange receives, on average, 5 calls per minute. Find the probability that in a 1-minute period no calls are received.

Solution:

We assume that the number of incoming calls in one minute has the Poisson distribution with mean 5. Hence, if X is the number of incoming calls in one minute,

$$Pr(X = 0) = e^{-5}\ \frac{5^0}{0!} = e^{-5} = 0.007$$

The Power Law Family of Distributions

This is expressed as

$$p_x(k) = c\ k^{-\gamma} \quad k = 1, 2, 3 \ldots$$

It is required that $\gamma > 1$ as otherwise the sum of the probabilities will be infinite. To satisfy that condition, the value of c can be determined by setting the sum to 1.

$$1.0 = \sum_{k=1}^{\infty} ck^{-\gamma} = c \int_{1}^{\infty} k^{-\gamma} dk = c/(\gamma–1)$$

Therefore $c = \gamma - 1$.

Continuous Probability Models

There are two types of random variables: discrete random variable and continuous random variables. This is defined in the following example: Suppose we throw a dart at random at the interval [0,1]. Let D denote the spot we hit. By "at random" we mean that all the points are equally likely to be hit. Also, all sub-intervals of equal length are equally likely to get hit. For instance, the probability of the dart landing in (0.7,0.8) in the same as for (0.1,0.2) or (0.556,0.556) as all have length 0.1.

We call D a continuous random variable because its support is a continuum of points, in this case, the entire interval [0,1]. Because of this randomness

$$P(u \leq D \leq v) = u–v$$

for any case $0 \leq u \leq v \leq 1$.

Here we note that individual values have probability zero

$$P(D = c) = 0$$

for any individual point c. Consider the case $c = 0.3$. Then

$$P(D = 0.3) \leq P(0.29 \leq D \leq 0.31) = 0.02$$

so, $P(D = 0.3) \leq 0.02$. However, we can also replace 0.29 and 0.31 in the above equation by 0.299 and 0.301 and get $P(D = 0.3) \leq 0.002$ and continue this. The point is that $P(D = 0.3)$ must be smaller than any positive number. Therefore, it is zero.

Cumulative Distribution Function

For any random variable W, its cumulative distribution function (cdf), F_w, is defined by

$$F_w(t) = P(W \leq t) \quad -\infty < t < \infty$$

where t is an argument to a function. Consider the random dart example above. For $t = 0.23$

$$F_D(0.23) = P(D \leq 0.23) = P(0 \leq D \leq 0.23) = 0.23$$

2.3.6 Density Function

The density function for continuous random variables is similar to the probability mass function for discrete random variables. For a discrete random variable, its cdf can be calculated by summing its pmf

$$F_z(t) = \sum pz(j)$$

Consider a continuous random variable W. Define

$$f_w(t) = \frac{d}{dt} F_W(t) \quad -\infty < t < \infty$$

wherever the derivative exists. The function f_w is called the probability density function (pdf) or just the density of W.

Notation: the lower case f denotes a density with a subscript consisting of the name of a random variable.

Properties of Density Functions

$$P(a < W < b) = F_W(b) – F_W(a) = \int_a^b f_w(t)dt$$

where $F_W(b)$ is all the probability accumulated from $-\infty$ to b while $F_W(a)$ is all the probability accumulated from $-\infty$ to a.

$$\int_{-\infty}^{\infty} f_W(t)dt = 1$$

Note that in this integral $f_W(t)$ will be zero in various ranges of t corresponding to values that W cannot take on. Density is a non-negative function that integrates to 1.

The expected value of the probability density function is

$$E(W) = \int t\ f_W(t)\ dt$$

where t ranges over the support of W such as the interval [0,1]. Also, we have

$$E(W^2) = \int_{-\infty}^{\infty} t^2\ f_W(t)dt$$

and for a function, $g(t)$

$$E[g(W)] = \int_t g(t) f_W(t) dt$$

Example:

Consider X with a density function equal to $2t/15$ on the interval [1,4] and zero everywhere else. Therefore,

$$EX = \int_1^4 t.2t/15 dt = 2.8$$

$$P(X > 2.5) = \int_{2.5}^4 2t/15 dt = 0.65$$

$$F_X(s) = \int_1^s 2t/15 dt = (s^2 - 1)/15$$

Mean of $1/t$ is

$$E(1/t) = \int_1^4 \frac{1}{t}.2t/15 dt = 2/5$$

Question 2.6: Finding a probability density function

A continuous random variable X, with mean unity, has probability density function $f_X(x)$ given by

$$\begin{cases} a(b-x)^2, & 0 \leq x \leq b, \\ 0, & otherwise. \end{cases}$$

Find the values of a and b.

Solution:

In order to solve for the two unknown values a and b , we need two equations involving a and b. These equations are obtained from the two items of information that we are given, as follows:

The function $f_X(x)$ is a probability density function. Hence,

$$\int_0^b a(b-x)^2 dx = 1$$

and so,

$$\left[\frac{-a(b-x)^3}{3}\right]_0^b = 1$$

resulting in the equation $ab^3 = 3$.

The mean of X is unity. So,

$$\int_0^b a(b-x)^2 x dx = 1$$

and thus,

$$a\int_0^b (b-x)^2(x-b+b)dx = 1$$

Splitting terms in the second bracket gives:

$$-a\int_0^b (b-x)^3 dx + ab\int_0^b (b-x)^2 dx = 1$$

and these integrals are evaluated as above to yield,

$$-\frac{ab^4}{4} + \frac{ab^4}{3} = 1.$$

which simplifies to give $ab^4 = 12$. This is to be solved together with the equation $ab^3 = 3$ obtained earlier. We now see that $b = 4$ and $a = \frac{3}{64}$.

Parametric Families of Continuous Distributions

In the dart example above, we can throw the dart at the interval (q, r). For a uniform distribution with all the points having equal probability, the density must be constant in that interval. It should also integrate to one. Therefore, the density is one divided by the interval:

$$f_D(t) = \frac{1}{(r-q)}$$

for $t \in (q, r)$ and zero elsewhere.

The Normal (Gaussian) Distribution

The density for a normal distribution is

$$f_W(t) = \frac{1}{\sqrt{2\pi\sigma}e^{-0.5(\frac{(t-\mu)}{\sigma})^2}} \quad -\infty < t < \infty$$

This is a two-parameter family indexed by parameters μ and σ which are the mean and standard deviation respectively.

Central Limit Theorem

The central limit theorem states that if $X_1, \ldots, X_n$ are independent and of the same distribution, then the new random variable $Y = X_i, \ldots, X_n$ has an approximately normal distribution.

The Exponential Distribution

The density in this family has the form

$$f_W(t) = \lambda e^{-\lambda t} \quad 0 < t < \infty$$

.

After integration we find that $E(W) = 1/\lambda$ and $Var(W) = \frac{1}{\lambda^2}$.

2.3.7 Confidence Intervals

Suppose we have a random sample $X_1, \ldots, X_n$ from some population with mean μ and variance σ^2. The function

$$Z = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}}$$

has an approximate normal distribution with $N(0, 1)$.

Now suppose we have a random sample $X_1, \ldots, X_n$ from some population with mean μ and variance σ^2. We will be interested in the central 95% of the distribution $N(0, 1)$. Due to symmetry, the distribution has 2.5% of its area in the left tail and 2.5% of the area in the right tail. By consulting an $N(0, 1)$ table we find that the cut-off points are -1.95 and 1.95. In other words if some random variable T has a $N(0, 1)$ distribution, then $-1.96 < T < 1.96) = 95\%$. Therefore

$$0.95 = P(-1.96 < \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} < 1.96)$$

This becomes

$$0.95 = P(-z < Z < z) = P(\bar{X}\text{–}1.96\sigma/\sqrt{n} < \mu < \bar{X}\text{+}1.96\sigma/\sqrt{n})$$

.

The lower point of the 95% confidence interval of μ is $(\bar{X}\text{–}1.96\sigma/\sqrt{n}$ and the upper point is: $(\bar{X} + 1.96\sigma/\sqrt{n})$. If the standard deviation of the population σ is known in this case, the distribution of the sample mean $\bar{X}$ is a normal distribution with μ the only unknown parameter. If the standard deviation is not known, this turns to the student t-distribution.

Percentile

How to estimate the percentiles: We aim to measure 25th percentile for the 8 numbers in this Table. Numbers are given ranks from 1 to 8.

1. Compute the rank R for 25th percentile: $R = P/100 \times (N+1)$ Where P is the percentile (25) and N the total number 8. $R = 25/100 \times (8+1) = 9/4 = 2.25$ If R is an integer, the P^{th} percentile is the number with rank R. Where R is not an integer, we compute the P^{th} percentile with interpolation

2. Define the integer part of $R(IR)$. Here it is $IR = 2$. Define the fractional part of $r(FR)$. Here it is $FT = 0.25$.

3. Find the score with rank IR and IR+1. From the table this is the score with rank 2 and with rank 3. These are 5 and 7

4. Interpolate by multiplying the difference between the scores by FR and add the result to the lower score $(0.25)(7-5)+5 = 5.5$ - The 25th percentile is 5.5.

2.3.8 Student's t-Distribution

Consider $X_1, \ldots, X_n$ to be an independent sample from a normally distributed population with unknown parameters mean μ and variance σ^2. Let

$$\bar{X} = (X_1 + \cdots + X_n)/n$$

Number	Rank
3	1
5	2
7	3
8	4
9	5
11	6
13	7
15	8

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2$$

where $\bar{X}$ is the sample mean and S^2 the sample variance. Then

$$\frac{T = \bar{X} - \mu}{S/\sqrt{n}}$$

has a Student's t-distribution with $n-1$ degrees of freedom. The distribution of T does not depend on the values of the unobservable parameters μ and σ^2.

We now calculate the 95% confidence interval for μ. Then considering c as the 97.5th percentile of this population

$$P(-c < T < c) = 0.95$$

there is 2.5% chance that T would be less than $-c$ 2.5% chance that it will be larger than c. We now have the 95

$$\bar{X}–cS/\sqrt{n} < \mu < \bar{X} + cS/\sqrt{n})$$

After calculating $\bar{X}$ and s, we can calculate the confidence intervals.

Question 2.7: Peeling potatoes faster

An experiment was conducted to compare the performance of two potato peelers and in particular to discover whether the typical user might be able to peel potatoes faster with one rather than the other. Ten volunteers were used, and each was given both peelers for a period before the experiment in order to gain familiarity with them. In the experiment the volunteer subjects used both peelers on standardised amounts of potatoes, and then repeated the experiment with the peelers used in the opposite order, so as to eliminate any effect due to ordering. The table below gives, for each subject, the mean of the natural logarithms of time (in seconds) needed to complete the tasks.

Subject	Peeler A	Peeler B
1	2.33	2.34
2	2.76	2.79
3	1.91	1.91
4	2.62	2.60
5	2.01	2.03
6	1.77	1.80
7	1.81	1.81
8	1.99	2.00
9	1.97	1.98
10	2.26	2.30

Use Student's t-test, at the 5% level of significance, to test whether the peelers differ in their efficiency.

Solution:

For data of this type a paired sample t-test is required. We therefore form the 10 differences $d_1, \ldots, d_{10}$ between the results for peelers A and B:

$$-0.01, -0.03, 0.00, 0.02, -0.02, -0.03, 0.00, -0.01, -0.01, -0.04.$$

Treating them as a random sample from $N(\mu, \sigma^2)$, we test the null hypothesis $\mu = 0$ (with σ^2 unknown). In a natural notation, we thus compare

$$t = \frac{\bar{d}-0}{s/\sqrt{n}}$$

with the two-tailed 5% points of Student's t-distribution on 9 degrees of freedom, i.e. with ± 2.262. Since $\bar{d} = -0.013$ and $s = 0.0177$, the value of t is -2.327. We thus reject the null hypothesis, and conclude that the peelers differ in their mean level of efficiency.

Multivariate Statistics

Scientific problems are rarely limited to measurements of a single (random) variable. Modern experiments involve large numbers of simultaneous variables. A fraction of these variables may be correlated but some may be independent from one another. However, it is not known a priori as which variables are statistically correlated and which are independent. Therefore, we need to formulate the notion of multivariate (multiple variables) probability densities that encompass all measured variables.

Probability Distribution

A probability distribution is a mathematical function that provides the probabilities of occurrence of different possible outcomes in an experiment. The probability distribution is a description of a random phenomenon in terms of the probabilities of events. For example, if random variable X denotes the probability of a coin toss, the probability distribution of X would take the value 0.5 for X=heads and 0.5 for X tails. Probability distributions are divided

into two classes: Discrete and Continuous. Discrete probability distribution is when the set of outcomes are discrete- like tossing a coin. This can be encoded by a discrete list of the probabilities of the outcomes known as a probability mass function. Continuous probability distribution is applicable where the set of the possible outcomes can take on values in a continuous range- like the temperature in a given day. This is described by probability density function (PDF). A probability distribution whose sample is one-dimensional (i.e. real numbers) is called univariate while a distribution whose sample is a vector space is called multivariate. A univariate distribution gives the probability of a single random variable. A multivariate distribution (a joint probability distribution) gives the probability of a random vector- a list of two or more random vector- a list of two or more random variables taking on various combinations of values.

2.4 Joint Probability Density Function

Let's consider a system involving two continuous random observables. Let X represent the hypothesis that the first observable lies in the range $[x, x + dx]$ and Y is the second observable lying in the range $[y + dy]$, given some prior information about the system, I. The conjunction of the two hypothesis X, Y is true if both hypothesis are true. The quantity $p(X, Y|I)$ then expresses the degree that the hypothesis X and Y might be true jointly. Given that X and Y are continuous, the conjunction X, Y is also a continuous hypothesis. In this case $p(X, Y|I)$ (probability of X and

Y if I is true) corresponds to a joint probability density function

$$p(X, Y|I) = lim_{\Delta x, \Delta y 0} \frac{p(x \leq Xx + \Delta x, y \leq Yy + \Delta y)}{\Delta x \Delta y}$$

This expresses the probability that variables X and Y are found jointly in the intervals $[x, x + dx]$ and $[y, y + dy]$ respectively.

One could also consider alternative hypothesis that the observables X and Y are found in other intervals. Summing the probabilities of hypothesis spanning the entire hypothesis space, yields unity. One therefore gets the normalization condition

$$\int \int_H p(X, Y|I) dX dY = 1$$

The figure shows a probability density function of two variables, x and y in the form of an iso-contour plot (the boundaries between different shades of grey delineate loci of equal probability densities. While x and y are both random variables with Gaussian distributions, their values are not independent of one another. Large values of x are accompanied by small values of y and vice versa. The two variables are then said to be *correlated*.

Example 1: Consider the flip of two fair coins. Let A and B be discrete random variables associated with the outcome of the first and second coins respectively. Each coin flip is a Bernoulli trial and has a Bernoulli distribution (either 1 or 0 outcome). If

a coin displays "heads" then the associated random variable takes the value 1, and it takes the value 0 otherwise. The probability of each of these outcomes is . Therefore, the marginal (unconditional) probability density functions are $P(A) =$ and $P(B) = 1/2$ for $A \in 0, 1$ and $B \in 0, 1$ respectively.

The joint probability density function of A and B defines probabilities for each pair of out come with all outcomes as

$$(A = 0, B = 0), (A = 0, B = 1), (A = 1, B = 0), (A = 1, B = 1)$$

Since each outcome is equally likely, the joint probability density function becomes

$$P(A, B) = 1/4 \quad for \quad A, B \in 0, 1$$

since the coin flips are independent, the joint probability density function is the product of the two marginal

$$P(A, B) = P(A)P(B) \quad for A, B \in \ 0, 1$$

Example 2: Consider the roll of a fair dice and let $A = 1$ if the number is even (2, 4 or 6) and $A = 0$ otherwise. Furthermore, let $B = 1$ if the number is prime (2, 3, or 5) and $B = 0$ otherwise. The joint distribution of A and B is demonstrated in the following table, with their probability mass function (since the distributions are discrete) as

$$P(A=0,B=0)=P1=1/6 \quad P(A=1,B=0)=P4,6=2/6$$
$$P(A=0,B=1)=P3,5=2/6 \quad P(A=1,B=1)=P2=1/6$$

These probabilities sum to 1 since the probability of some combination of A and B occurring is 1.

2.4.1 Joint Cumulative Distribution Function

For a pair of random variables X, Y, the joint cumulative distribution function $F_{X,Y}$ is

$$F_{X,Y} = P(X \leq x, Y \leq y)$$

where the right hand side represents the probability that the random variable X takes on a value less than or equal to x and Y takes on a value less than or equal to y.

Discrete case

The joint probability mass function of two discrete random variables X, Y is:

$$p_{X,Y}(x,y) = P(X = x \text{ and } Y = y)$$

or written in terms of the conditional distributions

$$\begin{aligned} p_{X,Y}(x,y) &= P(Y=y|X=x).P(X=x) \\ &= P(X=x|Y=y).P(Y=y) \end{aligned}$$

where $P(Y = y|X = x)$ is the probability of $Y = y$ given that $X = x$.

the generalization of this is the joint probability distribution of n discrete random variables $X_1, \ldots, X_n$ which is

$$p_{X_1,\ldots,X_n} = P(X_1 = x_1 \text{ and } \ldots \text{ and } X_n = x_n)$$

or

$$\begin{aligned} p_{X_1,\ldots,X_n}(x_1, \ldots, x_n) = P(X_1 = x_1) \times P(X_2 = x_2|X_1 = x_1) \\ \times P(X_3 = x_3|X_1 = x_1, X_2 = x_2) \ldots \\ \times P(X_n = x_n|X_1 = x_1, X_2 = x_{2n-1} = x_{n-1}) \end{aligned}$$

Continuous case

The joint probability density function $f_{X,Y}(x, y)$ for two continuous random variables is defined as the derivative of the joint cumulative distribution function

$$\frac{\partial^2 F_{X,Y}(x, y)}{\partial x \partial y}$$

which equals to

$$f_{X,Y}(x, y) = f_{Y|X}(y|x) f_X(x) = f_{X|Y}(x|y) f_Y(y)$$

where $f_{Y|X}(y|x)$ and $f_{X|Y}(x|y)$ are the conditional distributions of Y given $X = x$ and of X given $Y = y$ respectively, with $f(x)$

and $f(y)$ being the marginal distributions for X and Y respectively. This definition can be extended to more than two random variables. Again, the sum of all the probabilities is one.

2.4.2 Marginal Distributions

The marginal distribution of a subset of a collection of random variables is the probability distribution of the variables contained in the subset. It gives the probability of various values of the variables in the subset without reference to the other variables. This contrasts with a conditional distribution, which gives the probabilities contingent upon the values of the other variables.

Consider PDF $p(X, Y|I)$. One could derive the probability density $p(X|I)$ corresponding to the probability that the hypothesis X is true irrespective of other hypothesis. It is possible to show that $p(X|I)$ is obtained by integration of $p(X, Y|I)$ over all hypothesis Y. This operation is referred to as marginalization.

For discrete case, marginal probability mass function can therefore be expressed as

$$p(X|I) = \sum_{i=1}^{n} p(X, Y_i|I)$$

and

$$p(Y|I) = \sum_{j=1}^{n} p(X_j, Y|I)$$

This can also be written as probability density function

$$p(X|I)dx = (\int p(X,Y|I)dy)dx$$

or simply

$$p(X|I) = \int p(X,Y|I)dy$$

The probability density function $p(X|I)$ is said to be the marginal probability density of $p(X,Y|I)$ or, one can say that $p(X,Y|I)$ has been marginalized or that, the "uninteresting" parameter Y has been eliminated by marginalization.

2.4.3 Conditional Probability

Let's first define the concept of conditional probability. Consider a sample space S, with subsets A and B such that $p(B) \neq 0$. One then defines the conditional probability, $p(A|B)$, as the probability of A given B:

$$p(A|B) = \frac{p(A \cap B)}{p(B)}$$

This corresponds to the probability of observing the random variable X within A when it is also within B. In fact, the probability of $p(A)$ can itself be viewed as a conditional probability $p(A) = p(A|S)$ since $p(S) = 1$ by construction.

The two subsets A and B and the measurement outcomes they

represent, are said to be independent if they satisfy the condition

$$p(A \cap B) = p(A)p(B)$$

This means that the probability that X is a member of A and B simultaneously is equal to the product of the probabilities of X being in A and B independently. This allows the evaluation of conditional probability

$$p(A|B) = \frac{p(A \cap B)}{p(B)} = \frac{p(A)p(B)}{p(B)} = p(A)$$

when A and B are statistically independent, the conditional probability of A given B is equal to the probability of A itself- in other words, the probability of A doesn't depend on B. Similarly

$$p(B|A) = \frac{p(A \cap B)}{p(A)} = \frac{p(A)p(B)}{p(A)} = p(B)$$

Now, to establish the relation between the conditional probabilities $p(A|B)$ and $p(B|A)$ - which are not independent, we consider that by definition of the conditional probability, one has

$$p(A|B) = \frac{p(A \cap B)}{p(B)}$$

and similarly

$$p(B|A) = \frac{p(B \cap A)}{p(A)}$$

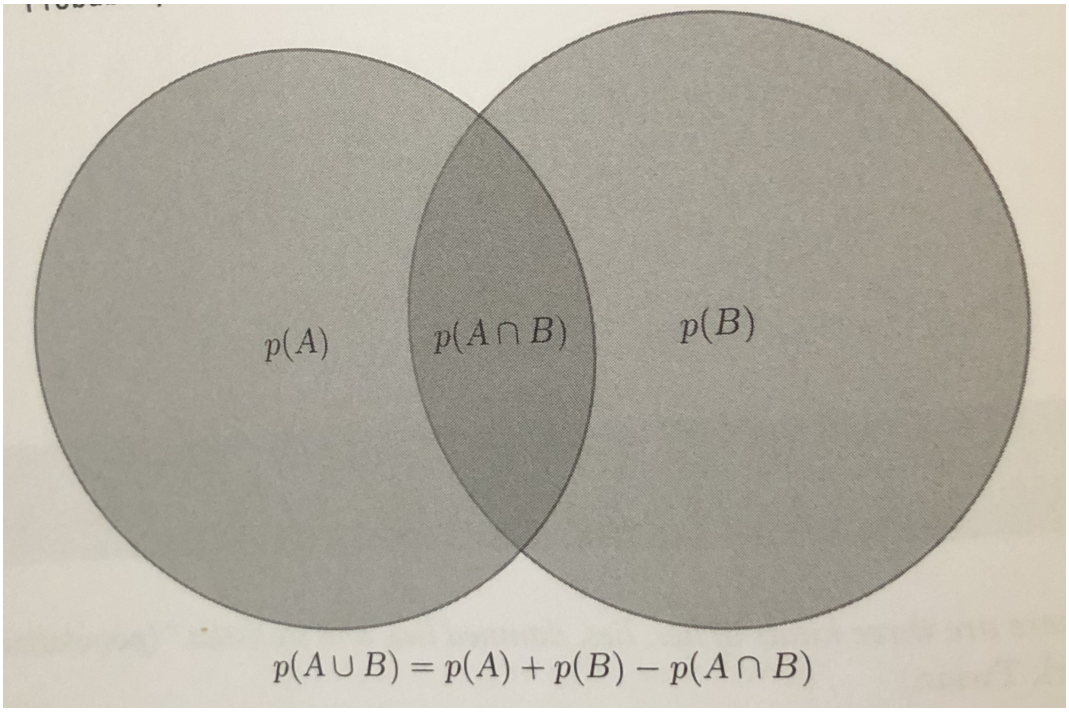
Given that the intersection of two sets: $B \cap A = A \cap B$ are commutative, the two conditional probabilities are related as follows

$$p(A \cap B) = p(B|A)p(A) = p(A|B)p(B)$$

provided neither $p(A)$ or $p(B)$ are null. This implies that it is possible to calculate conditional probability $p(B|A)$ as follows:

$$p(B|A) = \frac{p(A|B)p(B)}{p(A)}$$

This is known as Bayes theorem and has significant application in data science.



Conditional Probability Density Function

Given the joint PDF, $p(X, Y|I)$, and the marginal PDFs $p(X|I)$ and $p(Y|I)$, it is also of interest to evaluate the probability density, $p(X|Y, I)$, corresponding to the probability density that X is true when Y is known to be true. Since X and Y are continuous hypothesis, this amounts to the conditional probability density for X to be in the interval $[x, x + dx]$ given that Y is known to be in $[y, y + dy]$. Applying the product rule we have

$$p(X, Y|I) = p(Y|I)p(X|Y, I)$$

Rearranging this we get

$$p(X|Y, I) = p(X, Y|I)/p((Y|I) = p(X, Y|I)/\int_H p(X, Y|I)dx$$

where we used marginalization relation $p(Y|I) = \int_H p(X, Y|I)dx$. The above relation is the Conditional Probability Density Function.

Question 2.8: The school assembly

At the morning assembly, five schoolchildren - Alan, Barbara, Clare, Daniel and Edward- sit down in a row along with five other children (whose names need not concern us here). If the children arrange themselves at random, find the probability that Alan and Barbara sit together.

Solution:

If we consider the position of A, there are two different situations: when A is in an end position, and when A is in an internal position. The required probability is thus:

$$\begin{gathered}\Pr(\text{A in end position})\ \Pr(\text{B next to A}|\text{A in end position}) \\ +\Pr(\text{A in internal position})\ \Pr(\text{B next to A}|\text{A in internal position}) \\ = (\tfrac{2}{10} \times \tfrac{1}{9}) + (\tfrac{8}{10} \times \tfrac{2}{9}) = \tfrac{1}{5}\end{gathered}$$

The conditional probabilities are obtained by regarding the nine possible positions for B, given that of A, as equally likely.

Covariance of Two Variables

The covariance of two random variables measures the degree to which the variables are correlated or covarying. Let's consider the expectation value of products $x_i x_j$ for $i, j = 1, \ldots, n$ and $i \neq j$. We define the covariance of variables x_i and x_j, $Cov[x_i, x_j]$ as

$$Cov[x_i, x_j] = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} (x_i - \mu_i)(x_j - \mu_j) f(x) dx_1 \ldots dx_n$$

where μ_i and μ_j are the mean values of x_i and x_j respectively. Now, considering the covariane of two variables, x_1 and x_2 with μ_1 and μ_2 being their mean respectively, we have

$$Cov[x_1, x_2] = \int (x_1 x_2 – x_1\mu_2 - \mu_1 x_2 + \mu_1\mu_2) f(x_1, x_2) dx_1 dx_2$$

Splitting the terms of the integrand, we get

$$\begin{aligned} Cov[x_1, x_2] = & \int x_1 x_2 f(x_1, x_2) dx_1 dx_2 \\ & -\mu_2 \int x_1 f(x_1, x_2) dx_1 dx_2 \\ & -\mu_1 \int x_2 f(x_1, x_2) dx_1 dx_2 \\ & +\mu_1\mu_2 \int f(x_1, x_2) dx_1 dx_2 \end{aligned}$$

The integrals of second and third terms are μ_1 and μ_2 respectively, while the integral of last term is unity because of normalization of $f(x_1, x_2)$. Therefore we have

$$Cov[x_1, x_2] = \int x_1 x_2 f(x_1, x_2) dx_1 dx_2 - \mu_1\mu_2$$

A null covariance ($Cov[x_1, x_2) = 0$) indicates variables x_1 and x_2 are statistically independent. In this case we can write

$$\int x_1 x_2 f(x_1, x_2) dx_1 dx_2 - \mu_1\mu_2 = \int x_1 f(x_1) dx_1 \int x_2 f(x_2) dx_2$$

The interpretation of covariance can be shown with the joint probability densities of two random variables X and Y defined as the product of two Gaussian functions

$$p(x,y) = 1/\sqrt{2\pi}exp[-(x-\mu_x)^2/2\sigma_x^2] \times 1/\sqrt{2\pi}exp[-(y-\mu_y)^2/2\sigma_y^2]$$

with means μ_x and μ_y and standard deviations σ_x and σ_y respectively. The distribution in Figure shows cases for (a). $\sigma_x = \sigma_y$; (b). $\sigma_x > \sigma_y$; (c) $\sigma_x < \sigma_y$.

Mean Vectors and Covariance Matrices

Multivariate analysis is used to compute mean vector and the variance-covariance matrix. Let's demonstrate this with the following example.

Consider the following matrix:

$$X = \begin{bmatrix} 4.0 & 2.0 & 0.60 \\ 3.9 & 2.1 & 0.59 \\ 4.3 & 2.0 & 0.58 \\ 4.1 & 2.2 & 0.63 \end{bmatrix}$$

The matrix shows the set of 4 observations measuring 3 variables and can be described by its mean vector and variance-covariance matrix. The three variables from left to right are length, width

and height of a certain object. For example, each row vector X_i is another observation of the three variables (or components).

The mean vector consists of the means of each variable and the variance-covariance matrix consists of the variances of the variables and the covariances between each pair of variables in the other matrix positions.

The formula for computing the covariance of the variables X and Y is

$$COV = \frac{\Sigma_{i=1}^{n}(X_i - \bar{x})(Y_i - \bar{y})}{n-1}$$

with $\bar{x}$ and $\bar{y}$ denoting the means of X and Y respectively.

The mean vector contains the arithmetic means of the three variables and the variance-covariance matrix, S, is calculated by

$$S = \frac{\Sigma_{i=1}^{n}(X_i - \bar{X})(X_i - \bar{X})'}{n-1}$$

where $n = 4$ for this example.

$$\bar{x} = [4.10 \quad 2.08 \quad 0.604]$$

$$S = \begin{bmatrix} 0.025 & 0.0075 & 0.00175 \\ 0.0075 & 0.0070 & 0.00135 \\ 0.00175 & 0.00135 & 0.00043 \end{bmatrix}$$

In the S matrix, 0.025 is the variance of the length variable, 0.0075 is the covariance between the length and the width variables, 0.00175 is the covariance between the length and the height variables. 0.007 is the variance of the width variable, 0.00135 is the covariance between the width and height variables and 0.00043 is the variance of the height variable (the diagonal elements of the above matrix are all variances while other matrix elements are the covariance values.

2.5 Regression

Regression is a case of the general model fitting . It is defined as the relation between a dependent variable, y, and a set of independent variables, x, that describes the expectation value of y given x: $E[y|x]$. Techniques used in regression make a number of simplifying assumptions regarding the nature of the data, measurement uncertainties and complexity of the models.

If we have a model for the conditional distribution described by parameter θ, we can write the function as $y = f(x|\theta)$ where y is a scalar dependent variable and x an independent vector. Consider four points on the (x, y) plane with a simple straight line model $y_i = \theta_0 + \theta_1 x_i$. Each point provides a joint constraint on θ_0 and θ_1. If there were no uncertainties in the variables, then this constraint would be a straight line on the (θ_0, θ_1) plane $\theta_0 = y_i - \theta_1 x_i$. As the number of points is increased, the uncertainties in the data will transform the constraints from a line to a distribution. The

best estimate of the model parameters is now given by a posterior distribution. This is the multiplication of different probability distributions (constraints) for all points. Priors are accommodated within this picture as additional multiplicative constraints applied to the likelihood distribution.

Now, when error behavior for the dependent variable is known, and errors for the independent variables are negligible, we can use the Bayesian methodology to estimate the posterior probability density function (pdf) for the model parameters

$$p(\theta|x_i, y_i, I) \propto p(x_i, y_i)|\theta, I)p(\theta, I)$$

Where the information I describes the error behavior for the dependent variable. The data likelihood is the product of likelihoods for the individual points and can be expressed as

$$p(y_i|x_i, \theta, I) = e(y_i|y)$$

Where $y = f(x|\theta)$ is the adapted model and $e(y_i|y)$ is the probability of observing y_i given the true value (or the model prediction) y. For example, if the y error distribution is Gaussian, with the width for $i - th$ data point given by σ_i, and negligible errors on x, then

$$p(y_i|x_i, \theta, I) = 1/(\sigma_i\sqrt{2\pi})\ exp(-[y_i - f(x_i|\theta)^2/2\sigma_i^2))$$

2.5.1 Regression for Linear Models

Consider an independent variable x and a dependent variable y, a linear model can then be expressed as

$$y_i = \theta_0 + \theta_1 x_i + \epsilon_i$$

where θ_0 and θ_1 are the coefficients that describe the regression function that we are trying to estimate (ie. The slope and intercept of a straight line) and ϵ_i represents an additive noise term. The assumption here is that the uncertainties in the independent variable are negligible

$$p(y_i|x_i, \theta, I) = \prod_{i=1}^{N} 1/(\sigma_i\sqrt{2\pi})\ exp(-[y_i–(\theta_0 + \theta_1 x_i)]^2/2\sigma_i^2)$$

For a flat pdf ($p(\theta|I)$) where we have no knowledge about the distribution of the parameter θ, the posterior will be directly proportional to the likelihood function. If we take the logarithm of the posterior, we arrive at the definition of regression

$$ln(L) = ln((\theta|x_i, y_i, I)) \propto \sum_{i=1}^{N} exp(-[y_i–(\theta_0 + \theta_1 x_i)]^2/2\sigma_i^2)$$

Maximizing the likelihood function as a function of model parameters θ, is acquired by minimizing the sum of the square errors. The form of this likelihood function arises by our assumption of a Gaussian distribution of uncertainties in the dependent variables. Minimization of the above equation leads to

$$\theta_1 = \frac{\sum_{i=1}^{N} x_i y_i - \overline{x} \;\; \overline{y}}{\sum_{i=1}^{N} (x_i - \overline{x})^2}$$

and

$$\theta_0 = \overline{y} - \theta_1 \overline{x}$$

Where $\overline{x}$ and $\overline{y}$ are the mean values of x and y respectively. The variance and standard deviation of this regression are

$$\sigma^2 = \sum_{i=1}^{N} (y_i - \theta_0 + \theta_1 x_i)^2$$

$$\sigma_{\theta_1}^2 = \frac{\sigma^2}{\sum_{i}^{N} (x_i - \overline{x})^2}$$

$$\sigma_{\theta_0}^2 = \sigma^2(\frac{1}{N} + \frac{\overline{x}^2}{\Sigma_i^N (x_i - \overline{x})^2})$$

2.5.2 Matrix Formulation of the Regression

Here we generalize regressions using matrix formulation. We define regression in terms of a matrix, M, such that

$$Y = M\theta$$

Where Y is an N dimensional vector of value y_1

$$\mathbf{Y} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \\ y_n \end{pmatrix} \tag{35}$$

For the straight-line regression function, θ is a two-dimensional vector of regression coefficients

$$\theta = \begin{pmatrix} \theta_0 \\ \theta_1 \end{pmatrix} \tag{36}$$

and M is a $2 \times N$ matrix

$$\mathbf{M} = \begin{pmatrix} 1 & x_0 \\ 1 & x_1 \\ 1 & x_2 \\ \cdot & \cdot \\ 1 & x_{N-1} \end{pmatrix} \tag{37}$$

where the constant value in the first column represents θ_0 term in the regression. If the errors are different for different data points, we define a covariance matrix, C, as an $N \times N$ matrix

$$\mathbf{C} = \begin{pmatrix} \sigma_0^2 & 0 & \cdot & 0 \\ 0 & \sigma_1^2 & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & \sigma_{N-1}^2 \end{pmatrix} \tag{38}$$

where the diagonal elements of this matrix contain the uncertainties σ_i associated with the dependent variable, Y. The maximum likelihood solution for this regression is

$$\theta = (M^T C^{-1} M)^{-1} (M^T C^{-1} Y)$$

which minimizes the sum of the square errors $(Y - \theta M)^T C^{-1} (Y - \theta M)$ as we did in the previous section. The uncertainties in the regression coefficients., θ, can now be expressed as the symmetric matrix

$$\Sigma_\theta = \begin{pmatrix} \sigma^2_{\theta_0} & \sigma^2_{\theta_0\theta_1} \\ \sigma^2_{\theta_0\theta_1} & \sigma^2_{\theta_1} \end{pmatrix} = [M^T C^{-1} M^{-1}] \tag{39}$$

Whether we have sufficient data to constrain the regression (i.e. sufficient degrees of freedom) depends on whether $M^T M$ is an invertible matrix.

2.5.3 Multivariate Regression

For multivariate data we fit a hyperplane rather than a straight line. In this case we extend the description of the regression function to multiple dimensions with $y = f(x|\theta)$ given by

$$y_i = \theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \ldots + \theta_k x_{ik} + \epsilon_i$$

where θ_i are the regression parameters and x_{ik} is the k^{th} component of the i^{th} data within a multivariate dataset. This naturally follows from the definition of the matrix

$$\mathbf{M} = \begin{pmatrix} 1 & x_{01} & x_{02} & \cdot & x_{0k} \\ 1 & x_{11} & x_{12} & \cdot & x_{1k} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & x_{N1} & x_{N2} & \cdot & x_{Nk} \end{pmatrix} \tag{40}$$

The regression coefficients are the estimates of θ. Their associated uncertainties are as before

$$\theta = (M^T C^{-1} M)^{-1} (M^T C^{-1} Y)$$

and

$$\Sigma_\theta = [M^T C^{-1} M^{-1}]$$

2.5.4 Least Squares Problem

Consider the $m \times n$ matrix A and the linear equation $Ax = b$, where b is an $m-$vector. Suppose the equation is over-determined, which means that the matrix A generates more equations (m) than there are variables (n) to be determined. The only way these equations would have a solution is when b is a linear combination of the columns of A.

For most choices of b however, there is no $n-$vector x for which $Ax = b$. therefore, we need to seek an x for which $r = Ax - b$, called the residual, is minimized. This means that we should choose x so that we minimize the norm of the residual $||Ax - b||$.

Minimizing the norm of the residual is the same as minimizing its square. Therefore, we could minimize

$$||Ax–b||^2 = ||r||^2 = r_1^2 + r_2^2 + \ldots + r_m^2$$

The problem of finding an $n-$vector $\hat{x}$ that minimizes $||Ax–b||^2$ over all possible choices of x, is called the least squares problem.

It is expressed by

$$minimize\ ||Ax–b||^2$$

The matrix A and the vector b are called the data (i.e. they are given) and the vector x is to be found. This is the linear least squares problem. The non-linear least square problem is when r is an arbitrary function of x.

Any vector $\hat{x}$ that satisfies $||a\hat{x}||^2 \leq ||Ax - b||^2$ for all x is a solution of the least square problem.

Solution:

To minimize $\hat{x}$ the function $f(x) = ||ax - b||^2$ must satisfy

$$\frac{\partial f}{\partial x_i} = 0 \quad i = 1, \ldots, n$$

which can be expressed in vector equation $\nabla f(\hat{x}) = 0$

where $\nabla f(\hat{x})$ is the gradient of f evaluated at $\hat{x}$. The gradient can be expressed in matrix form as

$$\nabla f(x) = 2A^T(ax–b)$$

This is derived by writing the least squares equation as follows:

$$f(x) = ||Ax–b||^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (A_{ij}x_j–b_j)^2$$

Now take the partial derivative of f with respect to x_k by differentiating the sum term by term

$$\nabla f(x)_k = \frac{\partial f}{\partial x_k}(x) = \sum_{i=1}^{m} 2 \sum_{j=1}^{n} (A_{ij} x_j – b_i)(A_{ik})$$

$= \sum_{i=1}^{m} 2(A^T)_{ki}(Ax – b)_i = (2A^T(Ax – b))_k$

which is the above formula. Now, any $\hat{x}$ that minimizes $||Ax - b||^2$ must satisfy

$$\nabla f(\hat{x}) = aA^T(A\hat{x} – b) = 0$$

which can be written as

$$A^T A\hat{x} = A^T b$$

The coefficient $A^T A$ is the Gram matrix associated with A. Its entries are inner products of columns of A. The assumption that columns of A are linearly independent implies that the Gram matrix $A^T A$ is invertible, which implies that

$$\hat{x} = (A^T A)^{-1} A^T b$$

is the only solution of the normal equations and the least squares problem. The matrix $(A^T A)^{-1} A^T b$ is the pseudo-inverse of the

matrix A and is denoted by $A^{\dagger}$. Therefore, this can be written as

$$\hat{x} = A^{\dagger}b$$

This is least square approximate solution. Note that there is a difference between this equation and equation $x = A^{-1}b$ which is the solution for a square set of linear equations. The linear equation and the inverse actually satisfies $ax = b$. However, the least squares approximation called linear equation $\hat{x} = A^{\dagger}b$ generally does not satisfy $A\hat{x} = b$.

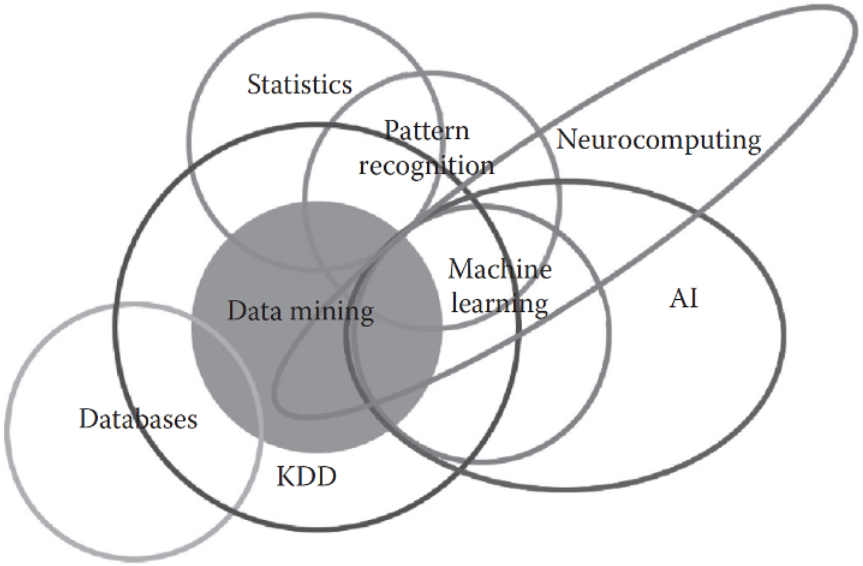
3 Machine learning

3.1 Definition

Machine learning turns data into information. It lies in the intersection between computer science, engineering and statistics. It can be applied to many fields. Machine learning is a branch of artificial intelligence that aims at enabling machines to perform their jobs skillfully by using intelligent software. The statistical learning methods constitute the backbone of intelligent software that is used to develop machine intelligence. Hal Varian, chief economist at Google once said: "The ability to take data, to be able to understand it, to process it, to extract value from it, to visualize it, to communicate it, it is a hugely important skill. Now we have essentially free data. So the complementary scarce factor is the ability to understand that data and extract value from it." Machine learning teaches computers to learn from the experience. ML algorithms use computational methods to learn information directly from data without relying on a predetermined equation as a model. The algorithms adaptively improve their performance as the number of samples available for learning increases. ML algorithms find natural patterns in data that generate insight and help you make better decisions and predictions.

In the article entitled "Discipline of Machine Learning" Tom Mitchel defined ML as: "Machine Learning is a natural outgrowth of the intersection of Computer Science and Statistics. We might say the defining question of Computer Science is 'How can we build

machines that solve problems, and which problems are inherently tractable/intractable?' The question that largely defines Statistics is 'What can be inferred from data plus a set of modeling assumptions, with what reliability?' The defining question for Machine Learning builds on both, but it is a distinct question. Whereas Computer Science has focused primarily on how to manually program computers, Machine Learning focuses on the question of how to get computers to program themselves (from experience plus some initial structure). Whereas Statistics has focused primarily on what conclusions can be inferred from data, Machine Learning incorporates additional questions about what computational architectures and algorithms can be used to most effectively capture, store, index, retrieve and merge these data, how multiple learning subtasks can be orchestrated in a larger system, and questions of computational tractability."



3.2 Terminology

Before discussing ML, it is helpful to review the terminology often used. We present this with an example. In Table 1 there are some values for four parts of various birds that we decided to measure. We measured the weight, wingspan, whether it has webbed feet and the color of its back. In reality, one would like to measure more than this. The four things we have measured are called features or attributes. Each of the rows in this Table is an instance made up of features. The first two features in Table 1 are numeric. The third feature is binary (only taking "yes" or "no" values. The fourth feature (back color) is an enumeration over color table we are using.

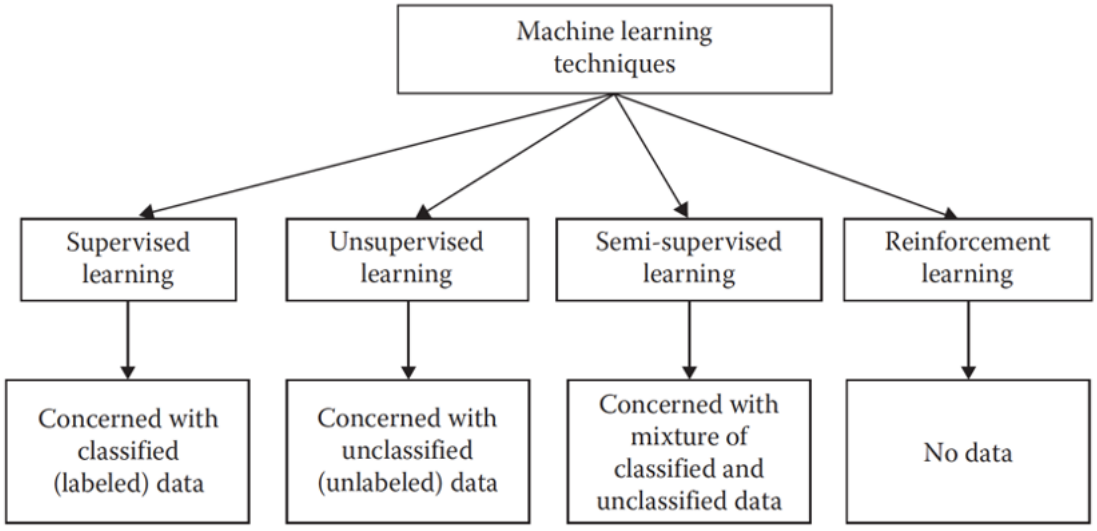
One task in ML is classification. Suppose we see a bird, identifying the type of bird given its features, is called classification. There are different types of classification in ML. Now, suppose we decide on a ML algorithm to use for classification. We need to train the algorithm. In order to train an algorithm, we give it a set of data, called training set. A training set is a set of training examples we will use to train our ML algorithm. In Table 1 our training set has six training examples. Each training example has four features and one target variable (the last column in Table 1. The target variable is what we are trying to predict with our ML algorithm (i.e. the species here). In classification the target variable takes a nominal value and in the task of regression, its value is continuous. In a training set the target variable is known.

Machine learns by finding some relationship between the features and the target variable. In classification the target variables are called classes and it is assumed to be a finite number of classes.

To test ML algorithm we usually have a training set of data and a separate dataset, called test set. The ML takes the following steps:

- The program is fed the training examples. This is when the ML takes place.
- The test set is fed into the program. The target variable for each example from the test set isn't given to the program and the program decides which class each example should belong to.
- The target variable or class that the training set belongs to is then compared to the predicted value and we can get a sense as how accurate the algorithm is.

> A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.



3.3 Key Tasks of ML

There are four techniques in ML as described below:

3.3.1 Supervised Learning

In supervised learning, the target is to infer a function or mapping from training data that is labeled. The training data consist of input vector X and output vector Y of labels or tags. A label or tag from vector Y is the explanation of its respective input example from input vector X. Together they form a training example. In other words, training data comprises training examples. If the labeling does not exist for input vector X, then X is unlabeled data. The output vector Y consists of labels for each training example present in the training data. A supervisor provides the labels for output vectors. Often, these supervisors are humans,

but machines can also be used for such labeling. Two groups come under supervised learning:

1 Classification

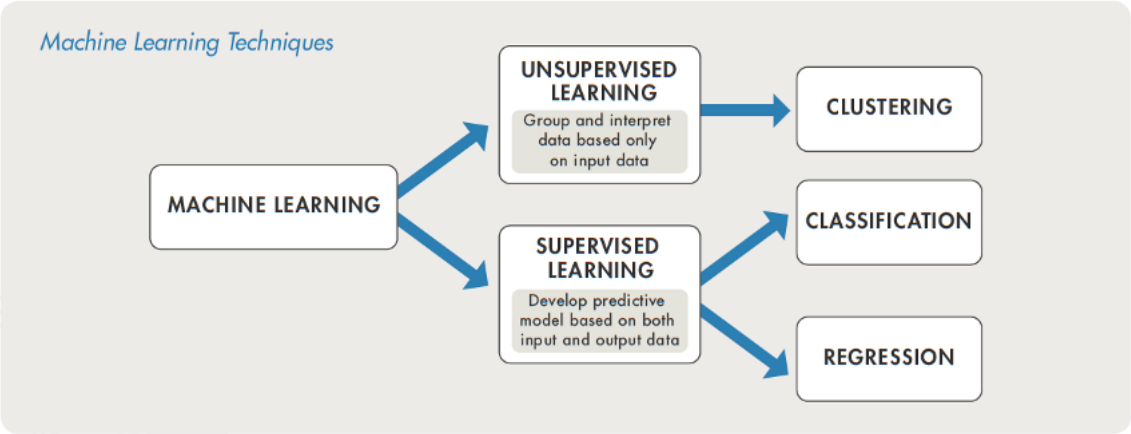
2 Regression

Question 3.1: Classification vs Regression

When should you use classification over regression?
Answer: Classification produces discrete values and dataset to strict categories, while regression gives you continuous results that allow you to better distinguish differences between individual points. You would use classification over regression if you wanted your results to reflect the belongingness of data points in your dataset to certain explicit categories.

3.3.2 Unsupervised Learning

In unsupervised learning, we lack supervisors or training data. In other words, all what we have is unlabeled data. The idea is to find a hidden structure in this data. Clustering comes under unsupervised learning.



Question 3.2: Supervised vs Unsupervised

Mention the functions of the supervised and unsupervised learning.

Answer: Supervised learning performs functions such as classifications, speech recognition, regression, time series prediction, and string annotation. On the other hand, unsupervised learning performs functions such as identifying data clusters. In addition, unsupervised learning also helps in finding low-dimensional data representations. Unsupervised learning helps in finding interesting directions in data.

3.3.3 Semi-supervised Learning

In this type of learning, the given data are a mixture of classified and unclassified data. This combination of labeled and unlabeled data is used to generate an appropriate model for the classification of data. In most of the situations, labeled data is scarce and

unlabeled data is in abundance. The target of semi-supervised classification is to learn a model that will predict classes of future test data better than that from the model generated by using the labeled data alone. The way we learn is similar to the process of semi-supervised learning.

For example, a child is supplied with unlabeled data provided by the environment. The surroundings of a child are full of unlabeled data in the beginning. Labeled data are provided by the supervisor- for example, a father teaches his children about the names (labels) of objects by pointing toward them and uttering their names.

3.3.4 Reinforced Learning

The reinforcement learning method aims at using observations gathered from the interaction with the environment to take actions that would maximize the reward or minimize the risk. In order to produce intelligent programs (also called agent) reinforcement learning goes through the following steps:

1. Input state is observed by the agent. 2. Decision making function is used to make the agent perform an action. 3. After the action is performed, the agent receives reward or reinforcement from the environment. 4. The state-action pair information about the reward is stored.

Using the stored information, policy for particular state in terms of

action can be fine-tuned, thus helping in optimal decision making for our agent.

Figure 17 shows the clustering of the data. The bottom panel shows data labeled by color.

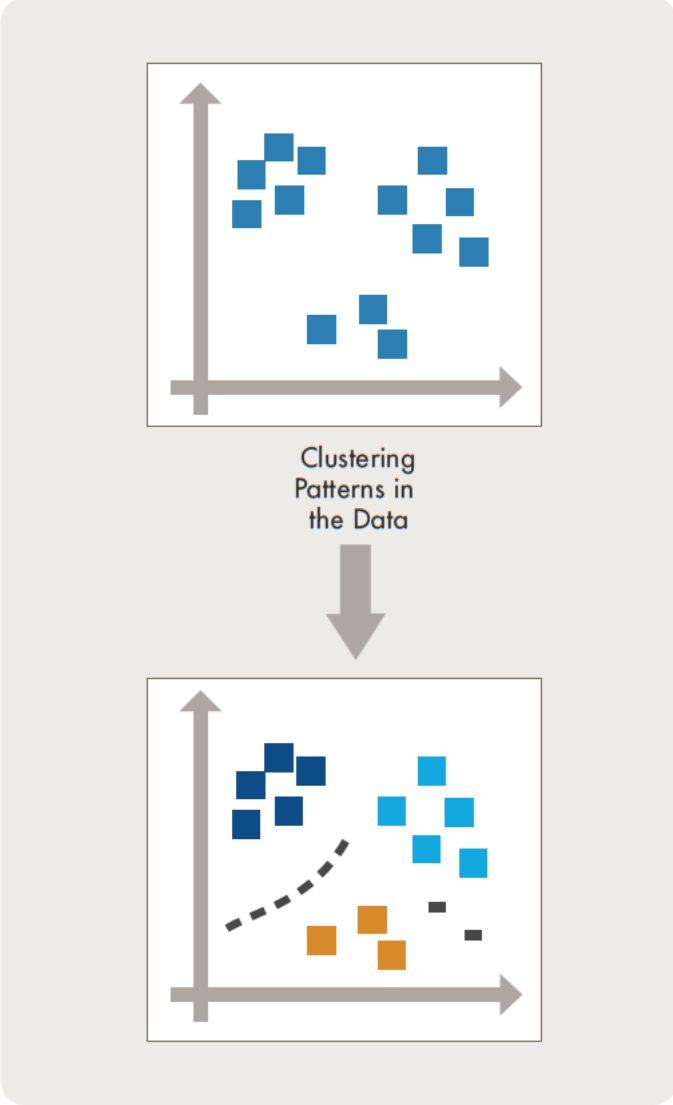


Figure 17: Clustering

3.4 Types of ML Algorithms

There are a few types of ML algorithms (Fig 18). Deciding the correct algorithm to use is a challenging problem. There is no best method. Finding the best algorithm is a matter of trial and error. Algorithm selection depends on the type and size of data one tries to use.

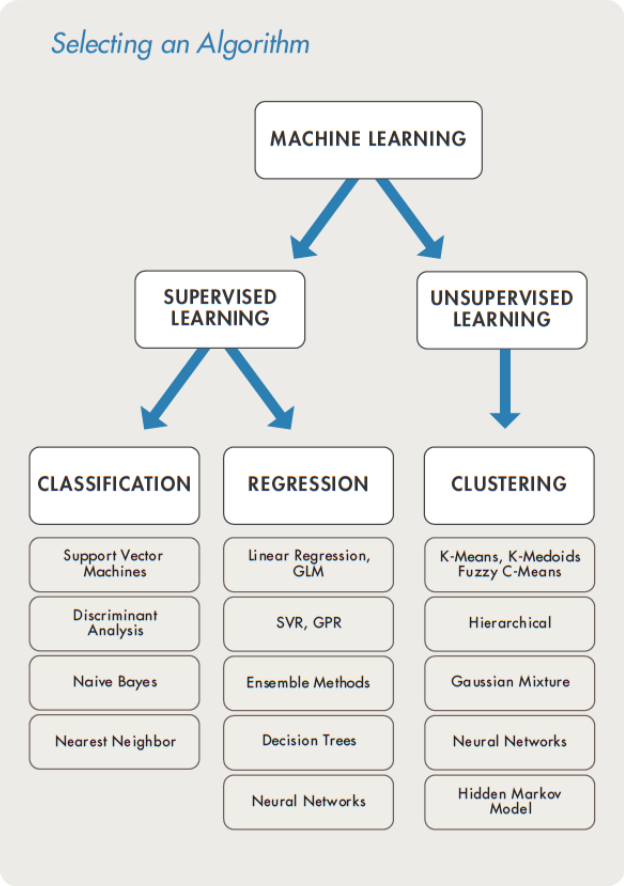


Figure 18: Types of ML Algorithms

Question 3.3: Neural Networks

What are the pros and cons of neural networks?

Answer: Neural networks can provide performance breakthroughs for unstructured datasets like video, images, and audio. The higher flexibility in neural networks helps in learning patterns better than other ML algorithms. The cons of neural network imply towards the requirement of a large amount of training data. Furthermore, neural networks also have setbacks in terms of selecting architecture and understanding of underlying internal layers.

3.5 K-Nearest Neighbors

The K-Nearest Neighbor (KNN) is a very effective classification method. It uses the concept of distance measurement to classify items. The KNN algorithm works as follows: We consider an existing set of data, the training set. We have labels for all of these data- in other words, we know which class each set of this data would belong to. When we are given a new piece of data without a label, we compare that new piece of data to the existing data, every piece of the existing data. We then take the most similar piece of data (the nearest neighbors) and look at their labels. We look at the top k most similar pieces of data from our known dataset (this is where the k comes from). Finally, we take a majority vote from the k most similar pieces of data and the

majority is the new class we assign to the data we were asked to classify.

The basic idea of KNN is that similar records congregate in n-dimensional space, with the same target class labels. This is the main logic behind the kNN algorithm. The entire training dataset is memorized and when unlabeled example records need to be classified, the input attributes of the new unlabeled records are compared against the entire training set to find the closest match. The class label of the closest training record is the predicted class label for the unseen test record. This is a nonparametric method where no generalization or attempt to find the distribution of the dataset is made. We then find the closest training record for each test dataset.

Example:
Figure 19 shows a training set with two dimensions and the target class values as circles and triangles. The unlabeled test record is the dark square in the center of the scattered plot. When $k = 1$, the predicted target class value of an unlabeled test record is triangle because the closest training record is a triangle. However, what if the closest training record is an outlier with an incorrect class in the training set? Then all the labeled test records near the outlier will get the wrong classification. To prevent this, we increase the value of k to 3. In this case, the nearest three training records are considered instead of one. From ths figure, based on a majority class of the nearest three training records, the predicted class of the test record is concluded as a circle. Since the class

of the target is often measured by voting, k is often assigned an odd number for a 2 class problem. The main step in kNN is measurement of the proximity of the data points.

3.6 Measure of Proximity

A measure of proximity between two records is a measure of the proximity of its attributes. The simplest way to identify the proximity is through measuring distances. The distance between two points $X(x_1, x_2)$ and $Y(y_1, y_2)$ in two-dimensional space is:

$$\text{Distance d} = \sqrt{(x_1 – y_1)^2 + (x_2 – y_2)^2}$$

We could generalize this to n attributes where X is $(x_1, x_2, \ldots, x_n)$ and Y is $(y_1, y_2, \ldots, y_n)$

$$\text{distance d} = \sqrt{(x_1 – y_1)^2) + (x_2 – y_2)^2) + \ldots + (x_n - y_n)^2}$$

Once the nearest k neighbors are determined, the process of determining the predicted target class is straightforward. The predicted target class is the majority class of the nearest k neighbors

$$y' = \text{majority class}(y_1, y_2, \ldots, y_k)$$

where y' is the predicted target class of the test data point and y_i is the class of the i^{th} neighbor n_i.

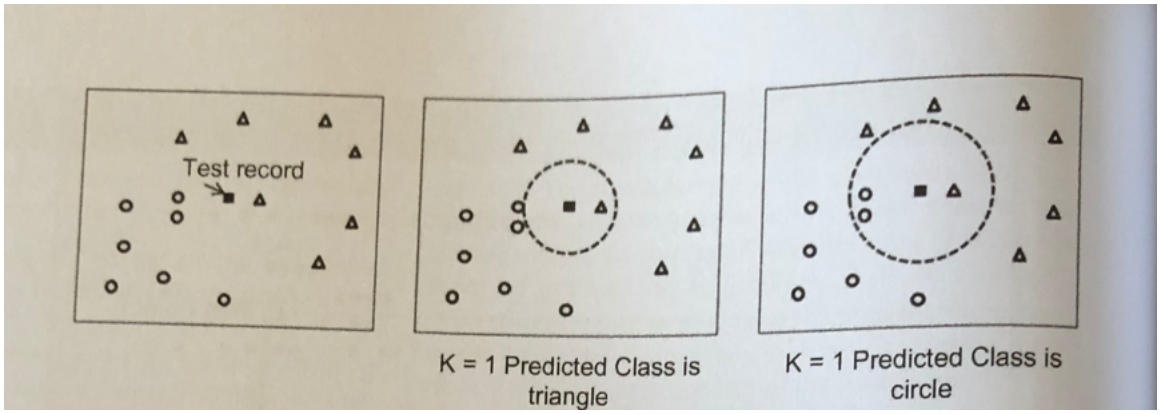


Figure 19: K-Nearest Neighbor

Question 3.4: K value in KNN

How can we find the best K value in the K-Nearest Neighbor algorithm?

Answer: Cross-validation is a smart way to find out the optimal K value. It estimates the validation error rate by holding out a subset of the training set from the model building process. Cross-validation (let's say 10 fold validation) involves randomly dividing the training set into 10 groups, or folds, of approximately equal size. 90% data is used to train the model and remaining 10% to validate it. The misclassification rate is then computed on the 10% validation data. This procedure repeats 10 times. Different group of observations are treated as a validation set each of the 10 times. It results to 10 estimates of the validation error which are then averaged out.

Weights

The point about the kNN algorithm is that the data points closer to each other are similar and hence, have the same target class labels. When k is more than one, it can be argued that the closest neighbors should have more say in the out come of the predicted target class than the farther neighbors. The farther neighbors should have less influence (in voting) in determining the final outcome. This can be accomplished by introducing a weighting scheme in the final voting step. Weights (w_i) should satisfy two conditions: they should be proportional to distance of the test point from the neighbor and the sum of the weights has to be one. An example for the weight is

$$w_i = \frac{e^{-d(x,n_i)}}{\sum_{i=1}^{k} e^{-d(x,n_i)}}$$

where w_i is the weight of i^{th} neighbor n_i, k is the total number of neighbors and x is the test data point. The weights are used in predicting target class y':

$$y' = \text{ majority class}(w_1 * y_1, w_2 * y_2, \ldots, w_n * y_n)$$

where y_i is the class outcome of neighbor i.

Categorical vs. Nominal Data

The distance measure works well for numeric attributes. However, if the attribute is categorical, the distance between two points is

either 0 or 1. If the attribute values are the same, the distance is 0 and if they are different, the distance is 1. For example, distance between (overcast, sunny) = 1 and distance between (sunny, sunny) =0. If the attribute has mor than two values, then the ordinal values can be converted to integer data types with values $0, 1, 2, \ldots, n-1$ and the converted attributes can be treated as a numeric attribute for distance calculation.

Pros and Cons of kNN

Pros: high accuracy, insensitive to outliers, no assumptions about data
Cons: computationally expensive, requires a lot of memory
Works with both numeric and nominal values

3.7 Decision Trees

Decision trees are widely used in data science. They are easy to set up and easy to interpret. There is a response variable in this technique and classification is done based on this. Normally, the response variables have two classes: Yes or No (1 or 0). Decision tree can also handle cases where response trees have more than two categories. Regression trees are like decision trees but used for numeric prediction problems when the response variable is numeric or continuous – like prediction of the price of consumer goods based on several input factors. In both cases the prediction values may be either categorical or numeric. The target variable indicates

what type of decision tree is needed.

A decision tree takes the form of a decision flowchart where an attribute is tested at each node. At the end of the decision tree, there is a leaf node where a prediction is made about the target variable based on the conditions set forward by the decision path. The nodes split dataset into subsets. The idea is to split the dataset based on the homogeneity of the data. For example, there are two variables, age and weight, that predict if a person is likely to sign up for a gym membership or not. In the training data, if it was seen that 90% of the people who are older than 40 signed up, the data can be split in two parts: one part consisting of people older than 40 and the other consisting of people younger than 40. The first part is now 90% pure from the standpoint of which class they belong to. In decision tree, a rigorous measure of impurity is needed. This is measured based on the following criteria:

1 The measure of impurity of a dataset must be at a maximum when all possible classes are equally represented. In the gym example above, if 50% of samples belonged to signed up and 50% to non-signed up, then this data would have maximum impurity.

2 he measure of impurity of a data set must be zero when only one class is represented. For example, if w group is formed of only those people who signed up for the membership (only one class=members), this subset has 100% purity or 0% impurity.

There are ways to measure these impurities, using entropy or Gini index.

Entropy

Suppose we have T events with equal probability of occurrence, P. The entropy is defined as $log_2(1/p)$ or $-logp$ where p is the probability of an event occurring. If the probability for all events is not identical, a weighted expression is needed. Then, the entropy, H, is defined as

$$H = -\sum_{k=1}^{m} p_k log_2(p_k)$$

where $k = 1, 2, 3, \ldots, m$ represent the m classes of the target variable and p_k is the proportions of the samples that belong to class k. For the gym class example there are two classes: member or non-member. If the dataset had 100 samples with 50% of each, then the entropy of the dataset is given by

$$H = -[(0.5log_20.5) + (0.5log_20.5) = -log_20.5 = -(-1) = 1$$

where $k = 2$- "member" or "non-member". If data can be partitioned into two sets of 50 samples each, that exclusively contain all members and all non-members, the entropy of either of these partitioned sets is given by

$$H = -log_2 1 = 0$$

any other proportion of the sample will give entropy values between 0 or 1.

The Gini index is similar to entropy in its characteristics and is given by

$$G = 1 - \sum_{k=1}^{m} p_k^2$$

The value of G ranges between 0 and 0.5. Otherwise, has similar properties to the entropy, H. In decision tree, there are two questions to be considered at each step: where to split the data and where to stop.

Table 4.1 The Classic Golf Dataset

Outlook	Temperature	Humidity	Windy	Play
Sunny	85	85	false	no
Sunny	80	90	true	no
Overcast	83	78	false	yes
Rain	70	96	false	yes
Rain	68	80	false	yes
Rain	65	70	true	no
Overcast	64	65	true	yes
Sunny	72	95	false	no
Sunny	69	70	false	yes
Rain	75	80	false	yes
Sunny	75	70	true	yes
Overcast	72	90	true	yes
Overcast	81	75	false	yes
Rain	71	80	true	no

Implementation of the decision tree

Here we demonstrate the decision tree with an example.

Step 1: Where to split the data

Suppose we want to decide whether or not play golf, given four attributes: Temperature, Humidity, outlook and Wind. The target variable is whether to play golf or not: yes or no.

We start partitioning the data on each of the four attributes. Let's start with Outlook. There are three categories for this attribute- sunny, overcast and rain. We see from the table that when it is overcast, there are four examples where the outcome was "Play=yes" for all four cases. Therefore, proportion of examples in this case is 100% or 1. If we split the sample here, the resulting four sample partition will be 100% pure for Play=yes. The entropy can be calculated as follows:

$$H_{outlook,overcast} = -(0/4)log_2(0/4)–(4/4)log_2(4/4) = 0.0$$

similarly for other attributes, we have

$$H_{outlook,sunny} = -(2/5)log_2(2/5)–(3/5)log_2(3/5) = 0971$$

and

$$H_{outlook,rain} = -(3/5)log_2(3/5)–(2/5)log_2(2/5) = 0.971$$

For the attribute on the whole, the total information I is calculated as the weighted sum of these component entropies. There are four instances of outlook=overcast, thus the proportion of overcast is given by 4/14. The other proportions are (for outlook=sunny and rain) 5/14 each:

$$I_{outlook} = P_{outlook,overcast} \times H_{outlook,overcast} + P_{outlook,sunny} \times H_{outlook,sunny} + P_{outlook,rain} \times H_{outlook,rain}$$

$$I_{outlook} = (4/14) \times 0 + (5/14) \times 0.971 + (5/14) \times 0.971 = 0.693$$

If the data had not been partitioned along the three values of outlook, the total information would have been the weighted average of the respective entropies for the two cases whose overall proportions were 5/14 (Play=no) and 9/14 (Play=yes).

$$I_{outlook,nopartition} = -(5/14) log_2(5/14) – (9/14) log_2(9/14) = 0.940$$

By creating these partitions, some entropy has been reduced and some information gained. This is called information gain. In case of the outlook, this is given by

$$I_{outlook,nopartition} – I_{outlook} = 0.940 – 0.693 = 0.247$$

Similar information gain values for other three attributes can now be computed.

For numeric variables, possible split points to examine are the averages of the available values. For example, the first potential split point for humidity could be average [65,70] which is 67.5, the next potential split point could be average[70,75] that is 72.5 and so on. Similar logic can be used for other numeric attributes, Temperature. The algorithm computes the information gain at each of these potential split points and chooses the one which maximizes it. Results are listed in the Table and show that if the split is done along the three outlook variables, the maximum information gain can be achieved. This gives the first node of the decision tree. The terminal node for the outlook =overcast branch consists of four samples all of which belong to a class, Play=yes. The other two consist of a mixture of classes- the outlook=rain branch has three yes results and outlook=sunny branch has three no results.

To address the question here again: Where to split the data? This is decided based on the information gain for all attributes (as in the table). The other attributes that achieve the highest gains are also used. Therefore, outlook=sunny branch can be split along humidity (which yields the second highest information gain) and the outlook=rain branch can be split along Wind (which yields

the third highest gain).

Table 4.2 Computing the Information Gain for All Attributes

Attribute	Information Gain
Temperature	0.029
Humidity	0.102
Wind	0.048
Outlook	0.247

Step 2: When to Stop Splitting Data? There are several situations where the process can be terminated:

1 No attribute satisfies the minimum information gain threshold

2 A maximum depth is reached are the tree grows larger, the interpretation gets harder and overfitting occurs.

3 There are less than a certain number of examples in the current sub-tree, a mechanism to stop overfitting.

Summary:

The application of the decision tree algorithm can be summarized as:

1 Using entropy, sort the dataset into homogeneous and non-homogeneous variables (by class). Homogeneous variables have low information entropy and non-homogeneous vari-

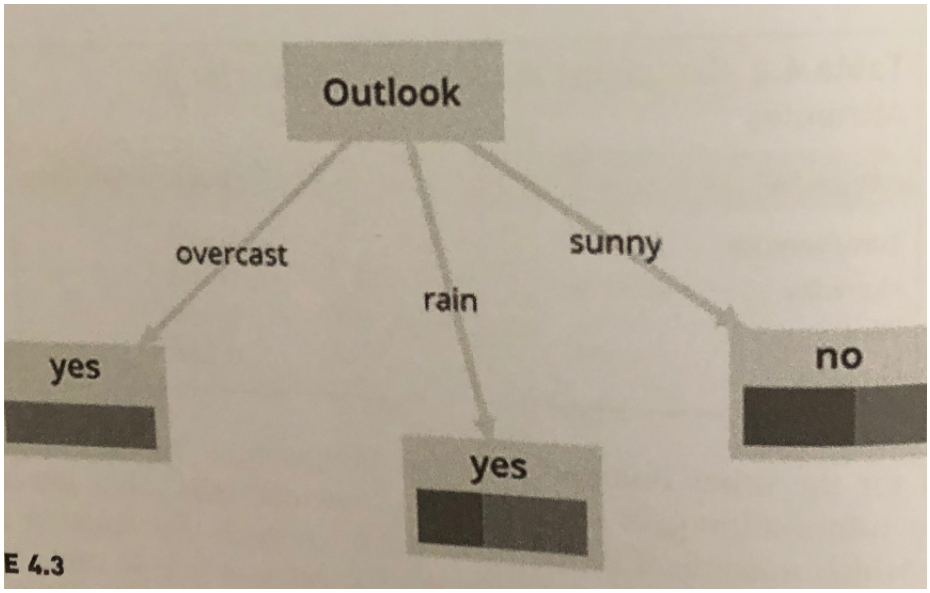
ables have high information entropy. This was done in the calculation of $I_{outlook,nopartition}$.

2 Weight the influence of each independent variable on the target variable using the entropy weighted averages. This was done during the calculation of $I_{outlook}$

3 Compute the information gain which is the reduction in the entropy of the target variable due to its relationship with each independent variable. This is the difference between the information entropy found in Step 1 and the joint entropy from Step 2. This was done during the calculation $I_{outlook,nopartition}{-}I_{outlook}$

4 The independent variables with the highest information gain will become the root or the first node on which the dataset is divided. This was done using the calculation of the information gain.

5 Repeat this process for each variable for which the entropy is nonzero. If the entropy of a variable is zero, then the variable becomes a "leaf" node.



3.8 Naïve Bayes

The objective of all data science algorithms is prediction of a target variable. The naïve Bayes algorithm is one of those techniques that relies more on statistics and probability theorem. In general, classification techniques find class labels by approximating the relationship between the attributes and the class label. The naïve Bayes technique assigns likelihood probability for the association. The classifications are not always that easy. Often there are many attributes and factors involved. One then needs to find the likelihood of the final prediction given each of the attributes.

Consider the case for a default in home mortgages and assume the average default rate is 2%. The likelihood of an average person defaulting o their mortgage is then 2%. However, if a given person credit history is high, that reduces the likelihood of default

for that person to below 2% (the average). Also, if the person's annual income is above average with respect to loan value, then the likelihood of default falls further. As more independent factors are considered that affect the outcome, the accuracy of our prediction (whether the person will default on the mortgage loan or not) will increase. The naïve Bayes algorithm estimates the probabilistic relationship between the attributes and the class label. The algorithm makes the naïve assumption of independence between the attributes and hence, the name. The independent assumption between the attributes may not always be true. For example, the house prices and incomes are independent but those with higher incomes often go for more expensive houses and this introduces a correlation among the attributes.

Principles of Naïve Bayes Algorithm

The naïve Bayes algorithm is built on the Bayes theorem. We formulated this in the statistics chapter. In simple terms, Bayes theorem provides an expression of how a degree of subjective belief changes to account for new evidence. Here we briefly review Bayes' theorem.

Assume X is the evidence (attribute set) and Y is the outcome (class label). Here X is a set and not an individual attribute and hence, $X = X_1, X_2, \ldots, X_n$, where X_i is an individual attribute. The probability of outcome $P(Y)$ is called prior probability. This can be calculated from the training dataset. The prior probability shows the likelihood of an outcome in a given dataset. For ex-

ample, in the mortgage case, $P(Y)$ is the default rate on a home mortgage, which is 2% here. $P(Y|X)$ is called the conditional probability, which provides the probability of an outcome given the evidence, that is when the value of X is known. In the mortgage example $P(Y|X)$ is the average rate of defaults given that one's credit history is known. For someone with good credit history, the probability of default is likely to be less than 2%. $P(Y|X)$ is called posterior probability. Calculating the posterior probability is the aim of the Bayes techniques. This presents the likelihood of an outcome as the conditions are learned.

Bayes' theorem states that

$$P(Y|X) = \frac{P(Y) \times P(X|Y)}{P(X)}$$

where $P(X|Y)$ is the conditional probability. It is the probability of the existence of conditions given an outcome and is calculated from the training dataset. $P(X)$ is the probability of the evidence. In the mortgage example, this is the proportion of the individuals with a given credit score. To classify a new record, one can compute $P(Y|X)$ for each class of Y and see which probability wins. Class label Y with the highest value of $P(Y|X)$ wins for the given condition X. Since $P(X)$ is the same for every class value of the outcome, it can be considered as a constant. More generally, for an example set with n attributes $X = X_1, X_2, \ldots, X_n$,

$$P(Y|X) = \frac{P(Y) \times \prod_{i=1}^{n} p(X_i|Y)}{P(X)}$$

Example:

Consider one wants to calculate the probability of the weather conditions allowing play golf. There are the following attributes to be considered: Temperature (X_1), Humidity (X_2), Outlook (X_3) and Wind (X_4). Given these, one wants to use Bayesian method to decide whether or not play golf. Here weather conditions are the evidence and decision to play or not is the belief. The Table provides the training data, with 14 examples with 9 examples conditions to play golf ("yes") and 5 examples not play golf ("no"). Based on the data in Table, one wants to decide whether to play golf or not under certain conditions. We take the following steps:

Step 1: Calculate Prior Probability $P(Y)$

Prior probability is the probability of an outcome. In this case, there are two possible outcomes: Play=yes and Play=no. From the table, out of 14 records with the "no" class and 9 records with the "yes" class, the probability of the outcome is:

$$P(Y = no) = 5/14 \quad P(Y = yes) = 9/14$$

It is important that the dataset used for calculation of tehse prob-

abilities is representative of the whole population. The stratified sampling will be ideal for the Bayes technique. That ensures the class distribution in the sample is the same as the population.

Step 2: Calculate Class Conditional Probability $P(X_i|Y)$

Class conditional probability is the probability of each attribute value for an attribute, for each outcome value. This is repeated for all the attributes: Temperature, Humidity, outlook and wind and for every outcome value. For example, for each value of the Temperature attribute, $P(X_1|Y = no)$ and $P(X_1|Y = yes)$ can be calculated by forming a class conditional probability table as shown here.

From the training dataset, there are 5 $Y = no$ and 9 $Y = yes$ records. Out of the $Y = no$ records, the probability of this happening can be calculated for when the temperature is High, Medium or Low. The values will be 2/5, 1/5 and 2/5 respectively. For the $Y = yes$, the probability is2/9, 3/9 and 4/9 respectively.

Similarly, the class conditional probability can be calculated for other attributes, as shown in Table 3.

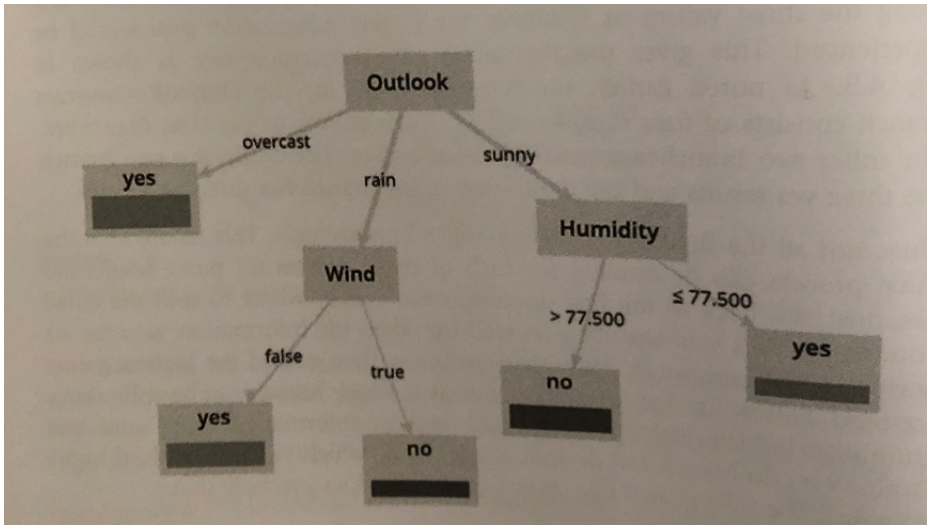


Table 4.4 Golf Dataset With Modified Temperature and Humidity Attributes

No.	Temperature X_1	Humidity X_2	Outlook X_3	Wind X_4	Play (Class Label) Y
1	High	Med	Sunny	false	no
2	High	High	Sunny	true	no
3	Low	Low	Rain	true	no
4	Med	High	Sunny	false	no
5	Low	Med	Rain	true	no
6	High	Med	Overcast	false	yes
7	Low	High	Rain	false	yes
8	Low	Med	Rain	false	yes
9	Low	Low	Overcast	true	yes
10	Low	Low	Sunny	false	yes
11	Med	Med	Rain	false	yes
12	Med	Low	Sunny	true	yes
13	Med	High	Overcast	true	yes
14	High	Low	Overcast	false	yes

Step 3: Predict the Outcome using Bayes' Theorem

Having all the class conditional probabilities, we can now predict the outcome. If a new unlabeled test record has the conditions Temperature = high, Humidity= low, outlook=sunny, and Wind=false, what would the class label prediction be?

This can be calculated based on the Bayes' theorem by estimating the posterior probability $P(Y|X)$ for both values of Y. Once $P(Y = yes|X)$ and $P(Y = no|X)$ are calculated, one can determine which outcome has higher probability with the predicted outcome being the one with the higher probability. In the above equation, $P(X)$ is the same for both outcome classes,

$$P(Y = yes|X) = \frac{P(Y) \times \prod_{i=1}^{n} p(X_i|Y)}{P(X)}$$

$$= \frac{p(Y=yes)[P(Temp=high|Y=yes)P(Humidity=low|Y=yes)P(Outlook=sunny|Y=yes)P(Wind=false|Y=yes)]}{P(X)}$$

Now we replace the probabilities to this equation

$$P(Y = yes|X) = \frac{9/14 * [2/9 * 4/9 * 2/9 * 6/9]}{P(X)} = \frac{0.0094}{P(X)}$$

Similarly we have

$$P(Y = no|X) = 5/14 * [2/5 * 4/5 * 3/5 * 2/5] = \frac{0.00274}{P(X)}$$

The estimates can be normalized by dividing both conditional probabilities by $(0.0094 + 0.027)$. We get

$$\text{Likelihood of (Play=yes)} = \frac{0.0094}{(0.0094 + 0.0274)} = 26\%$$

and

$$\text{Likelihood of (Play=no)} = \frac{0.0274}{(0.0094 + 0.0274)} = 74\%$$

This means $P(Y = yes|X) < P(Y = no|X)$. Therefore, the prediction for the unlabeled test record will be "Play=no".

Advantages of the Bayesian Method

The method is robust with simple computation. This involves creating a look up table of probabilities. It is also robust in handling missing values. For example, if the test example set does not have one of the attributes, it simply omits the corresponding class conditional probability for all the outcomes. This is a problem in other methods.

Pros and Cons

Pros: works with a small set of data, handles multiple classes
Cons: Sensitive to how the input data is prepared
Works with nominal values

Table 4.5 Class Conditional Probability of Temperature

Temperature (X_1)	$P(X_1\|Y = \text{no})$	$P(X_1\|Y = \text{yes})$
High	2/5	2/9
Med	1/5	3/9
Low	2/5	4/9

Table 4.7 Test Record

No.	Temperature X_1	Humidity X_2	Outlook X_3	Wind X_4	Play (Class Label) Y
Unlabeled test	high	Low	Sunny	False	?

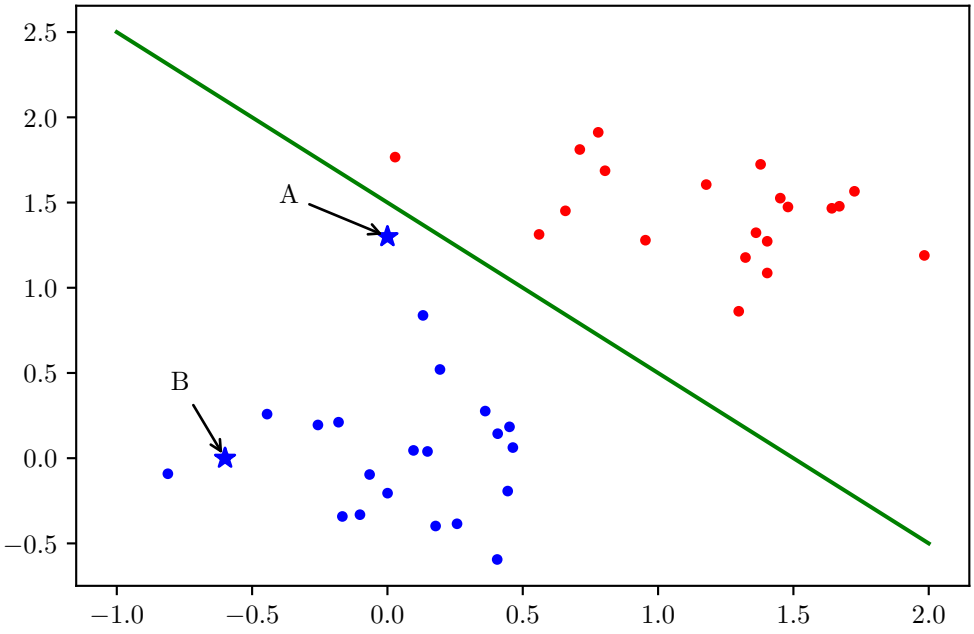
Table 4.6 Conditional Probability of Humidity, Outlook, and Wind

Humidity (X_2)	$P(X_2 \mid Y = \text{no})$	$P(X_2 \mid Y = \text{yes})$
High	2/5	2/9
Low	1/5	4/9
Med	2/5	3/9
Outlook (X_3)	**$P(X_3 \mid Y = \text{no})$**	**$P(X_3 \mid Y = \text{yes})$**
Overcast	0/5	4/9
Rain	2/5	3/9
Sunny	3/5	2/9
Wind (X_4)	**$P(X_4 \mid Y = \text{no})$**	**$P(X_4 \mid Y = \text{yes})$**
False	2/5	6/9
True	3/5	3/9

3.9 Support Vector Machines

In machine learning, support vector machine is a learning algorithm which can be used for both classification and regression problems (Here we are interested in the classification problem). The main idea behind the support vector machine is to find the hyperplane that is capable of separating the classes with maximum possible margins from the data points in each class.

Let's take a look at the following classification problem (linearly separable classification problem):



For the point A in the figure, you can see that the point is very close to the decision boundary, which means that by changing the boundary we are going to change our classification of point A as blue to red. But at the same time if you look at the point B we are pretty certain that our classification should be blue since it is not in the vicinity of the decision boundary or in a more general case of multi-dimension our **separating hyperplane**. So the task is to find a decision boundary that makes our confidence in our classification better.

Now let's look at the following problem. We want to have a linear classifier for a binary classification problem.

$$z = \sum_{i=1}^{n} w_i x_i + b = w^T x + b$$

$$y = h_{w,b}(x) = g(z) = g(w^T x + b)$$

In which g is defined as and $h_{w,b}$ is our hypothesis function:

$$g(z) = \begin{cases} 1, & \text{if } z \geq 0 \\ -1, & \text{if } z < 0 \end{cases}$$

Now we should formalize our qualitative analysis of the closeness to the decision boundary by introducing two concepts:

Functional Margin:

Imagine that we have a set of training examples $\{(X_j, y_j)\}$, we can define the functional margin of our hypothesis to be:

$$\zeta_j = z y_j = (w^T x_j + b) y_j$$

So for having a better classification for $y_j = 1$ we need the $w^T x_j + b$ to be large value, and for the case of $y_j = -1$ we will seek $w^T x_j + b$ to be a large negative value. So the larger the ζ is, we expect a larger confidence. Since our g function is only sensitive to sign of z, multiplying it by any positive value will not change our decision but will change our Functional margin. So we should make sure

that we use a normalized version of (w, b). Also, we define the functional margin of $h_{w,b}$ on the training set $\{(x_j, y_j)\}$ as:

$$\zeta = \min_{j=1,\dots,n} \zeta_j$$

So basically the worst functional margin among the training set.

Geometric margin:

We can use geometry to find a reasonable margin and that is to define the margin to be the distance from hyperplane of S : $w^T x + b = 0$. From this definition the $\vec{w}$ is orthogonal to the hyperplane and the distance is measure along this vector:

$$x_o = x_j - \zeta_j \frac{\vec{w}}{\|\vec{w}\|}$$

In which $x_o \in S$ which means:

$$\begin{cases} w^T x_0 + b = 0 \\ w^T (x_j - \zeta_j \frac{\vec{w}}{\|\vec{w}\|}) + b = 0 \\ \frac{w^T x_j + b}{\|\vec{w}\|} = \zeta_j \end{cases}$$

And for a more general case of both positive and negative points

we can define the geometrical margin to be:

$$\zeta_j = (\frac{w^T x_j + b}{\|\vec{w}\|})y_j$$

This is identical to our functional definition if we take $\|\vec{w}\| = 1$. The definition of the margin for the training set is like the functional margin:

$$\zeta = \min_{j=1,\ldots,n} \zeta_j$$

We should try to maximize this ζ for having a hypothesis parameters that have higher confidence.

Here we have to find the parameters which make ζ maximum:

$$\begin{cases} \max_{w,b} \zeta \\ (\frac{w^T x_j + b}{\|\vec{w}\|})y_j \geq \zeta \quad \text{For } j \in \{1, \ldots, N\} \\ \|\vec{w}\| = 1 \end{cases}$$

We can turned this into another optimization problem:

$$\begin{cases} \max_{w,b} \frac{\zeta}{\|\vec{w}\|} \\ (\frac{w^T x_j + b}{\|\vec{w}\|})y_j \geq \zeta \quad \text{For } j \in \{1, \ldots, N\} \end{cases}$$

As we discussed the $(\vec{w}, b)$ are scalable without changing the results. So, we can conclude that ζ is scaled as a result. Therefore, we can force $\zeta = 1$.

$$\begin{cases} \max_{w,b} \frac{1}{\|\vec{w}\|} \\ (\frac{w^T x_j + b}{\|\vec{w}\|}) y_j \geq 1 \quad \text{For } j \in \{1, \ldots, N\} \end{cases}$$

Notice that none of the above objective function are not convex (In the case of the first one the constraint of $\|\vec{w}\| = 1$ is not convex); which means that our previous method for finding the optimized solution would not work.

However, we can turn the maximization of $\frac{1}{\|\vec{w}\|}$ to minimization of $\|\vec{w}\|^2$, which is something that we have already know algorithm to optimize.

$$\begin{cases} \min_{w,b} \|\vec{w}\|^2 \\ (\frac{w^T x_j + b}{\|\vec{w}\|}) y_j \geq 1 \quad \text{For } j \in \{1, \ldots, N\} \end{cases}$$

Now we have our optimal margin classifier after solving the above quadratic optimization problem.

As we mentioned earlier this is a linear classifier. However, since the optimization problem only depends on the values of $< x_i . x_j >$ we can turn this problem to a non-linear classifier using a general method called **kernel trick**.

Kernel trick:

First we can define a feature map $\psi : \Re^n \to \Re^q$, in which we have a set of n attributes and we are mapping that to q features.

If we are using a feature map of order p and if we consider all the monomials of X_j we are going to have a ψ that look like this:

$$\vec{\psi}(X_j) = \begin{pmatrix} 1 \\ X_{j,1} \\ X_{j,2} \\ \vdots \\ X_{j,1}^2 \\ X_{j,1}X_{j,2} \\ X_{j,1}X_{j,3} \\ \vdots \\ X_{j,1}^3 \\ X_{j,1}^2X_{j,2} \\ X_{j,1}^2X_{j,3} \\ \vdots \end{pmatrix}$$

If we count them we have 1, 0^{th} order term, we have n, 1^{th} order terms, $\binom{n+2-1}{2}$, 2^{th} order terms and so on: (Using the famous *star and bar argument*)

$$q = \binom{n-1}{0} + \binom{n+1-1}{1} + \binom{n+2-1}{2} + \cdots + \binom{n+p-1}{p}$$
$$= \sum_{i=0}^{p} \binom{n+i-1}{i} = \binom{n+p}{p}$$

So for $n = 10$ attributes and feature map of order $p \leq 3$ with all the monomials, we have a weight vector of size 120 and if we have $n = 100$ attributes it will become a vector of size 161700. So any high order polynomial using along with large number of attributes make this quite unreasonable to use!

This inner product is the **Kernel** of the feature map ψ, so the kernel is a function $\Re^n \times \Re^n \to \mathbb{R}$:

$$K(X_i, X_j) = \langle \psi(X_i), \psi(X_j) \rangle$$

K is the $n \times n$ matrix form by elements of $K_{i,j} = K(X_i, X_j)$ is the kernel matrix.

For the ψ mentioned we have:

$$\langle \psi(X_i), \psi(X_j) \rangle = 1 + \langle X_i, X_j \rangle + \langle X_i, X_j \rangle^2 + \cdots + \langle X_i, X_j \rangle^p$$

Which means that we only need to calculate $\langle X_i, X_j \rangle$ explicitly

and we can make a kernel function out of that instead of calculating the direct inner product $\langle \psi(X_i), \psi(X_j) \rangle$.

We can view this kernel function as how similar to vectors are, so basically when two instances are very similar we expect to get large values for the kernel and when they are not so similar we expect smaller values. If you think about the kernel as this, it is quite natural to define something close to a Gaussian function such as this:

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

So, if two examples are close we get a value close to 1 and if they are far off we get a value close to 0. This kernel is called a **Gaussian kernel**.

This method can be used to turn the linear boundary support vector machines into a non-linear boundary, simply using the kernel trick.

3.10 Principal Component Analysis

This is one of the main dimensional reduction and feature extraction techniques that tries to find the intrinsic subspace (linear subspace) that contains the original data. This is equivalent to finding the maximum variation in the data and finding the transformations that allows us to explore the data in the new subspace

with basis that are called the principal components.

Suppose that we have the following dataset:

$$\boldsymbol{x} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}]_{d \times n}$$

With n data points and d features. We are trying to find the $\boldsymbol{u_1} \in \Re^d$ (first principal component) that maximizes the following:

$$\text{Variance}(\boldsymbol{u_1^T x}) = \boldsymbol{u_1^T} S \boldsymbol{u_1}$$

Where S is the sample (data) covariance. As we discussed before we want to find the vector $\boldsymbol{u_1}$ that maximizes the variance of the data. This is an optimization problem which is quadratic and resemble a form of $f(x) = x^2$ function. If this is the case then we can argue that the problem is ill-defined since the function f does not have any maximum (it is unbounded). We could find this to be the case by the following argument: Imagine $\boldsymbol{u^*}$ is the principal component that maximizes the variance of the data on the projected space and we call the variance:

$$S^* = \text{Variance}(\boldsymbol{u^{*T} x}) = \boldsymbol{u^{*T}} S \boldsymbol{u^*}$$

$\gamma \boldsymbol{u^*}$ is also a solution for $\forall \gamma \in \Re$ and if we take $|\gamma| > 1$ then:

$$\text{Variance} = \gamma^2 S^* > S^*$$

. This is a contradiction with our assumption that $\boldsymbol{u^*}$ was the solution that maximize the variance.

In order to circumvent this problem, we introduce an additional constraint which does not change the objective of our problem which was finding a new basis to express our data which preserve the intrinsic variance. We can simply require that $\boldsymbol{u_1^T u_1} = 1$, so we can have a maximum solution under this constraint.

$$\mathcal{L}(\boldsymbol{u_1}, \lambda_1) = \boldsymbol{u_1^T} S \boldsymbol{u_1} - \lambda_1(\boldsymbol{u_1^T u_1} - 1)$$

So we can solve the optimization problem using the Lagrange multipliers.

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{u_1}} = 0$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{u_1}} = 2S\boldsymbol{u_1} - 2\lambda_1 \boldsymbol{u_1} = 0$$

$$S\boldsymbol{u_1} = \lambda_1 \boldsymbol{u_1}$$

Which means that λ_1 is the eigenvalue and $\boldsymbol{u_1}$ is the eigenvectors of the covariance matrix (S). Any eigenvector can satisfy the Lagrangian. However, using the eigenvector equation we can write:

$$\text{Variance}(\boldsymbol{u_1^T x}) = \boldsymbol{u_1^T} S \boldsymbol{u_1} = \boldsymbol{u_1^T} \lambda_1 \boldsymbol{u_1} = \lambda_1 \underbrace{\boldsymbol{u_1^T u_1}}_{=1} = \lambda_1$$

So the largest eigenvalue of covariance matrix correspond to the eigenvector that reflect the direction of the maximum variability of the data. This can be applied to other principal components. (second largest correspond to second principal component and so on)

Few applications of the PCA:

1. Mapping the original feature space to lower dimension $p < d$:

$$y = u_p^T X$$

2. Reconstruct the data based on the first p principal components. This is one way we can reduce the noise in the data set assuming that the noise is much smaller than the highest signal (pattern) in our data

$$\hat{X} = u_p u_p^T X$$

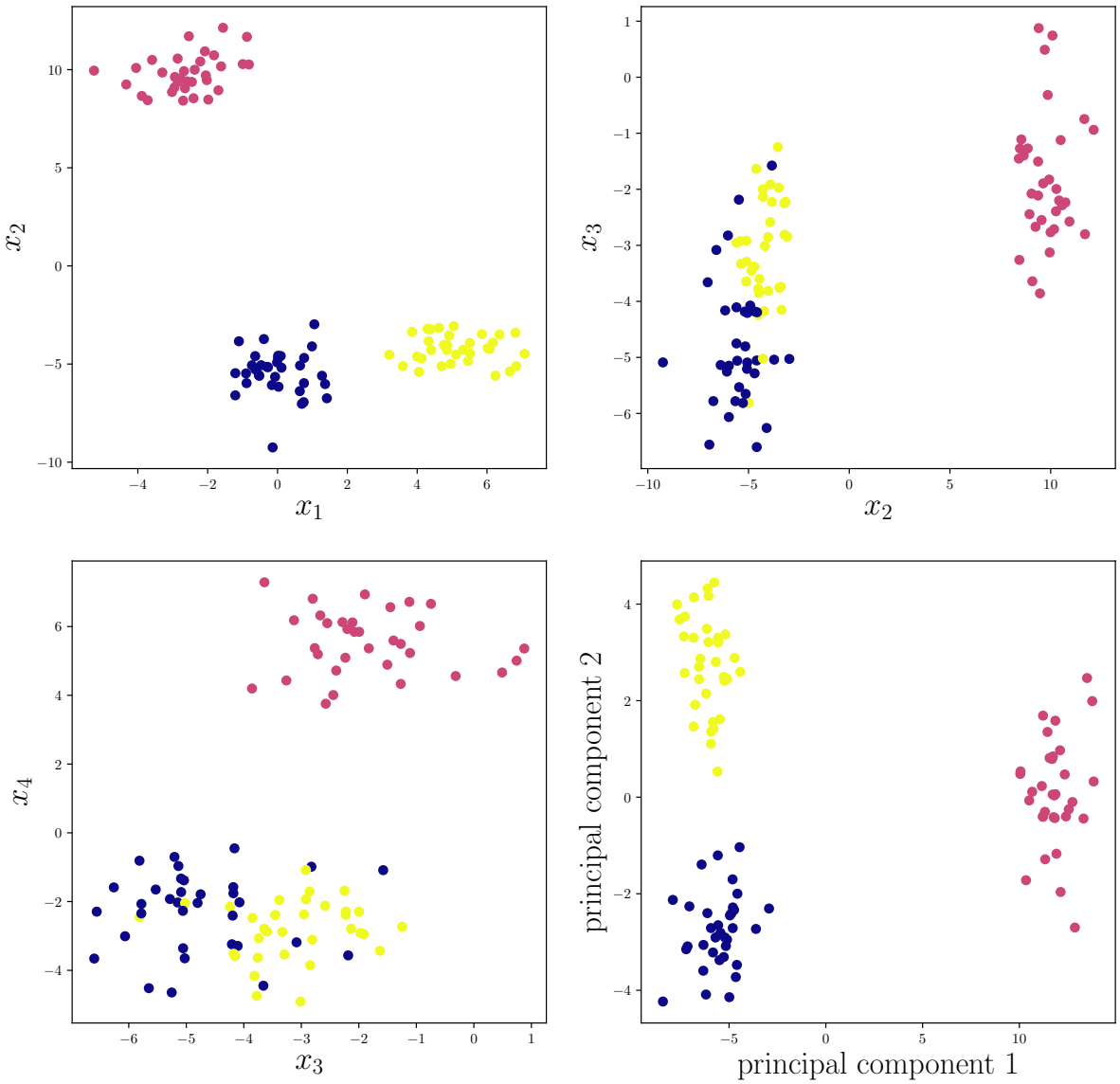
3. We can reconstruct the out of sample data based on the sample.

4. We can use the kernel trick described in the previous section and build kernel PCA which can grasp some non-linearity within our data.

Now let's look at the following examples:

1. Four dimensional data set (figure shows the cross-section of different dimensions). The bottom right figure which is the transformation of the data points into it's two first principal components

Four dimensional data set projected into the first two principal component



2. MNIST digits data set, with 500 examples and 64 features (8×8 pixels) transformed into its first two principal components. As it is shown in the figure many pattern in the data set has been conserved as we see a degree of separation

between classes. (Remember that PCA is an unsupervised learning algorithm and we are doing dimension reduction from $\Re^{64} \rightarrow \Re^2$)

MNIST digits transformed into the First two Principal components

3.11 Cluster analysis

In the cluster analysis, which is an unsupervised learning technique, we aim at putting objects (examples) into groups based on a similarity measure. Basically we want to put similar objects into

the same group (cluster). This allows one to find all the meaningful groups in the data. Depending how this is done, teh number of groups and clusters are either user defined or automatically determined by the algorithm from the data set.

Clustering Techniques

Prototype-based clustering: in this algorithm each cluster is represented by a central data object, also called a prototype. The prototype of a cluster is often the center of the cluster. This is therefore, also called centroid clustering. As an example, each customer cluster could have a central prototype customer and customers with similar properties that are associated with the prototype customer of a cluster.

Density Clustering: This is defined based on the excess number density of points per unit space and are separated by sparse space. It is defined as a dense region where data objects are concentrated surrounded by a low density area with sparse data points. In this form of clustering the noise objects are unassigned to any cluster.

Hierarchical clustering: This is a process where a cluster hierarchy is created based on the distance between data points. The output of a hierarchical clustering is a *dendrogram*- a tree diagram that shows different clusters at any point of precision specified by

the user.

Model-based clustering: this comes from statistics and probability distribution models. A cluster can be thought of as a grouping that has the data points belonging to the same probability distribution. Therefore, each cluster can be represented by a distribution model (eg. Gaussian or Poisson) where the parameters of the distribution can be optimized between the cluster data and the model.

k-means clustering is a proto-type clustering technique whereas Density-Based Spatial Clustering of Application with Noise (DBSCAN) belongs to the density clustering category. When measuring clustering, one needs to find the distance between the points and the similarity of the points (variance) belonging to the same cluster. These are the useful quantities needed to define clustering:

1. **Variability of a cluster**: This shows how much difference exist between each element of the cluster and the mean of the cluster.

$$V(c) = \sum_{x \in c} d(mean(c), x)$$

In which c is a cluster, and $d(x_0, x_1)$ is the distance between the mean and each point in the cluster. For any meaningful cluster, $V(c)$ must be a minimum.

2. **Dissimilarity of the set of clusters**: This is a measure for a aggregate variability of a set of clusters. Having calculated $V(c)$ for each cluster, we can define this as:

$$D(C) = \sum_{c \in C} V(c)$$

This indicates how similar different clusters are and if we could continue to make bigger clusters by combining them.

3. **Metric**: In order to define any notion of similarity, it is quite natural to define a measure of distance between data points. Mathematically speaking any function with the following properties can be considered as the distance function (metric). A metric on set S is defined as:

$$d : S \times S \to [0, \infty)$$

For $\forall x, y, z \in S$

(a) $d(x, y) \geq 0$

(b) $d(x, y) = 0 \iff x = y$

(c) $d(x, y) = d(y, x)$

(d) $d(x, y) \leq d(x, z) + d(z, y)$

Usually we define the distance between two points to be *Euclidean*. If we take two vectors as $X, Y \in \Re^M$, we define

the Euclidean distance to be:

$$d(\vec{X}, \vec{Y}) = \sqrt{\sum_{i=1}^{M}(X_i - Y_i)^2}$$

This is a special case for a more general class of distance definitions which are called the *Minkowski* distance, which is defined as:

$$d(\vec{X}, \vec{Y}) = \sqrt[n]{\sum_{i=1}^{M}(X_i - Y_i)^n}$$

Another special cases from Minkowski are the *Taxicab* distance (also known as Manhatan distance):

$$d(\vec{X}, \vec{Y}) = \sum_{i=1}^{M} \|X_i - Y_i\|$$

And the Tchebychev distance:

$$d(\vec{X}, \vec{Y}) = \lim_{n\to\infty} \sqrt[n]{\sum_{i=1}^{M}(X_i - Y_i)^n}$$

Now let's go back to our original clustering problem. Now if you look at the definition of Variablity, you see that for the special case of Euclidean distance we are just missing a factor of $1/\sqrt{k}$

in which k is the number of points in a given cluster, from the definition of variance.

What does it mean to not include the number of points in a cluster when calculating variability?

By doing that we are assigning higher variability to the larger cluster than the smaller cluster of the same variance.

You may have guessed by now, that we are trying to come up with an objective function for our optimization problem. So we can ask for a set of clusters that would minimize the dissimilarity of the clusters.

But if we put a cluster on each point, we are going to get zero dissimilarity which obviously is not a useful answer. For avoiding this we need to define a constraint; for example, we can constrain the total distance between clusters to be smaller than some value, or enforce a maximum number for clusters.

Hierarchical Clustering

In this method of clustering we find all the possible clustering according to our similarity measure without any restriction on the number of clusters. (Or equivalently no restriction on the threshold distance for assigning two objects in the same cluster) There are two approaches to create a hierarchy of clusters. A bottom-up approach is where each data point is considered a cluster, with the clusters merged to form one massive cluster. The top-down

approach is where the data set is considered one cluster and they are divided into different sub-clusters until individual data are defined as separate clusters.

The naive algorithm is as follows:

1. Assign a cluster to each point, so N clusters for N points.
2. Find the most similar clusters and merge them together. Now we have $N - 1$ clusters.
3. We do the second part until we get to a cluster that contains all of the N points

This is called the **Agglomerative hierarchical clustering**. Now we need to quantify what we mean by similarity (linkage) of two clusters.

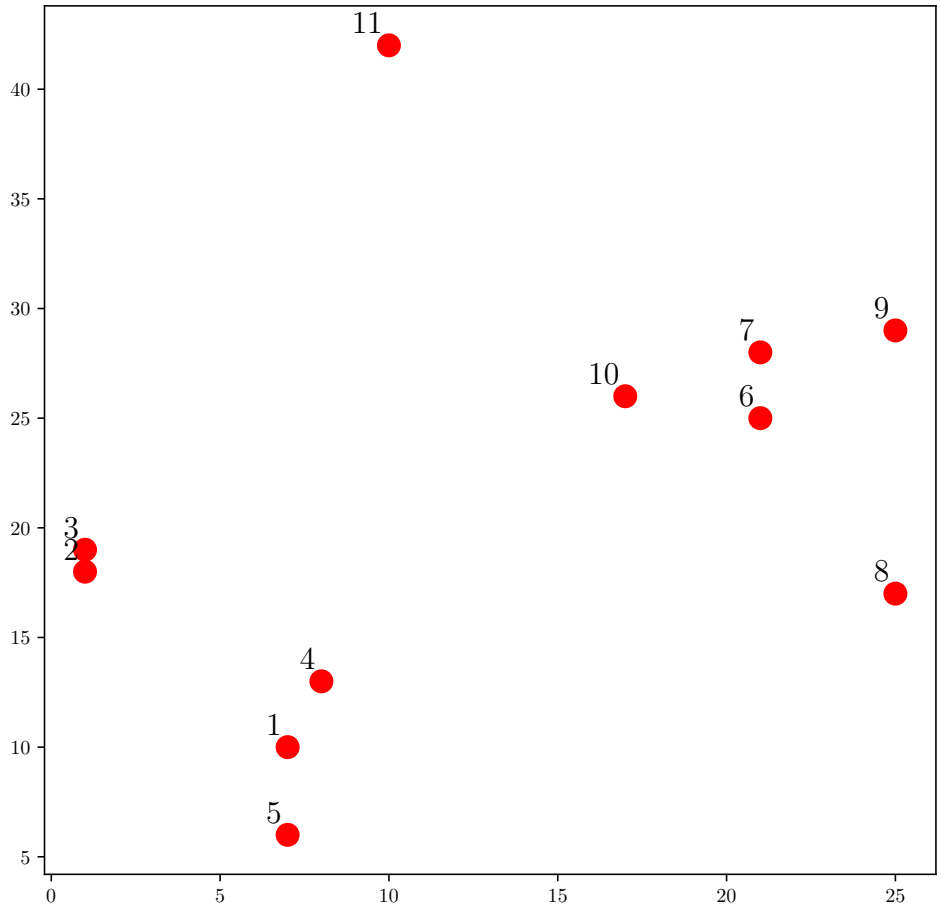
These are some Linkage Metrics:

1. **Single-linkage:** It is the shortest distance between any member of one cluster and any member of another.
2. **Complete Linkage:** It is the greatest distance between any member of one cluster and any member of another.
3. **Average Linkage:** It is the average distance between any member of one cluster and any member of another.

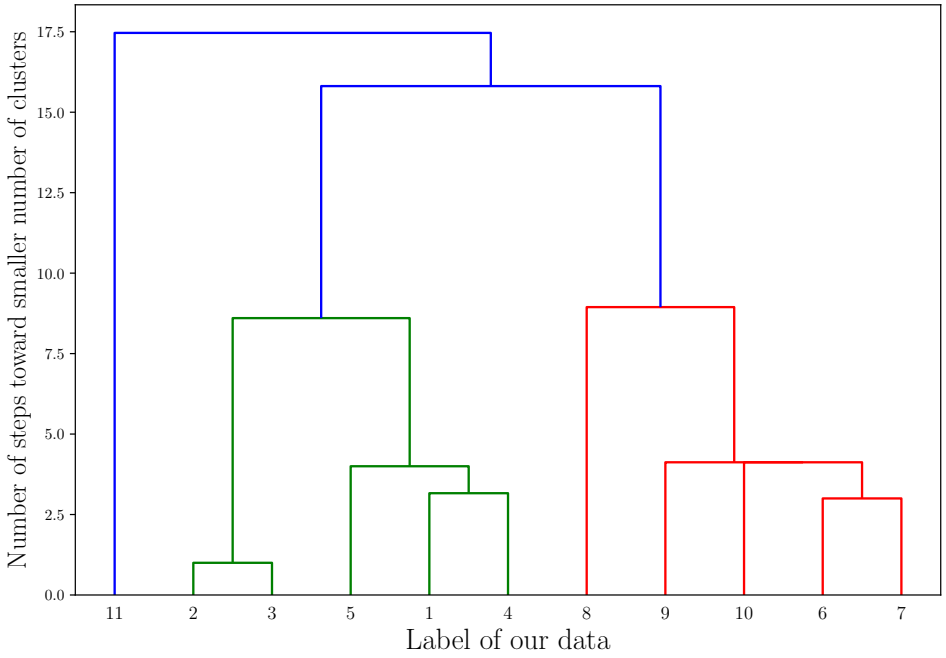
Now let's take a look at the following example:

Imagine the following set of 11 points that we we want to the

clustering.



After running the above algorithm we get a **dendogram** of the hierarchical clustering as the figure below.



This is our resulting dendogram that contains all of the possible clustering following the algorithm mentioned above. It should be noted that this is completely deterministic for a given linkage metric. Also, the answer might not be optimal, since the hierarchical clustering is a greedy algorithm. In other words, your answer could be optimal but not globally optimal. However, this can become very slow and the number of calculations grows with $O(N^3)$ which can make the algorithm useless quickly by increasing the number of points. However for some linkage metrics (i.e. single-linkage), there exist $O(N^2)$ algorithms but still not very fast. One way to avoid such burdensome calculation is to employ the k-means clustering described in the following section.

k-means clustering

Here we divide space around a point into cells. If you know how many clusters you want then this is going to be a better choice since it is much faster. k is the number of clusters we want. The algorithm for this clustering method is as follows:

1. Start by randomly choosing k examples as our initial centeroid. The number of clusters, k, should be specified by the user

2. create k clusters by assigning examples to closest cendroid. Properties of cluster members here will be similar to those of their associated centroid. Here the "nearest" is indicated by proximity measure. Euclidean distance measurement is the most common. The Eulcidean distance between two data points$X(x_1, X_2, ..., x_n)$ and $C(c_1, C_2, ..., c_n)$ with n attributes is given by

   $$\sqrt{(x_1 - c_1)^2 + (x_2 - c_2)^2 + ... + (x_n - c_n)^2}$$

   This leads to partitioning the data.

3. Assign a new values for the centeroid: The average of previous cluster configuration. Basically, for each cluster a new centroid can be defined. This is the most representative of all the data points. Here we minimize the sum of square errors (SSE) of all data points in a cluster to the centroid of the cluster. This is to minimize the SSEs of individual

clusters

$$\sum_{i=1}^{k} \sum_{x_j \in C_i} ||x_j - \mu_i||$$

[2]$WhereC_i$ is the ith cluster, j are the data points in a given cluster, μ_i is the centroid for the i^{th} cluster and x_j is the specific data point.The centroid with minimum SSE for the given cluster i is the new mean of the cluster, which is calculated as

$$\mu_i = 1/j_i \sum_{x \in C_i} X$$

whwre X is the data vector $(x_1, ..., x_n)$. In the case of k-means clustering the new centroid will be the mean of all data points.

4. Go to (2) while centroids are changing, else break. We have our final configuration.

Notice that the algorithm is not deterministic and the result can change by choosing another initial configuration. Also, for N and k clusters we need to find kN distances. Which is much smaller than $O(N^2)$. Also, we will see that we do not need to go through many iteration to converge. (Relative to N)

3.12 Regression

The basic idea of regression is to find a function that describes and predicts the value of a target variable when given the value of the

predictor variables.

Suppose we want to define the relation between the number of rooms in a house and the price of the house (the median price).On average, houses with more rooms are more expensive. Therefore, by drawing a line between the points in a sample, one could predict the price of any house given the number of rooms in that house. This is the linear regression problem.If there are two predictors, then the fit is best done by a surface in three dimensional space. Given the distribution of the points, a number of straight lines can be passed through them. The point is which is the optimum.

We use the concept of an error function. In the single predictor case (linear regression), the predicted value $\hat{y}$ for a value of x that exists in the data set is

$$\hat{y} = b_0 + b\ x$$

Then the error is estimated as

$$e = y - \hat{y} = y - (b_0 + b_1 x)$$

which defines the error at a single location $x, y)$ in the data set. We compute error for all the existing data points to come with an aggregate error. We find the square of the error to make sure it is a positive entity

$$\frac{J}{n} = \frac{\sum e^2}{n} = \sum (y_i - \hat{y})^2 = \frac{\sum (y_i - b_0 - b_i x_i)^2}{n}$$

where n is the number of points in the data set and J the total square error. We now need to find the best combination of (b_0, b_1)

that would minimize the error function. This is a minimization problem.This is done by taking the derivatives of the error function with respect to b_0 and b_1 and set them to zero. This gives two equations with two unknowns that can be solved. The solutions for b_1 and b_0 is

$$b_1 = (\sum x_i y_i - \bar{y} \sum x_i)/(\sum x_i^2 - \bar{x} - \sum x_i)$$

and

$$b_0 = (\bar{y}_i^2 - \bar{x} \sum x_i y_i)/(\sum x_i^2 - \bar{x} \sum x_i)$$

Normally linear regression algorithms use an optimization technique called gradient descent to identify the combination of b_0 and b_1 which will minimize the error function. The advantage of this is that even with multiple predictors, the algorithm works well.

The linear regression can be generalized to multiple dimensions where there are many predictors. For example, for some points, there seems to be no relation between the number of rooms and the price of the house. In this case it is clear an extra variable is needed (like location of the house) to best explain the data. In this form, we build a vector that contains different values of b, the coefficients and solve for those.

Logistic Regressions

In linear regression, we find a function to fit the $x's$ that vary

linearly with y and use the function to predict y for any value of x. The assumption here is that both the predictor and target variables are continuous. Now, what happens if this is not continuous? for example, sometimes the data are so that for values of x we only have $y = 1$ or 0. Here, we cannot express the data with a straight line. An S shaped curve would be a better expression of data for this instance. This is called "sigmoid" curve.Logistic regression is therefore the process of obtaining an appropriate non-linear curve to fit the data when the target variable is discrete.

To measure the sigmoid curve, we consider the following steps:

If y is an event (response pass/fail)
and p is the probability of the event happening ($y = 1$)
Then $(1-p)$ is the probability of the event not happening ($y = 0$)
And $p/(1-p)$ are the odds of the event happening.

The logarithm of the odds, $log(p/(1-p))$ is linear in the predictor x and is called *logit function*. The *logit* can be expressed as the linear function of the predictors x

$$login = log(p/(1-p)) = b_o x + b_1$$

the logit can take any value. For each predictor, the logit can now be calculated. From the logit, it is then easy to compute the probability of the response y (occuring or not occuring) as

$$p = e^{logit}/(1 + e^{logit})$$

This logistic regression gives the probability of y occuring ($y = 1$) given specific values of x.

How do we implement logistic regression?

From the data $x's$ are known. Therefore, one can compute p for any x value. However, to do this, one needs to have b coefficients first. If on starts with trial of values for b. Given a training data set, one can compute

$$p^{\gamma} \times (1 - p)^{(1-\gamma)}$$

where γ is the original outcome variable that can take values of 1 or 0 and p is the probability estimated by the logit. For a specific training sample, if the actual outcome was $\gamma = 0$ and teh model estimate of p was high (say 0.9), that is the model was wrong, then this quantity reduces to 0.1. If the model probability was low (say 0.1), that is the model was good, this quantity incresaes to 0.9. Therefore, this quantity which is a simplified form of a likelihood function is maximized for good estimates and minimized for bad estimates. By computing the summation of the simplified likelihood function across all the training data, then a high value indicates a good model or a good fit. Often gradient descent is used to search for coefficients b with the aim of maximizing the

correct estimate of $p^{\gamma} \times (1-p)^{(1-\gamma)}$ summed over all training sample.

3.13 Artificial Neural Network

In supervised learning one models the relationship between the input and output variables. The neural network technique approaches this problem by developing a functional relationship between the input and output variable by resembling the working of neurons. This is demonstrated as follows: Consider the model

$$Y = 1 + 2X_1 + 3X_2 + 4X_3$$

Where Y is the output and X_1, X_2, X_3 are the input attributes with 1 being the intercept and 2,3 and 4 the scaling factors or coefficients for the input attributes X_1, X_2 and X_3 respectively. This model is represented as in Fig 1. Here X_1 is the input value that passes through a node (shown by a circle here). The value of X_1 is multiplied by its weight, which is 2. Similarly other attributes (X_2 and X_3) go through the nodes with a transformation. The last node has no input variables and just has the intercept. The values from all the nodes are collected in an output node that gives the predicted output Y. This is an artificial neural network (ANN). Neural Networks learn from adaptive adjustments of weights between the nodes. This is a computational model inspired by biological nervous systems.

In ANN the first layer of nodes closest to the input is called the input layer. The last layer of nodes is called the output layer. The

output layer acts as an aggregation function and can also have a transfer function which scales the output to the desired range. The output layer performs an activation function using the aggregation and transfer functions. This simple ANN with two layer topology is called a *perceptron*. A perceptron is a feedforward NN where the input moves in one direction with no loops in the topology. Apart from the input and output layers, there are hidden layers that contain a layer of nodes that connect input from previous layers and apply an activation function. The output in this case is calculated by complex manipulations of many hidden layers. Figure shows a NN with one hidden layer. In principle, one could use a topology with many hidden layers as well as looping where the output of one layer is used as input to the previous layer.

The activation function in the output layer is a combination of an aggregation function and a transfer function. Transfer functions commonly used are: sigmoid, normal bell curve, logistic or linear.

How an ANN Works?

An ANN learns the relationship between input attributes and the output class label through the back propagation techniques. The key task is to find the weights of the links. This closely follows the biological neurons. The model uses training records to estimate the error between the predicted and the actual output. The error is then used to adjust the weights to minimize the errors for the next training error. This step is repeated until the error falls within the acceptable range. The steps taken in setting up an ANN are:

Step 1: Determine the topology and activation function:

Imagine a dataset with three numeric input attributes (X_1, X_2, X_3) and one numeric output (Y). A topology with two layers and a simple aggregation activation function is being used. There is no transfer function here.

Step 2: Initiation:

Assume the initial weights for the four links- 1, 2, 3 and 4. Take an example model and a training record with all the inputs as 1 and the known output as 15. Therefore, $X_1, X_2, X_3 = 1$ and $Y = 15$. This way we initiate the first training record.

Step 3: Calculating Error:

The predicted output of the record can now be calculated through a feed-forward process when the input data passes through the nodes and the output is calculated. The predicted output $\bar{Y}$ according to the current model is:

$$1 + 1 \times 2 + 1 \times 3 + 1 \times 4 = 10$$

The difference between the training and predicted output is $e = Y - \bar{Y} = 15 - 10 = 5$.

Step 4: Weight adjustment:

This is where the learning actually takes place. The error calculated in the previous step is fed back from the output node to all other nodes in the reverse direction. The weights of the links

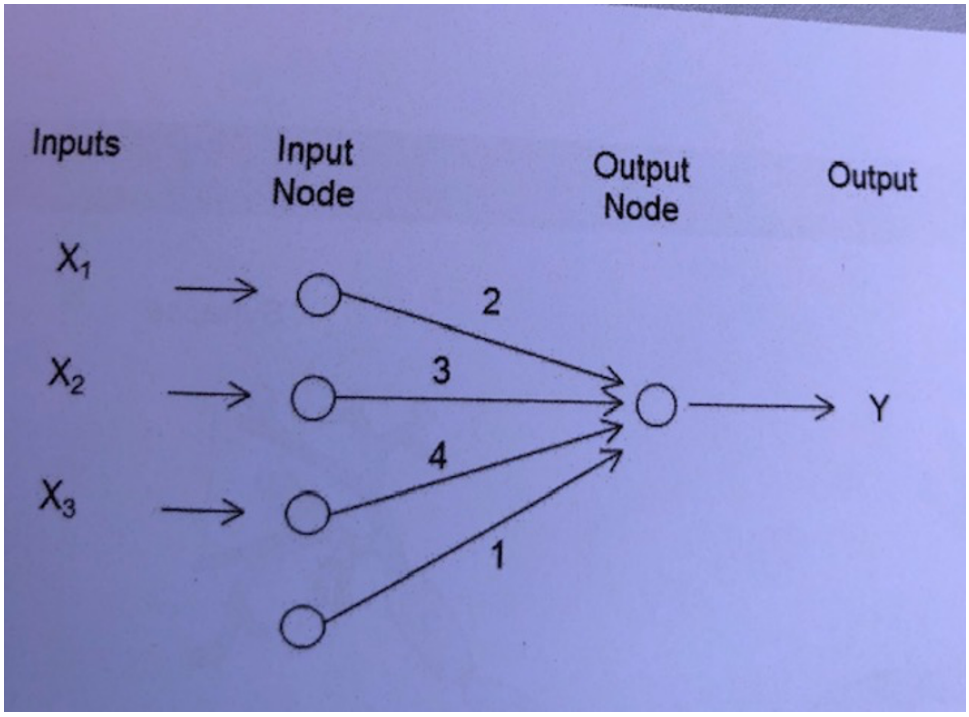
are adjusted from their old value by a fraction of the error. The fraction λ that is applied to the error is called the *learning rate*. It takes values fro 0 to1. A value close to 1 requires drastic change to the model for each training record and a value close to 0 results in smaller changes and less correction. The new weight (w) of the link is the sum of the old weight (w') and the product of the learning rate and proportion of the error ($\lambda \times e$)

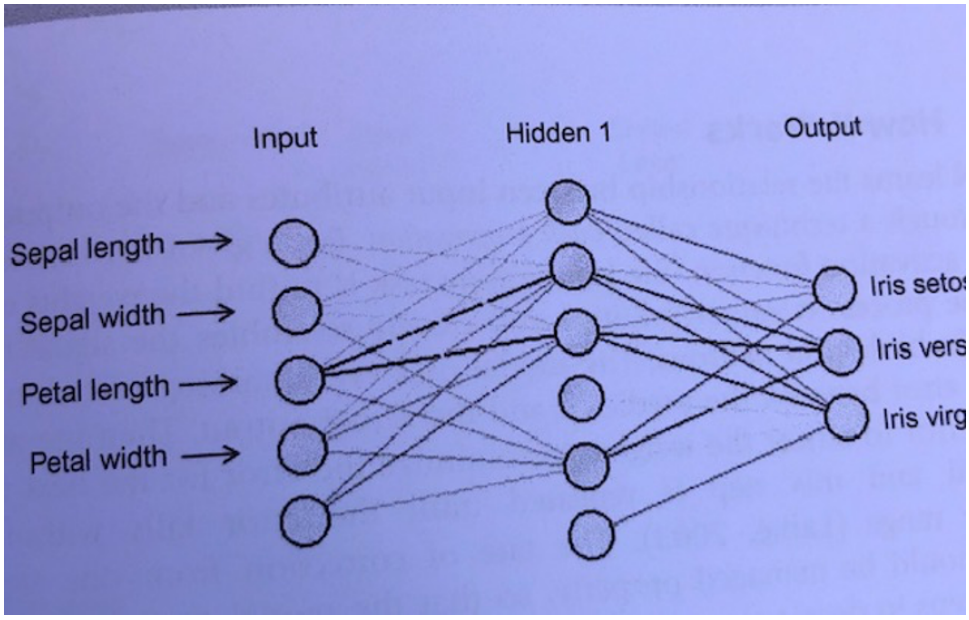
$$w = w' + \lambda \times e$$

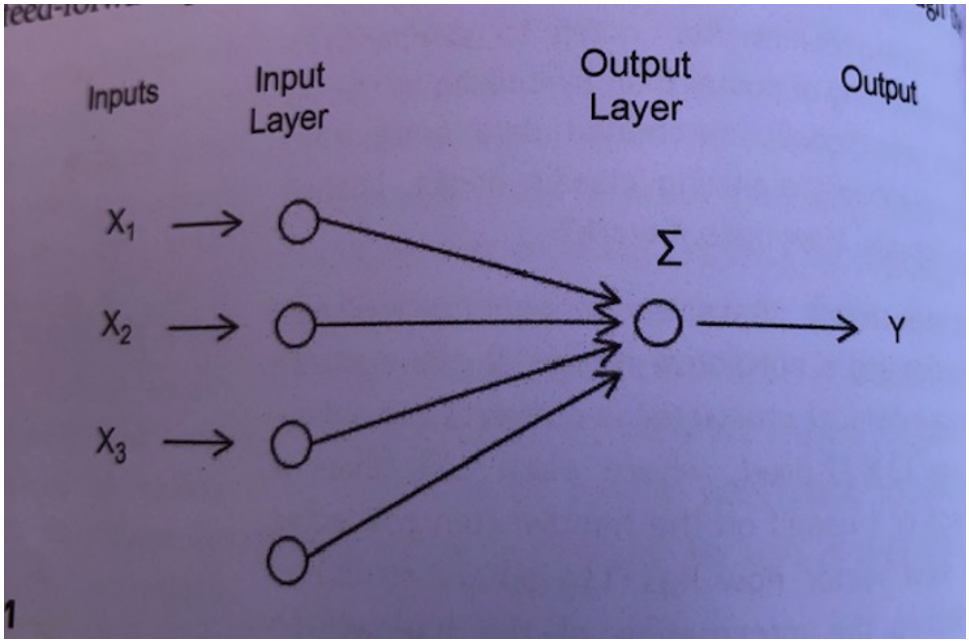
The most important step is the choice of λ. By starting λ close to 1 and reducing its value while training each cycle, any outlier record in the training cycle will not degrade the model.

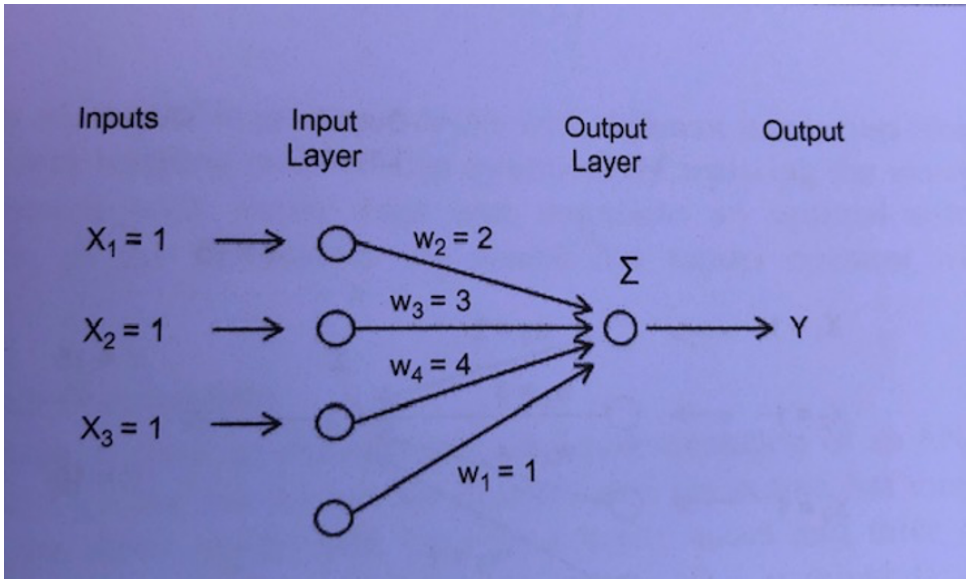
The current weight of the first link is $w_2 = 2$. Assume the learning rate is 0.5, the new weight will be $w_2 = 2 + 0.5 \times 5/3 = 2.83$. Here the error is divided by 3 because it is backpropagated to three links from the output node. The weight will be adjusted for all the links. In the next cycle a new error will be computed for the next training record. The same training example is repeated until the error rate is less than a threshold.

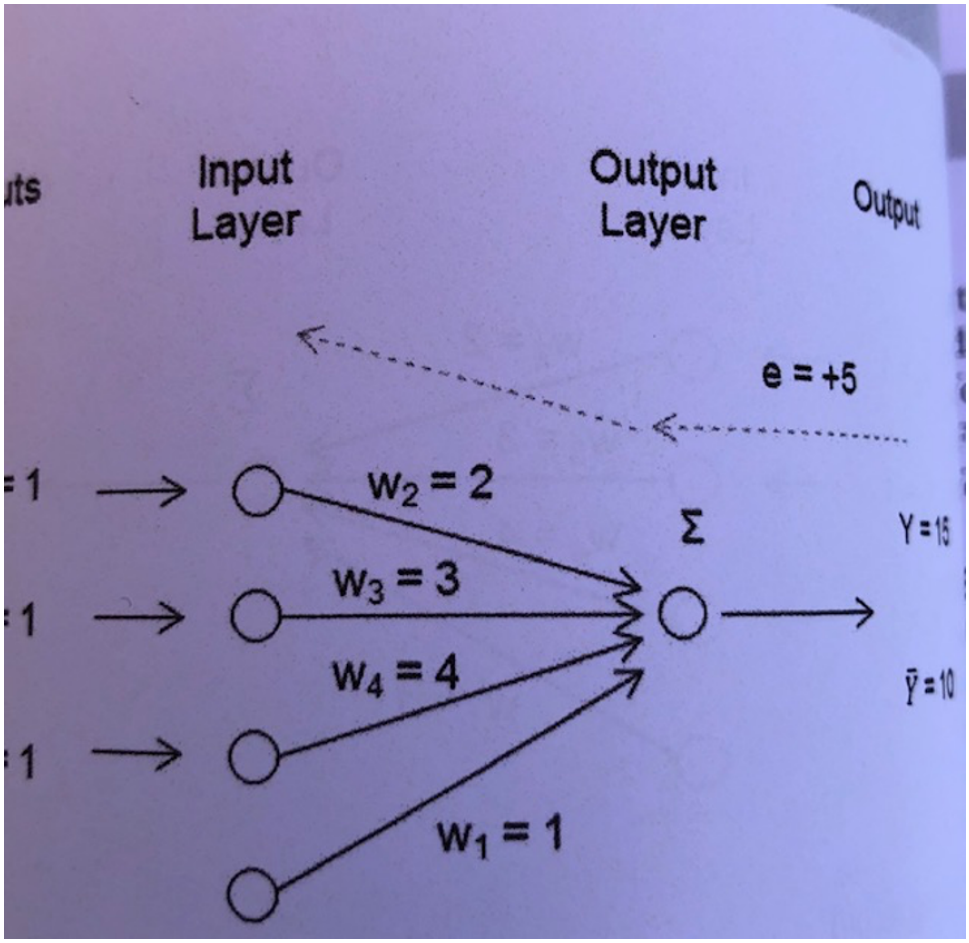
In real cases, there will be multiple hidden layers and multiple output nodes, one for each nominal class values.


Inputs
Input Node
Output Node
Output
X_1
X_2
X_3
2
3
4
1
Y


Input
Hidden 1
Output
Sepal length
Sepal width
Petal length
Petal width
Iris seto
Iris vers
Iris virg

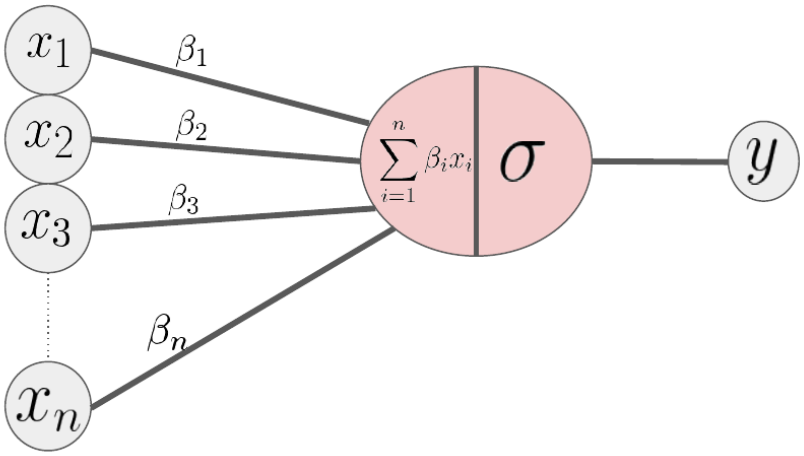

Inputs
Input Layer
Output Layer
Output
X1
X2
X3
Σ
Y


Inputs
Input Layer
Output Layer
Output
X1 = 1
X2 = 1
X3 = 1
w2 = 2
w3 = 3
w4 = 4
w1 = 1
Σ
Y



3.14 Perceptron

A perceptron consist of weights βs and activation function σ applied on the input data as shown in figure below. It can be shown that a perceptron is a linear classifier which means the decision boundaries of the classifier is a linear hyperplane.



Let's start with a two class classification problem with the following labels:

$$y = \{-1, 1\}$$

We need to find the best $\boldsymbol{\beta}$ and β_0 that makes the classifier optimal. We have the following linear decision boundary for the classifier:

$$\boldsymbol{\beta^T x} + \beta_0$$

And the following classification rule:

$$y = \sigma(\boldsymbol{\beta^T x} + \beta_0) = \text{sign}(\boldsymbol{\beta^T x} + \beta_0)$$

Now we are going to find the objective function, one measure could be the number of misclassified points but in order to make the problem easier we can use the sum of distances from the decision boundary for the misclassifed points:

$$L(\beta) = -\sum_{i \in S_{MC}} y_i(\boldsymbol{\beta^T x_i} + \beta_0)$$

where S_{MC} is the set of misclassified points. Multiplying the signed distance by the classification makes the quantity in the sum to be always positive which is what we need since we are only interested on the distant from the boundary and there is no new information about the sign of this distance. (Can be chosen arbitrarily w/o changing the problem at hand)

Now let's look at some geometric argument for the definitions mentioned. Let's assume $\boldsymbol{x_0}, \boldsymbol{x_1}, \boldsymbol{x_2} \in DB$ which DB is the decision boundary and $\boldsymbol{x}$ is an arbitrary point. Then by definition we have the following:

$$\boldsymbol{\beta^T x_1} + \beta_0 = \boldsymbol{\beta^T x_2} + \beta_0 = 0$$
$$\boldsymbol{\beta^T}(\boldsymbol{x_1} - \boldsymbol{x_2}) = 0$$
$$\boldsymbol{\beta} \perp (\boldsymbol{x_1} - \boldsymbol{x_2})$$

$$\boldsymbol{\beta^T x_0} + \beta_0 = 0$$
$$\beta_0 = -\boldsymbol{\beta^T x_0}$$

$$\text{Signed distance of } \boldsymbol{x} \text{ from the hyperplane } = \boldsymbol{\beta}.(\boldsymbol{x} - \boldsymbol{x_0})$$
$$= \boldsymbol{\beta^T}(\boldsymbol{x} - \boldsymbol{x_0}) = \boldsymbol{\beta^T x} + \beta_0$$

Now we can optimize the objective function:

$$L(\boldsymbol{\beta}, \beta_0) = -\sum_{i \in S_{MC}} y_i(\boldsymbol{\beta^T x_i} + \beta_0)$$

$$\begin{cases} \frac{\partial L}{\partial \boldsymbol{\beta}} = -\sum_{i \in S_{MC}} y_i \boldsymbol{x_i} \\ \frac{\partial L}{\partial \beta_0} = -\sum_{i \in S_{MC}} y_i \end{cases}$$

Now we can find the local minimum iteratively by updating $\boldsymbol{\beta}$ according to the gradient at every step.

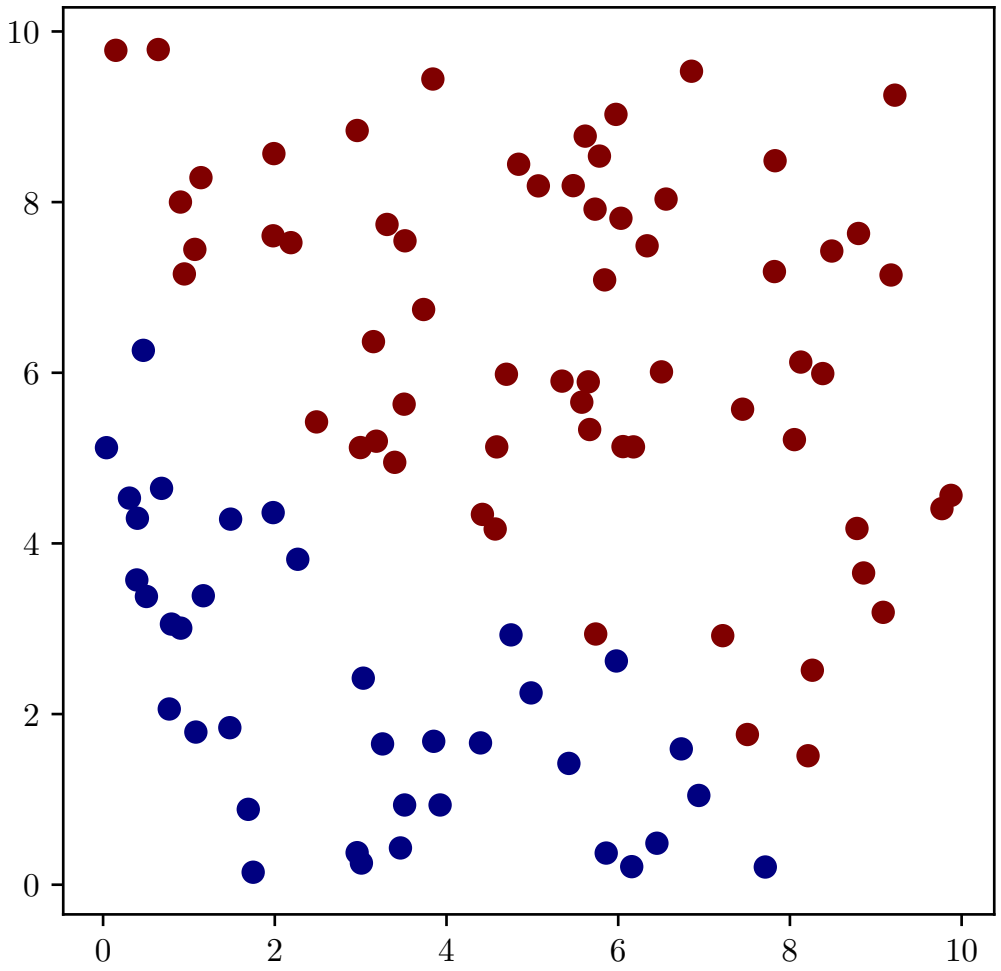
$$\begin{bmatrix} \boldsymbol{\beta}^{t+1} \\ \beta_0^{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}^{t} \\ \beta_0^{t} \end{bmatrix} - \eta \begin{bmatrix} \frac{\partial L}{\partial \boldsymbol{\beta}} \\ \frac{\partial L}{\partial \beta_0} \end{bmatrix}$$

For a single point update we can write: (All from the misclassified points at each step)

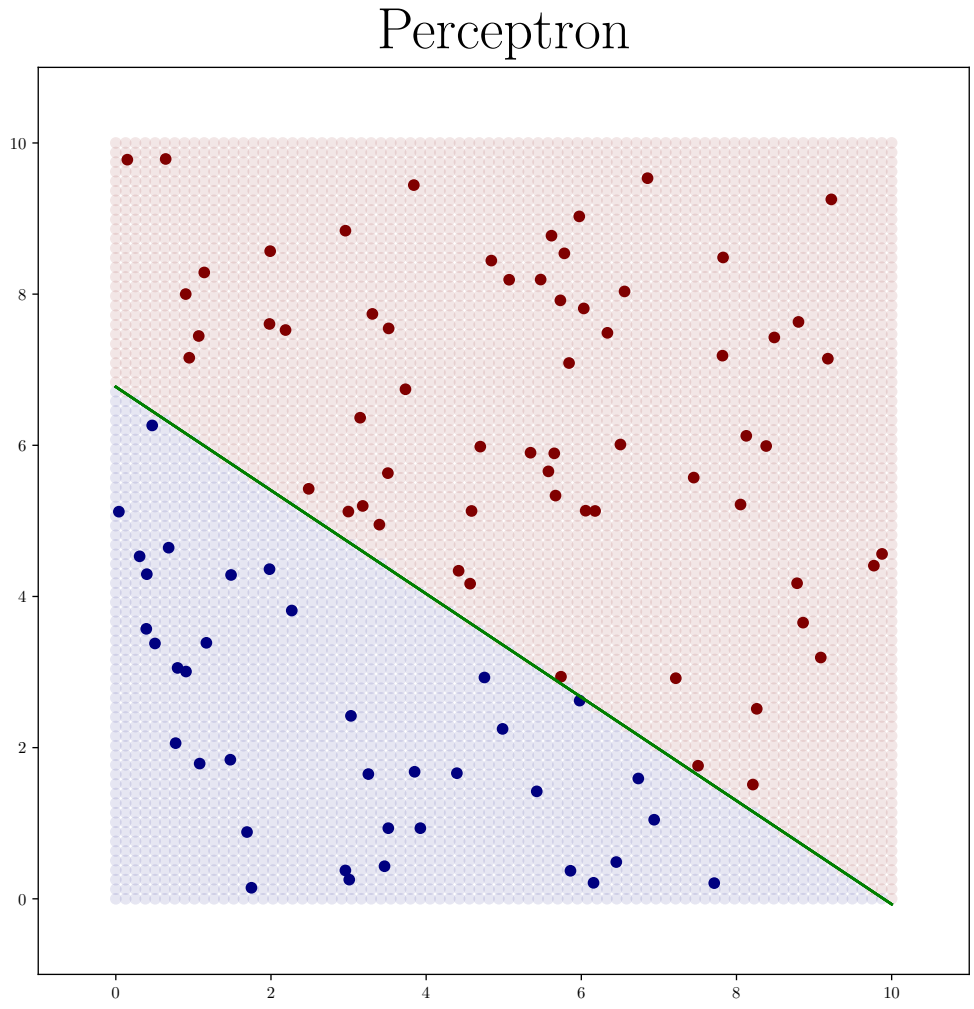
$$\begin{bmatrix} \boldsymbol{\beta}^{t+1} \\ \beta_0^{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}^{t} \\ \beta_0^{t} \end{bmatrix} + \eta \begin{bmatrix} y_i \boldsymbol{x_i} \\ y_i \end{bmatrix}$$

where η is the learning rate. Using the above algorithm and choosing a threshold for our expected error rate, we can iterate until the threshold is met.

Now let's look at the following example for a binary classification problem:



After applying the algorithm and letting the perceptron learn iterativly according to the update rule we had above, we can build a perceptron classifier.



One important feature we should remember for the perceptron is that the decision boundary solution may not be optimal and is one solution to the classification problem. For example, if we have two classes that are very well separated we can have multiple decision

boundary that satisfy our threshold on the error rate.

3.15 Regression Models as Neural Networks

The linear regression models we discussed in the last lecture can be recast into an Artificial Neural Network (ANN). Figure 1 is the simple linear regression model shown as a network. This network is only two layer deep. It has an input layer and an output layer. Multiple regression models require additional nodes (one node for each feature) in the input layer. A logistic regression model can also be represented by a simple two-layer network model. We know that the output of logistic regression is not a single number but the probability of an event, p, rather than a real number value as in linear regression. Therefore, we need to transfer the output variable in such a way that its domain ranges from 0 to 1 instead of $-\infty$ to $+\infty$. From the previous lecture we therefore have

$$log(\frac{p}{1-p}) = b_0 + b_1\ x_1$$

and

$$p = (\frac{1}{1+e^z})$$

where $z = b_0 + b_1 x_1$. The output can be more generally summarizes as

$$p(\gamma) = \sigma(z)$$

Therefore, the output node in the above network can be expressed as in Figure 2. The above equations represent the sigmoid function. The sigmoid domain is [0,1] for $z = [-\infty, \infty]$ so that any arbitrary values for b and x will result in $p(\gamma_n) = [0, 1]$. $p(\gamma_n)$ is the prediction from the logistic regression model for sample n, which needs to be compared with the actual class value p_n for that sample in order to evaluate the model.

3.16 Gradient Descent

In the regression analysis, an error function or cost function was introduced which will be used here. By taking the log of the cost function, we convert the product to summation when one needs to compute across all samples. γ_n is the probability based on the model and can range between $[0 - 1]$ and p_n is the target which is either 0 or 1. N is the number of samples

$$J = -\sum_{n=1}^{N}[p_n log(\gamma_n) + (1 - p_n) log(1 - \gamma_n)]$$

Where J is the cross-entropy cost function. The negative sign is added in the front with the aim of minimizing the value. The problem therefore reduces to finding the $b's$ so that to minimize the function. Gradient descent will be used to iteratively find the location where it is minimum.

The cross entropy cost function is expressed in terms of the weights

b, by substituting

$$\gamma = \sigma(z) = \frac{1}{(1 + e^{-z})}$$

where $z = b_0 x_0 + b_1 x_1$. The weights, b, can now be found by minimizing J, expressed in terms of b by differentiating this and setting it to zero

$$\frac{dJ}{db} = 0$$

Therefore,

$$\frac{dJ}{d\gamma} \times \frac{d\gamma}{dz} \times \frac{dz}{db} = 0$$

We now calculated each of these derivatives:

Step I:

$$\frac{dJ}{d\gamma} = (\frac{p_n}{\gamma_n}) - (\frac{1 - p_n}{1 - \gamma_n})$$

Step II:

$$\gamma = \frac{1}{1 + e^{-z}}$$

$$\frac{d\gamma}{dz} = \frac{-e^{-z}}{(1 + e^{-z})}$$

or

$$\frac{d\gamma}{dz} = \frac{\gamma_n}{(1-\gamma_n)}$$

Step III, given $z = b_0x_0 + b_1x_1$ and since often x_0 is set to 1,

$$\frac{dz}{db} = x_i \quad i = 1, 2, \ldots, n$$

Substituting these into the equation we get

$$\frac{dJ}{db} = -\sum_{n=1}^{N}[(\frac{p_n}{y_n} - (\frac{1-p_n}{1-\gamma_n})] \times [\gamma_n(1-\gamma_n)] \times x_1$$

which simplifies to

$$\frac{dJ}{db} = -\sum_{n=1}^{N}(p_n - \gamma_n)x_1$$

and in matrix form where B, P, X and Y are vectors it becomes

$$\frac{dJ}{db} = (P_n - Y_n)^T.X$$

Where B is a $(d \times 1)$ vector whee d is the number of independent variables and the other three vectors are $(n \times 1)$ where n is the number of samples.

In gradient descent, rather than setting the derivative to zero, an iterative approach is adopted to solve for the vector B. We start with an initial value for weight vector, B_j, where j is the step size (called the learning rate) and use the slope in the above equation to iteratively reach the point where J is minimized

$$B_{j+1} = B_j – Learning\ Rate \times [P_n – Y_n]^T.X$$

The iteration would stop when the incremental difference between B_j and B_{j+1} is very small. The main thing here is to calculate the gradient of the cost function, dJ/dB, and use it to calculate the B matrix

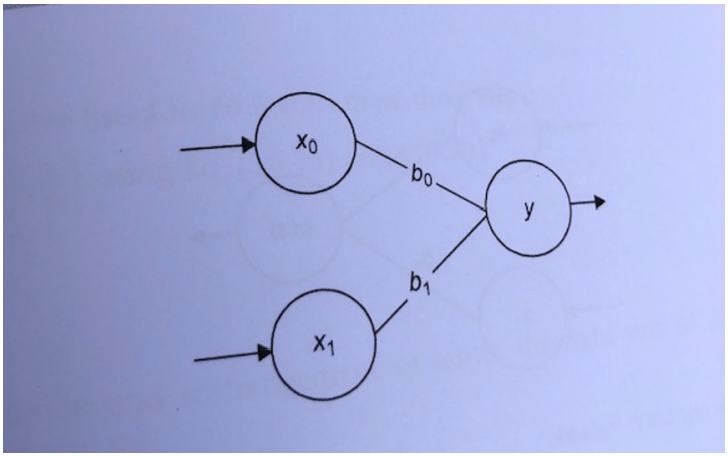
$$B_{j+1} = B_j + Learning\ Rate \times \frac{dJ}{dB}.X$$

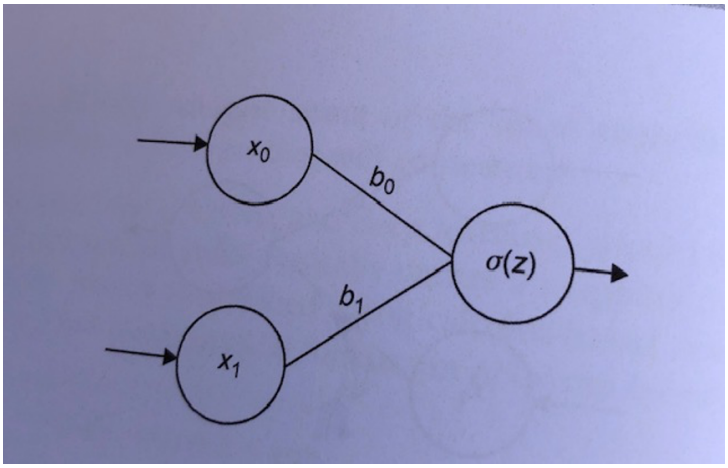
The iterative components of the gradient computation for logistic regression and gradient descent have both the following form:

(predicted Vector – Target vector).(Matrix of input Data)

The main difference between the two is the way the two are calculated. In logistic regression these are evaluated using the sigmoid transformation ($\gamma = \sigma(b_0 + b_1 x)$ and in linear regression a unit transformation ($\gamma = b_0 + b_1 x$) is used, that is no scaling or transformation is applied to the computed output.

The above is basically the concept of an activation function in ANN and deep learning. This is basically a weighted averaging scheme. If the weighted average (b_0+b_1x) crosses a present threshold, the output evaluates to 1, if not, it evaluates to 0. This is what happens if the activation function is a sigmoid.





3.17 Backpropagation

Weighted averaging allows a full ANN to be conceptualized starting from simple two-layered models of logistic and linear regres-

sion. Gradient descent incrementally updates the weights based on the output generated from a previous iteration. In the first iteration the weights are chosen randomly. The question is: can the process be made more efficient by choosing a series of starting weights in parallel? Can one start with building three logistic (or linear) regression models in parallel so that instead of (b_1, b_2) as the starting weight, there are three sets (b_{11}, b_{21}), (b_{12}, b_{22}) and (b_{13}, b_{23})? Here the first subscript refers to the input node or feature and the second subscript refers to the node in the intermediate or "hidden" layer. This is shown in Figure 3. Here one arrives at the output with smaller number of iterations by optimizing the weights. Finally, the output from the hidden layer is once again weight-averaged by three more weights (c_1, c_2 and c_3) before the final output is computed.

As is clear in Figure 3, with the "hidden layers" in place, the output is computed from left to right: that is $\sigma(z_1)$, $\sigma(z_2)$, $\sigma(z_3)$, are computed and weights for c_1, c_2 and c_3 assumed to compute the final sigmoid $\sigma(v)$. This would be the first iteration. The output is now compared to the correct response and the model performance evaluated using the cross-entropy cost function. The aim now is to reduce this cost (error) function by incrementally updating the weights and then go backward to update the weights $b_{11} : b_{23}$. This process is called "backpropagation". This is fundamental to understand how ANN works.

Classifying More than 2 Classes

Using logistic regression one can address a binary classification problem. However, most real world classifications require categorization into more than two classes- i.e. face recognition, numerals, text identification etc. In this case one needs a way to identify which of the several classes a given sample belongs to. In the previous discussion, there were only one output node and the probability that a given sample belonged to a class was obtained. In principle, one could add an output node for every class that needs to be categorized into and the probability that a sample belonged to that particular class could be computed (Figure 4). Here, Softmax function, that is a mathematical function representing normalized exponential function is used. This is a function that takes as the input a vector of K real numbers and normalizes it into a probability distribution consisting of K probabilities proportional to the exponentials of the input numbers. Prior to applying Softmax, some vector components might be negative or greater than one and may not sum to one but after applying Softmax, each component will be in the interval $(0, 1)$ with the components adding to one so that they could be interpreted as probabilities. This is used in ANN to map the non-normalized output of a network to a probability distribution over the predicted output classes.

$$\sigma(z)_i = \frac{e_i^z}{\sum_{j=1}^{K} e^{z_j}} \quad i = 1, 2, ..., K \; z = (z_1, \ldots, z_K)$$

Here we apply the exponential function to each element z_i of the input vector z and normalize these values by dividing by the sum

of all these exponentials. This ensures that the sum of the components of the output vector is 1.

Now consider the output layer with two nodes (instead of one)- one of class one and the other of class two (Figure 4). The probability of each class can be expressed as:

$$p(Y = class\ 1) = \frac{e^{z_1}}{e^{z_1} + e^{z_2}}$$

$$p(Y = class\ 2) = \frac{e^{z_2}}{e^{z_1} + e^{z_2}}$$

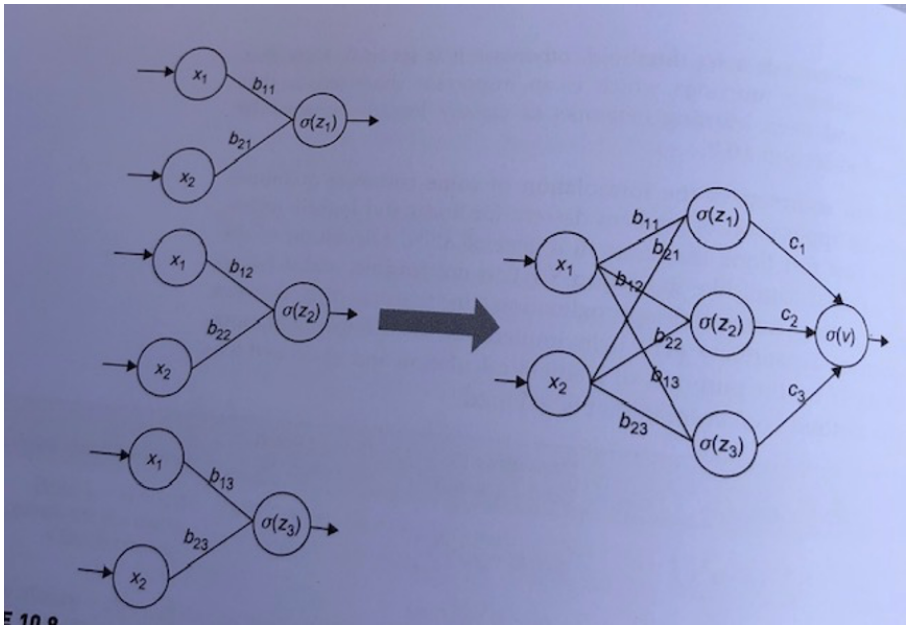
Dividing the numerator and denominator of the first class by e^{z_1}, we get

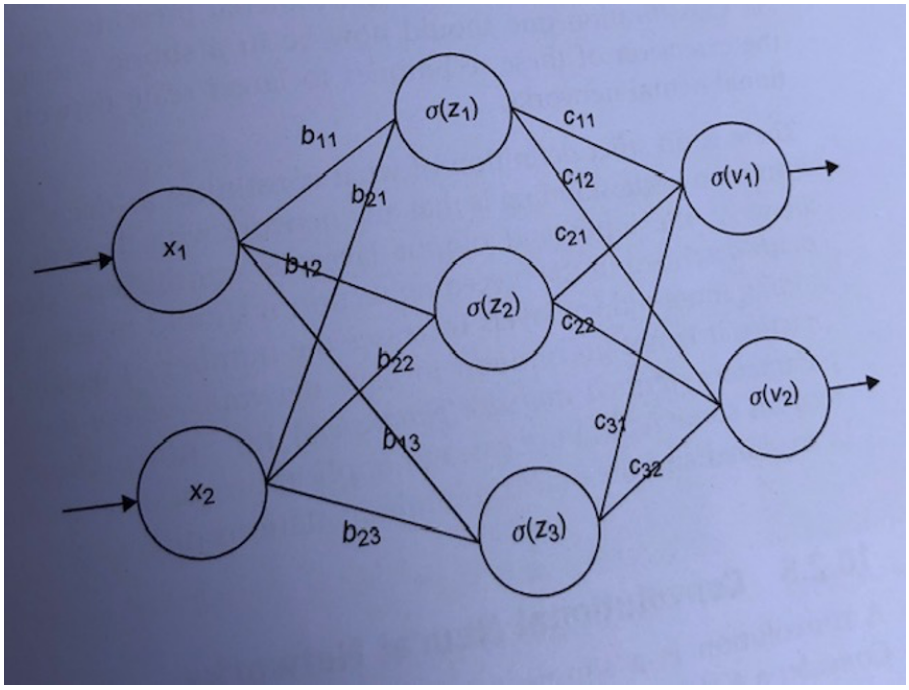
$$p(Y = class\ 1) = \frac{1}{e^{z_2 - z_1} + 1}$$

Here if $z_2 = 0$, this turns to the sigmoid binary classification problem. In the sigmoid, thresholding is used to decide about the binary output. In Softmax, the probability is calculated for each class.

Any network with more than three layers between the input and output is called *"Deep learning" network*. Adding more layers increases the number of weight parameters b_{ij}. The number of

weight parameters could run into millions in real applications of deep learning.





3.18 Convolutional Neural Net

We covered convolution in previous chapters. Briefly we remind it here. Consider a 6×6 matrix A and a 3×3 matrix B.

$$\text{A}= \begin{bmatrix} 10 & 9 & 9 & 8 & 7 & 7 \\ 9 & 8 & 8 & 7 & 6 & 6 \\ 9 & 8 & 8 & 7 & 6 & 6 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \text{B}= \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ \text{-1} & \text{-1} & \text{-1} \end{bmatrix}$$

The convolution between A and B, denoted by $A * B$, results in a new matrix, C, whose elements are obtained by a sum-product between the individual elements of A and B, as given below:

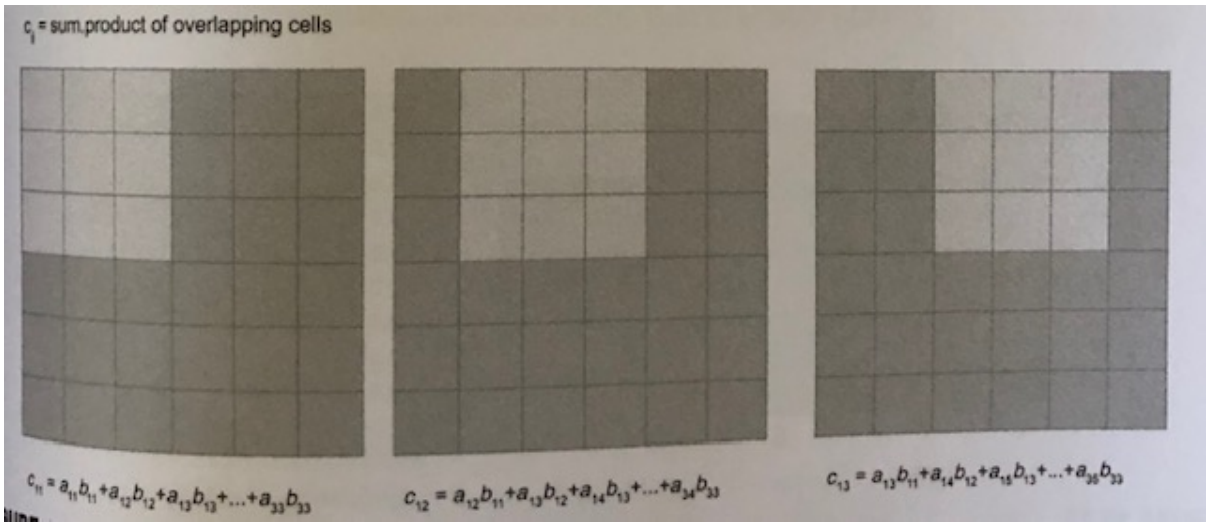
$$c_{11} = 10\times1+9\times1+9\times1+9\times0+8\times0+8\times0+9\times-1+8\times-1+9\times-1 = 3$$

$$c_{12} = 9\times1+9\times1+8\times1+8\times0+8\times0+7\times0+8\times-1+8\times-1+7\times-1 = 3$$

and so on.

This could be visualized as shown in the figures. Matrix B is the lighter shaded one which slides over the larger matrix A (darker

shade) from left (and top) to right (and bottom). At each overlapping position, the corresponding elements of A and B are multiplied and all the products are added to produce the corresponding elements for C. The resulting matrix C will be smaller in dimension than A and bigger than B.

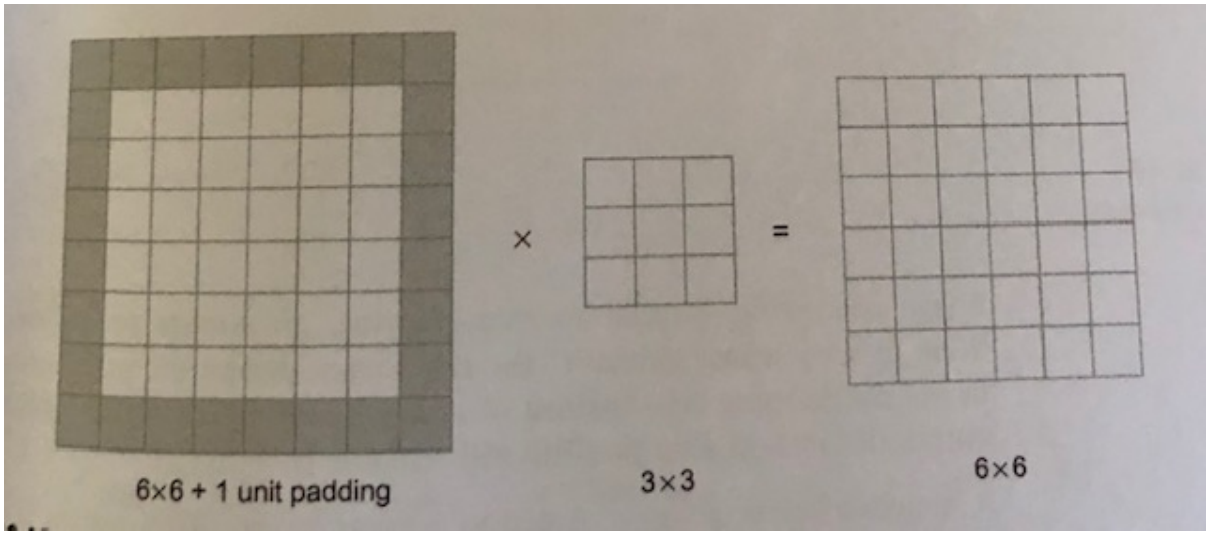


What is the physical meaning of this convolution process? Matrix A is a raw image where each cell in the matrix is a pixel value and matrix B is called a filter (or kernel) which, when convolved with the raw image, results in a new image that highlights only certain features of the raw image. The result is another pixel map C. Therefore, convolutions are useful in detecting basic features in images. The challenge is to determine the right filter for a given image. Machine learning can be used to optimize the filter shape (values). For example, in this example, finding the filter boils down to finding a $3 \times 3 = 9$ values for the matrix B in this example.

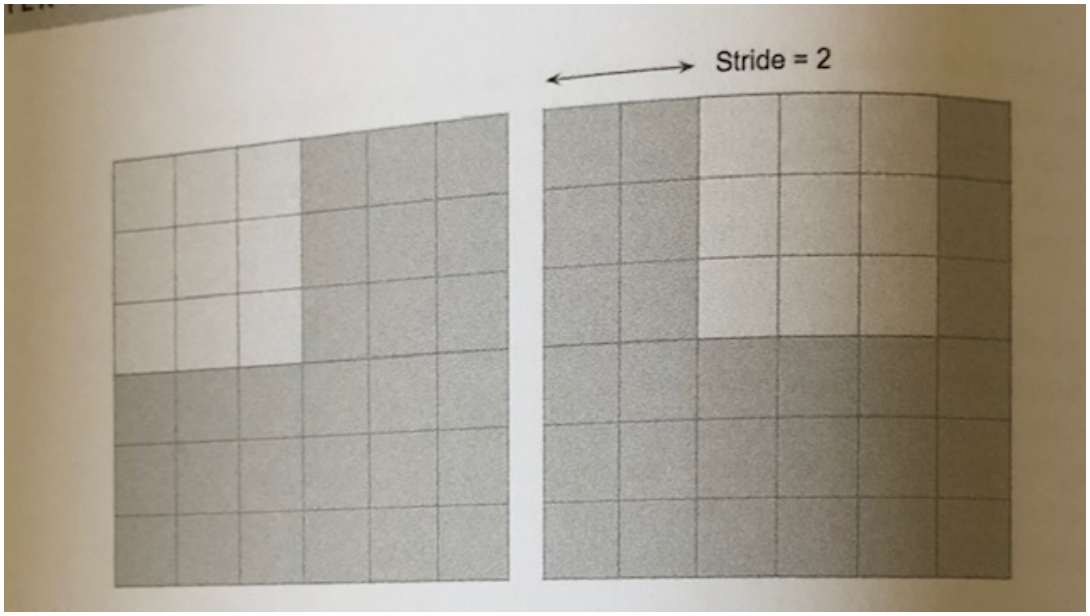
Matrix A is expressed in terms of $n_{width} \times n_{height}$ for a grey scale image. Color images have three channels: green, red and blue. Therefore, color images can be expressed as a 3-dimensional matrix $n_{width} \times n_{height} \times n_{channels}$.

Having dimensions of A and B, one could find the dimension of matrix C. Assuming $n_{width} = n_{height} = n$. Now if the filter is also a square matrix of size f, then the output C is square of dimension $n - f + 1$. In this case $n = 6$ and $f = 3$. Therefore, C becomes a 4×4 matrix.

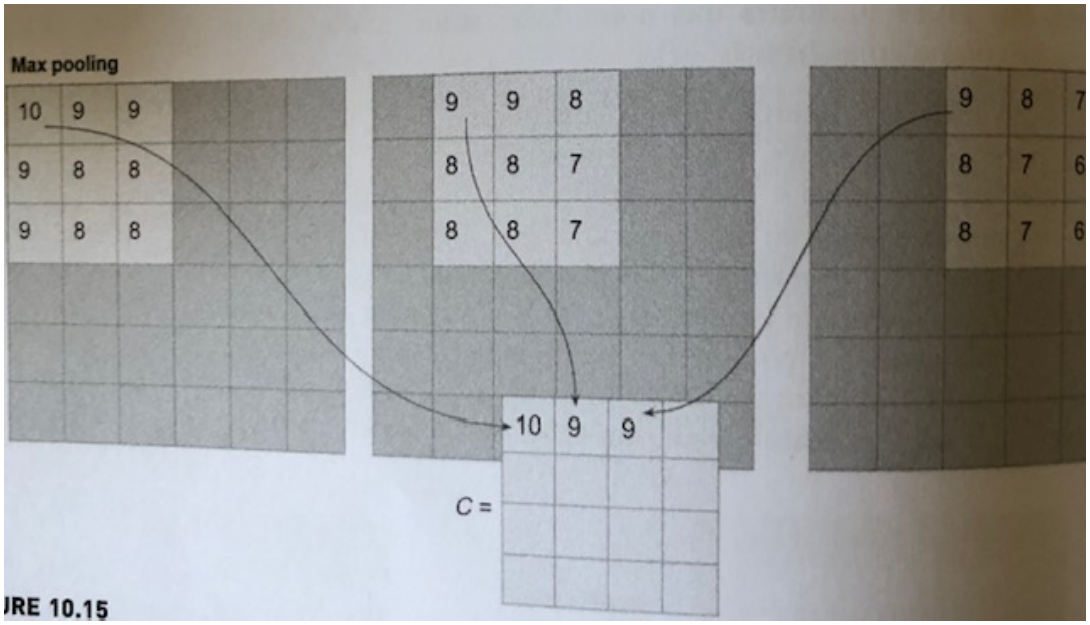
Since the convolution process reduces the raw image size, it is sometimes helpful to artificially increase the raw image size by having dummy pixels so that the original size is retained. This process is called padding. If p is the number of padded pixels, the output dimension is then given by $n + 2p - f + 1$. Therefore, in this example, the size of the C matrix becomes 6×6.



In the above example, the number of pixels advanced in each convolution was one. This is called the "stride". In principle, the filter could advance with more than one pixel- stride being 2 or 3. With stride s, the output dimension can be computed as $(n + 2p - f)/s + 1$.



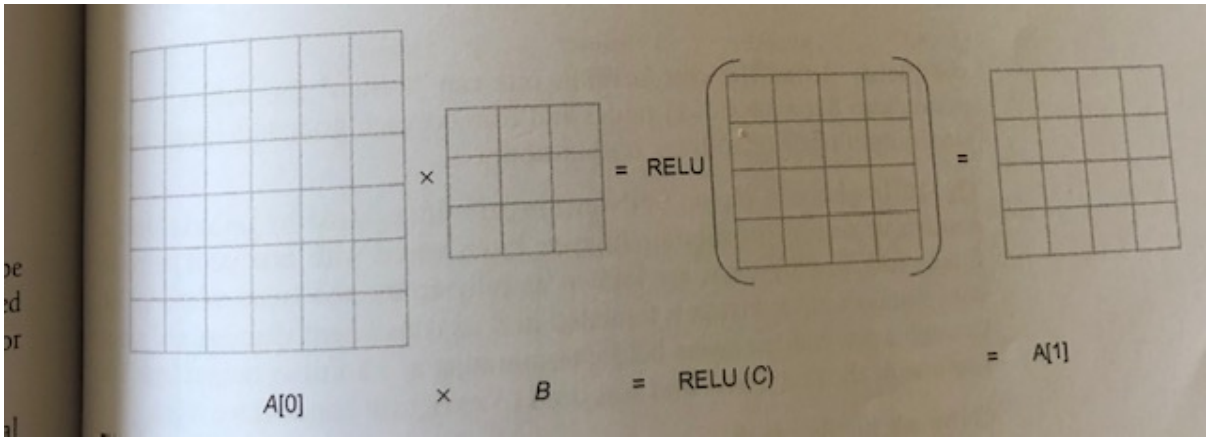
We discussed that convolution detects basic image features like edges. Sometimes instead of sum-product of the elements in the raw image and the filter, only could simply take the highest pixel value in the overlapping cell. This process is called max pooling. This is used in identifying the most prominent features in an image. Similar to max pooling, an average pooling could be used where the elements are the average of those in the overlapping cells.



A convolution is a linear function similar to what we used for neural networks. In this case the weights B are the pixels of the filter matrix and the inputs A are the image pixel values. The sum-product output C is analogous to $z = b_0 + b_1 x_1$. This process forms one convolutional layer. The output of this convolutional layer is sent to the next layer where it would be convolved with a different filter (weight) and sent to a regular layer of nodes, called "fully connected" layer.

In the following Figure $A[0]$ is the raw image and C is the result of the convolution with filter B. $A[1]$ is the result of adding a "bias" term to each element of C and passing them through an activation function. Here B is analogous to a weight function b and C is analogous to $\sigma(z)$. A NN backpropagation can be used to compute the elements of the weight matrix b with a similar

process used to determine the elements of the weight matrix B.



One could also apply multiple filters in the same layer. For example, B_1 can be the horizontal edge detector and B_2 a vertical edge detector. One can apply both on $A[0]$. Then the output C can be expressed as a volume. In this example, C has the dimension $4 \times 4 \times 2$ which is 2 matrices of 4×4 elements, each the result of a convolution of A and B_i $i = 1, 2$.

To determine the filter elements, we follow the same procedure as the linear regression before, by forming a cost function and minimizing it using descent gradient. The cost function is now depends on $3 \times 3 \times 2$ parameters, as in this example. To define the cost function one change the $4 \times 4 \times 2$ matrix C into 32 nodes and connect each node to a logistic regression output node or a softmax output nodes.

Convolutional Neural Networks are very popular in deep learning

because of their flexibility and efficiency. Figure shows a classic CNN architecture consisting of several convolutional layers followed by max pool layers and finally followed by fully connected layers where the last convolution output matrix is turned into its constituent elements and passed through a few hidden layers before terminating at a softmax output layer.

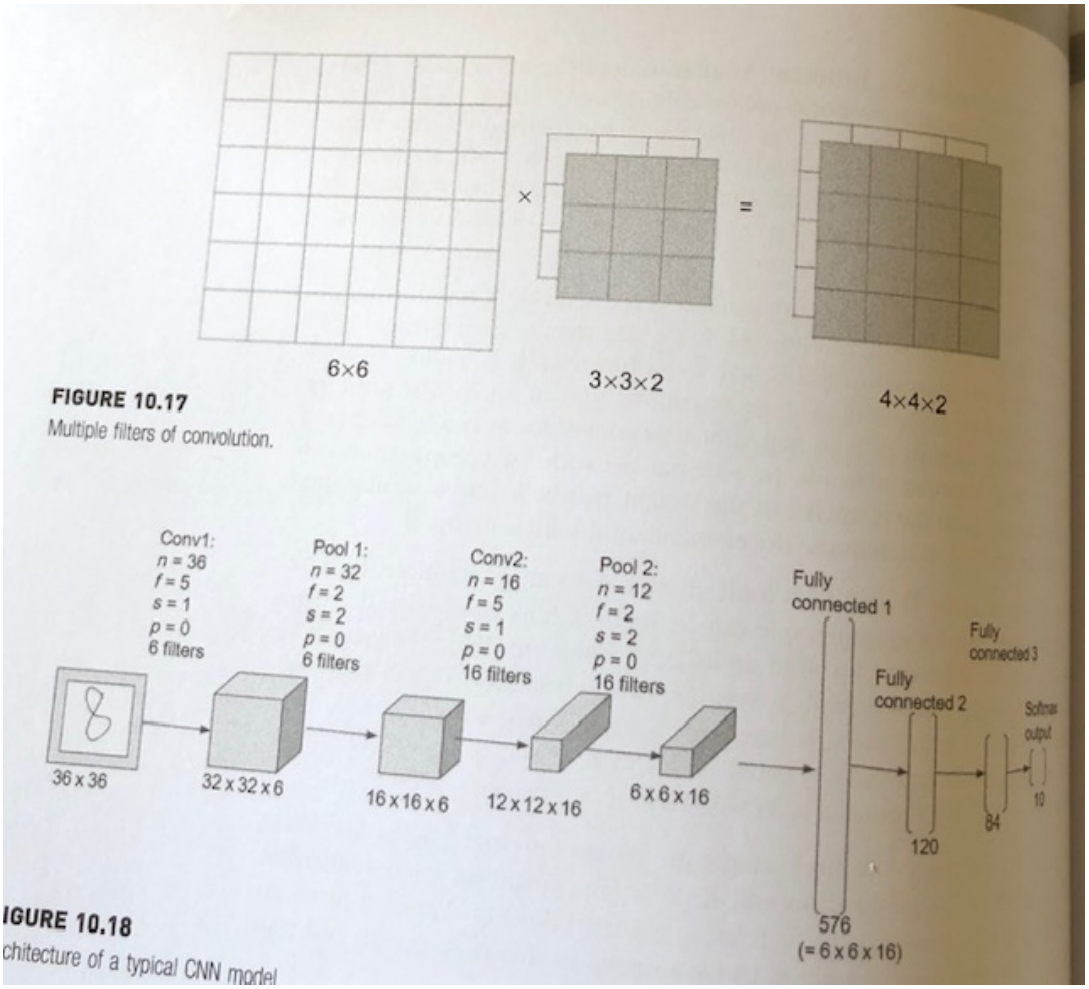


FIGURE 10.17
Multiple filters of convolution.

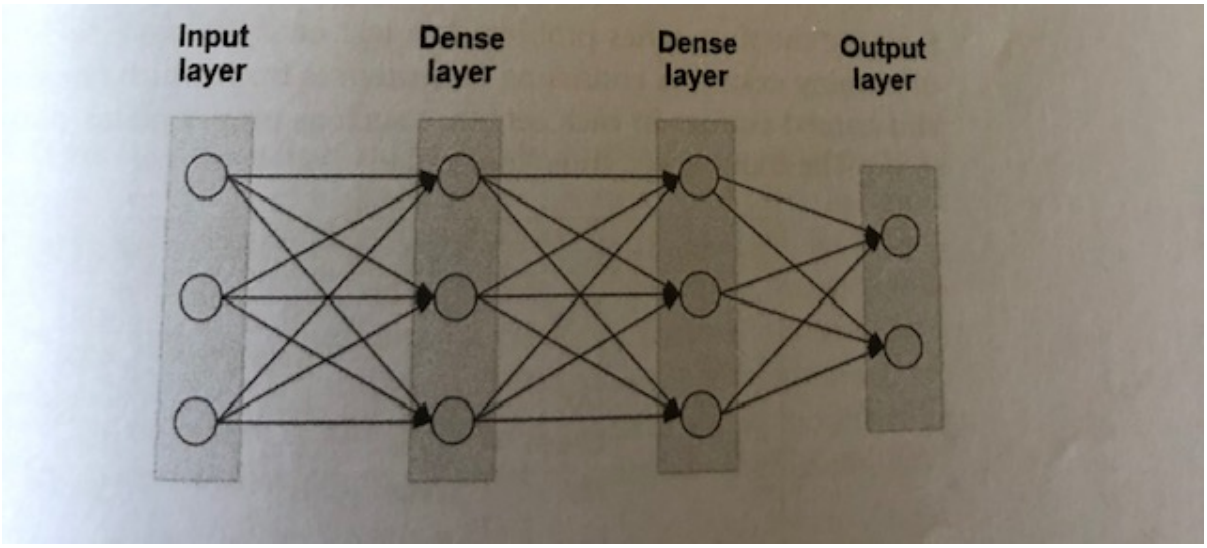
IGURE 10.18
chitecture of a typical CNN model

There are several convolutional and fully connected layers (known as dense layers). Each layer receives the inputs from the previ-

ous layer, applies the weights and aggregates with an activation function.

Dense Layers

A dense or fully connected layer is one where all the nodes in the prior layer are connected to all the nodes in the next layer.



Recurrent Neural Networks

Recurrent Neural Network (RNN) is another example of the use of deep learning methods. This is used when data has a temporal component. Examples are time series from financial data or sensor data, language related data when analyzing a sequence of words that make a sentence or translating words from one language to another. The idea behind an RNN is to train a network by passing the training data through it in a sequence (where each example

is an ordered sequence). In the Figure, for example, $x^{<t>}$ are the inputs where $< t >$ indicates the position in the sequence. There are as many sequences as there are samples. $y^{<t>}$ are the predictions made for each position based on the training data. The training here determines the set of weights of this network, b_x, which is a linear combination with $x_i^{<t>}$ and passed through a non-linear activation producing an activation matrix $a^{<t>}$. Therefore

$$a^{<t>} = g(b_x x^{<t>})$$

RNN also uses the value of the activation from the previous time step (or previous word from the sequence because in most sequences (such as sentences) the prediction of the next word is usually dependent on the previous word or set of words. For example, the previous words "name", "My", "is" would very likely make the next word a proper noun (so $y = 1$). This information is helpful in strengthening the prediction. Therefore, the value of the activation function can be modified by adding the previous steps' activation multiplied by another coefficient, b_a

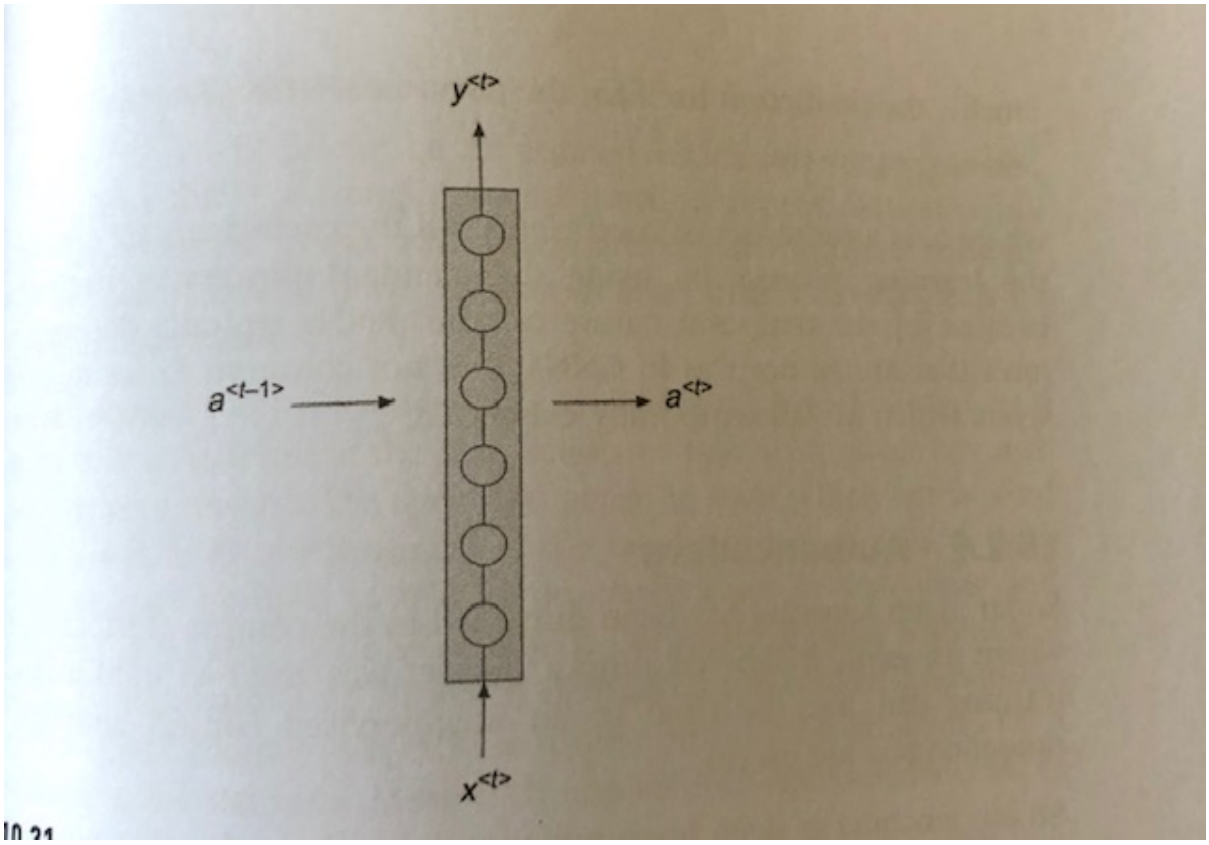
$$a^{<t>} = g(b_a a^{<t-1>} + b_x x^{<t>})$$

The prediction itself for the position $< t >$ is given by

$$y^{<t>} = g(b_y a^{<t>})$$

where b_y is another set of coefficients. All the coefficients are obtained through the learning process using backpropagation. Because of the temporal nature of the data, RNNs typically do not

have structures that are as deep as in CNNs. It is not very common to see RNNs with more than 4-5 layers that are temporally connected.



We covered convolution in previous chapters.