Week 3: Vectors and Linear Transformations:

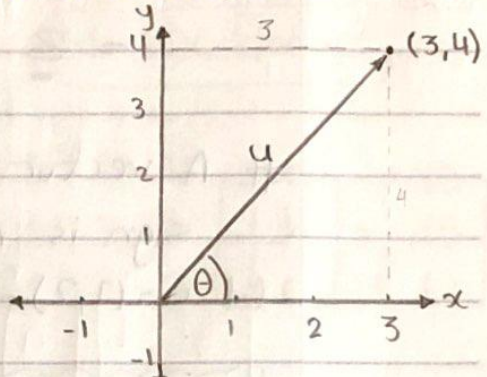
A vector is a simple array or matrix of one column of numbers:

→ It has a magnitude (size) and direction.

→ The number of coordinates of the vector represents the number of dimensions of the vector.

Example:

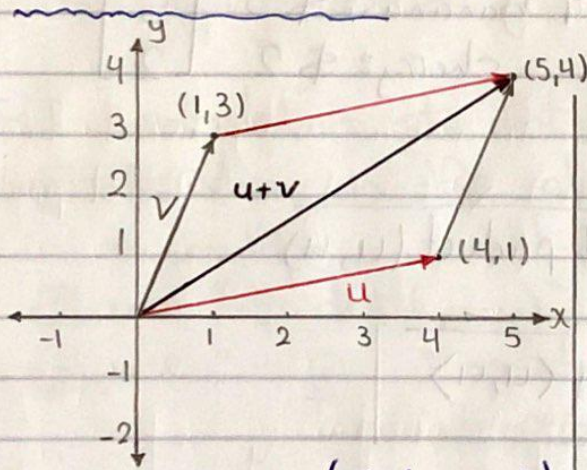
$$\vec{u} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$



Pythagorean Theorem → magnitude (norm) $\|\vec{u}\| = \sqrt{(3)^2+(4)^2}$

Direction → $\tan\theta = \frac{4}{3} \rightsquigarrow \theta = \arctan(4/3) = 0.9273 \text{ rad} = 53.13°$

⊛ Sum of vectors:

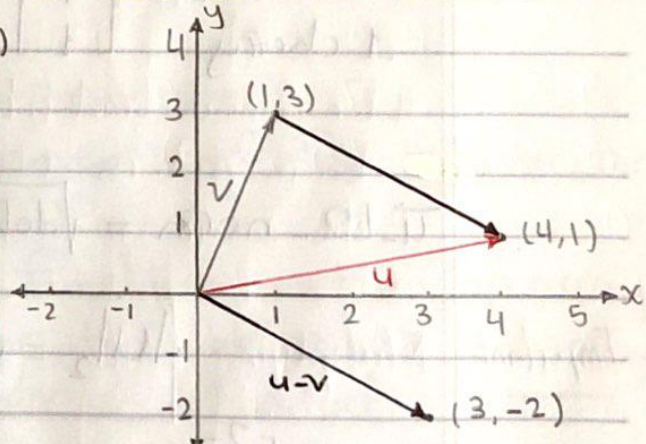


$u+v = (4+1, 1+3) = (5,4)$

start position doesn't matter in vectors

It can be found graphically by completing the parallelogram of the two vectors and get its main diagonal.

⊛ Difference of vectors:



$u-v = (4+1, 1-3) = (3,-2)$

It is the translated vector of conecting the two vector tops, beginning from the negative term to the positive term.

Distance Between Vectors

From the figure of difference between two vectors →

L1-distance	L2-distance (norm)	Cosine distance
$\|u-v\|_1 = \|\Delta x\| + \|\Delta y\|$	$\|u-v\|_2 = \sqrt{(\Delta x)^2 + (\Delta y)^2}$	$\cos(\theta)$
$= \|5\| + \|-3\|$	$= \sqrt{(5)^2 + (y)^2}$	↓ angle between two vectors
$= 8$	$= 5.83$	

A vector can be multiplied by a scalar, if the sign is negative, the direction will be reverted.

let $u = (1,2) \rightarrow \lambda u = (\lambda, 2\lambda)$

Real World Example: Dot Product

Quantities		Prices		Total Price
2 apples	$\begin{bmatrix}2\\4\\1\end{bmatrix}$	apples: \$3	$\begin{bmatrix}3\\5\\2\end{bmatrix}$	$(3*2) +$
4 bananas	·	bananas: \$5		$(4*5) +$
1 cherry		cherry: \$2		$(1*2)$
	vector 1		vector 2	\$28
				← dot product

$\vec{u}$-L2-norm = $\sqrt{\text{dot product}(u,u)}$

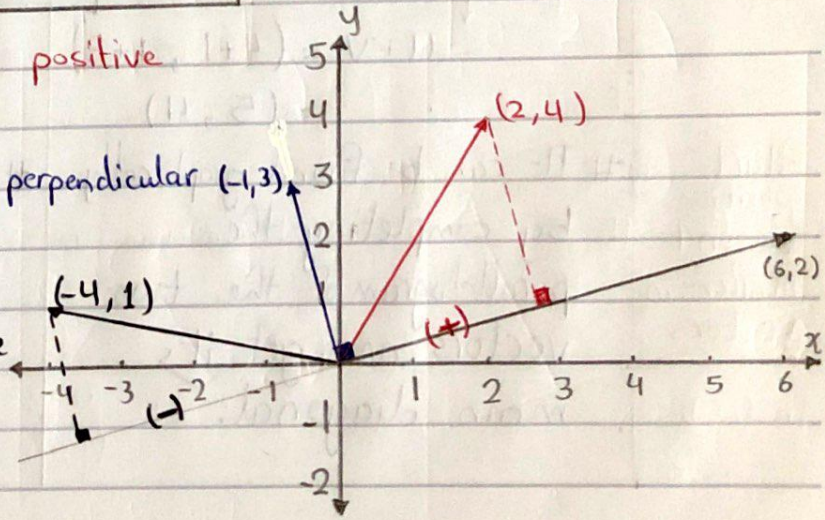
Important

$$\boxed{|u|_2 = \sqrt{\langle u,u \rangle}}$$

$\begin{bmatrix}6 & 2\end{bmatrix} \cdot \begin{bmatrix}2\\4\end{bmatrix} = 20$ positive

$\begin{bmatrix}6 & 2\end{bmatrix} \cdot \begin{bmatrix}-1\\3\end{bmatrix} = 0$ perpendicular

$\begin{bmatrix}6 & 2\end{bmatrix} \cdot \begin{bmatrix}-4\\1\end{bmatrix} = -22$ negative



Equations as Dot Product:

$a+b+c=10$ | $a+2b+c=15$ | $a+b+2c=15$

$$[1\ 1\ 1]\cdot\begin{bmatrix}a\\b\\c\end{bmatrix}=\$10 \qquad [1\ 2\ 1]\cdot\begin{bmatrix}a\\b\\c\end{bmatrix}=\$15 \qquad [1\ 1\ 2]\cdot\begin{bmatrix}a\\b\\c\end{bmatrix}=\$12$$

System of Equations | Martix Product

$$\begin{aligned} a+b+c &= 10\\ a+2b+c &= 15\\ a+b+2c &= 12 \end{aligned}$$

$$\begin{bmatrix}1&1&1\\1&2&1\\1&1&2\end{bmatrix}\cdot\begin{bmatrix}a\\b\\c\end{bmatrix}=\begin{bmatrix}10\\15\\12\end{bmatrix}$$

Dot product operations in Machine Learning (ML) are applied to large vectors with hundreds or thousands of coordinates (called high dimensional vectors). Therefore, speed of calculations is crucial for the training and deplayement of your models.

In calculation of dot product:

very slow — [1] Iteration on the elements of the two vectors to multiply and add to the sum

Fast — [2] use "np.dot(,)" function on two "np.array( )"

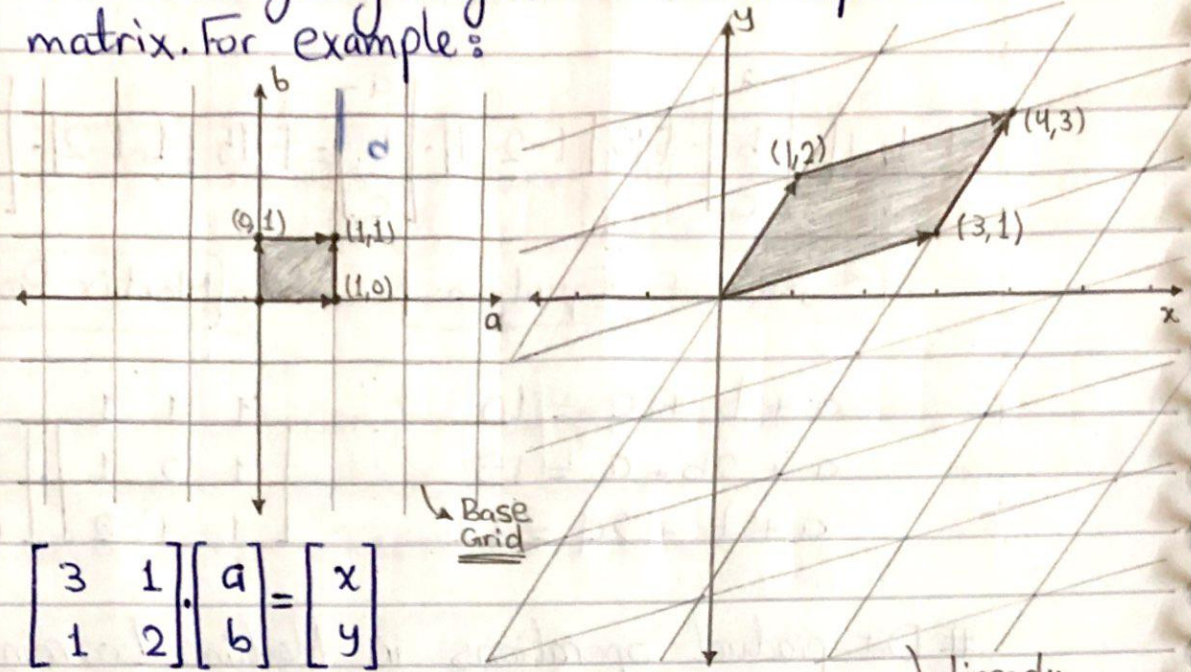
very Fast — [3] use " @ " operator to dot product 2 "np.array( )"

↳ explicit operator for dot product

Matrices as linear transformations:

It's about getting a grid of the multiples of the matrix. For example:

non-singular linear transformation



Base Grid

Linearly Transformed Grid

$$\begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}$$

$(a,b) \rightarrow (0,0) : (0,0) \leftarrow (x,y)$

$(1,0) : (3,1)$

$(0,1) : (1,2)$

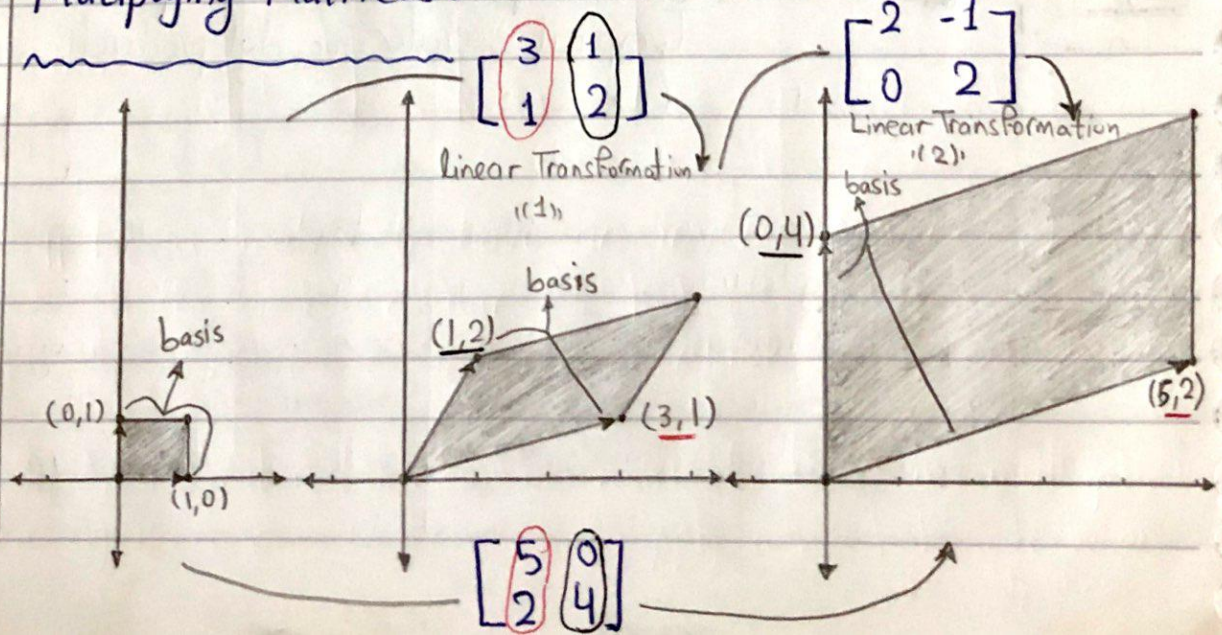
$(1,1) : (4,3)$

For linear transformation we only need the two basis vectors (1,0) & (0,1), and then we complete the linear transformation's parallelogram.

Mutiplying Matrices:

a basis is the basic two-vectors unit in the whole grid

will be discussed in details in week 4



Linear Transformation «1».

$$\begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} (3*1)+(1*0) \\ (1*1)+(2*0) \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} (3*0)+(1*1) \\ (1*0)+(2*1) \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Linear Transformation «2»

$$\begin{bmatrix} 2 & -1 \\ 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 1 \end{bmatrix} = \begin{bmatrix} (2*3)+(-1*1) \\ (0*3)+(2*1) \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} 2 & -1 \\ 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} (2*1)+(-1*2) \\ (0*1)+(2*2) \end{bmatrix} = \begin{bmatrix} 0 \\ 4 \end{bmatrix}$$

Combining the two linear transformations is the dot product of the two matrices, but the second matrix is the first term:

Multiply (2X2) Matrices Form →

second first

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \left[\begin{array}{c|c} [a\ b] \cdot \begin{bmatrix} e \\ g \end{bmatrix} & [a\ b] \cdot \begin{bmatrix} f \\ h \end{bmatrix} \\ \hline [c\ d] \cdot \begin{bmatrix} e \\ g \end{bmatrix} & [c\ d] \cdot \begin{bmatrix} f \\ h \end{bmatrix} \end{array}\right]$$

General Form →

First row by the columns of the other matrix in first row of the result matrix. Second row dot product by the columns of the other matrix in the second row of the result matrix.

very important

We can multiply matrices if only the rows of the term after the dot sign equal to the columns of the matrix term before the dot sign.

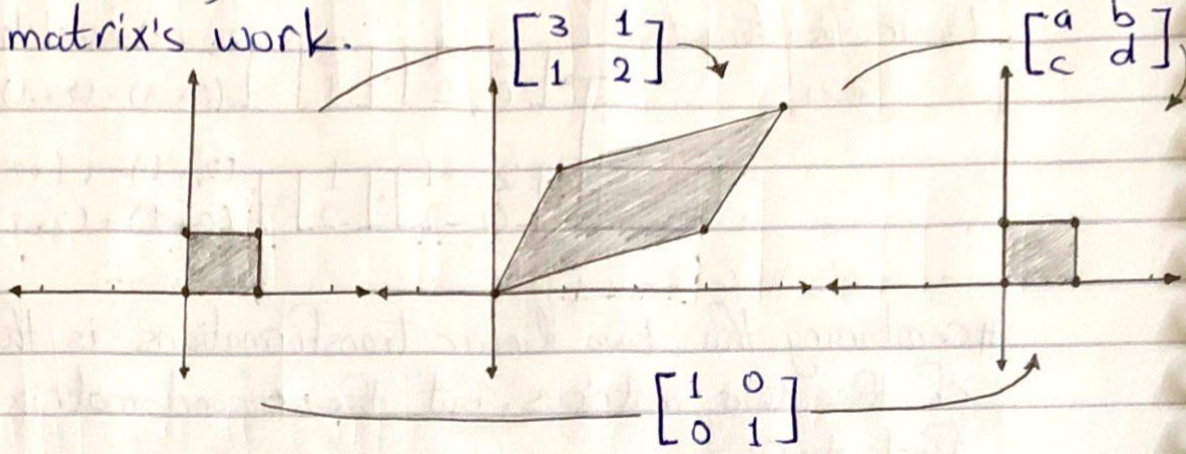
Proof →

$$\begin{bmatrix} 2 & -1 \\ 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} (2*3+-1*1) & (2*1+-1*2) \\ (0*3+2*1) & (0*1+2*2) \end{bmatrix} = \begin{bmatrix} 5 & 0 \\ 2 & 4 \end{bmatrix}$$

Identity Matrix: The matrix that if multiplied by any other matrix, doesn't change it. It has ones in the diagonal and zeros elsewhere.

Inverse Matrix:

It is the matrix that when multiplied by the original matrix gives the identity matrix. In linear transformation, it is the matrix that undust the original matrix's work.



$$\begin{bmatrix}3 & 1\\ 1 & 2\end{bmatrix}\cdot\begin{bmatrix}a & b\\ c & d\end{bmatrix}=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

$[3\ 1]\cdot\begin{bmatrix}a\\ c\end{bmatrix}=[1]$. $3a+1c=1$ $a=\frac{2}{5}$

$[3\ 1]\cdot\begin{bmatrix}b\\ d\end{bmatrix}=[0]$ $3b+1d=0$ $b=-\frac{1}{5}$

$[1\ 2]\cdot\begin{bmatrix}a\\ c\end{bmatrix}=[0]$ $1a+2c=0$ $c=-\frac{1}{5}$

$[1\ 2]\cdot\begin{bmatrix}b\\ d\end{bmatrix}=[1]$ $1b+2d=1$ $d=\frac{3}{5}$

Then: $\begin{bmatrix}3 & 1\\ 1 & 2\end{bmatrix}^{-1}=\begin{bmatrix}2/5 & -1/5\\ -1/5 & 3/5\end{bmatrix}$

very important

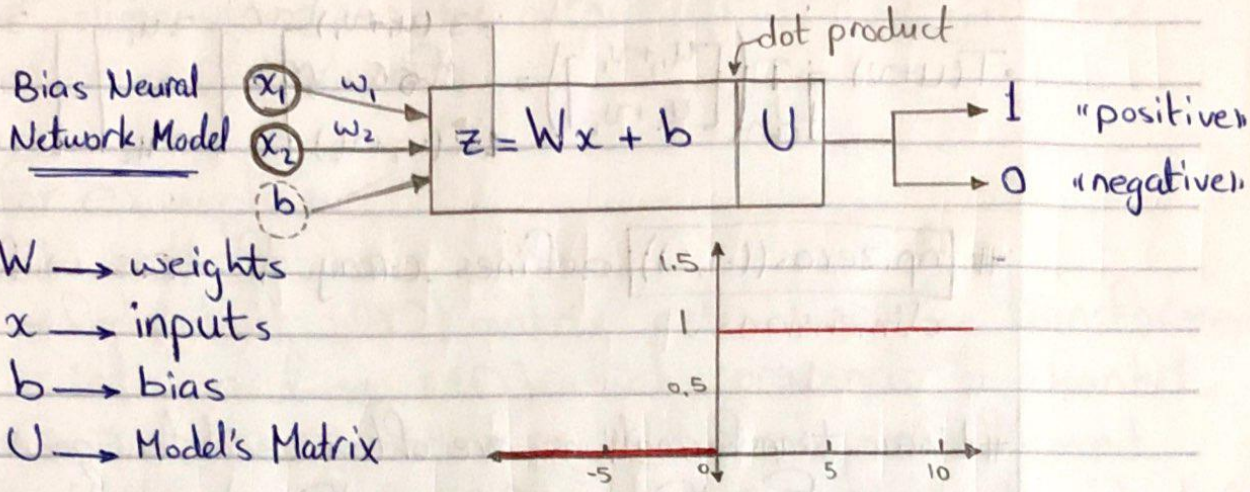
For a matrix to have an inverse (invertible) it has to be non-singular ($\det\neq 0$). If the matrix is singular, it means, it's ~~a singu~~ non-invertible ($\det=0$).

In neural networks, matrices of data are multiplied by the model's matrix, then apply a threshold check on the output.

For more complicated neural networks, the bias is more common than threshold checks.

Checks:

1. Threshold → Points > x : check on number
2. Bias → Points − x > 0 : check on positive



Code Observations:

If we have "numpy arrays" (A, B), we can multiply them using Numpy package function "np.matmul()", or the python operator "@", which gives "numpy.ndarray".

{columns of A has to be equal to rows of B.}

In Python, if we use `np.matmul()` on nxm and nxm matrices, the first matrix will be automatically transposed into mxn.

important # Python uses a technique called Broadcasting. In a matrix, if we used "Matrix − constant", python will broadcast the operation to all the elements within. The same thing happens using `np.dot()`, it broadcast the dot product to all rows; which is multiplying the matrices.

A transformation (T) is said to be linear, if:

→ $T(kv) = kT(v)$

→ $T(u+v) = T(u) + T(v)$

Example:

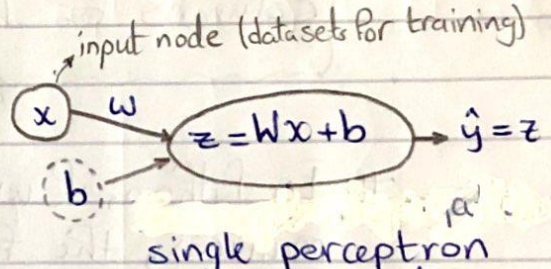
- $T(kv) = T\left(\begin{bmatrix} kv_1 \\ kv_2 \end{bmatrix}\right) = \begin{bmatrix} 3kv_1 \\ 0 \\ -2kv_2 \end{bmatrix} = k\begin{bmatrix} 3v_1 \\ 0 \\ -2v_2 \end{bmatrix} = kT(v)$

- $T(u+v) = T\left(\begin{bmatrix} u_1+v_1 \\ u_2+v_2 \end{bmatrix}\right) = \begin{bmatrix} 3(u_1+v_1) \\ 0 \\ -2(u_2+v_2) \end{bmatrix} = \begin{bmatrix} 3u_1 \\ 0 \\ -2u_2 \end{bmatrix} + \begin{bmatrix} 3v_1 \\ 0 \\ -2v_2 \end{bmatrix} = T(u)+T(v)$

`np.zeros(( , ))` defines array of zeros with needed dimensions.

Linear transformations are often used to generate complex shapes from the basic ones, through scaling, reflection, rotation, shearing, ... etc. The software responsible for rendering of graphics, has to process the coordinates of millions of vertices, then use matrix multiplications to manipulate coordinates and merge multiple transformations together.

The simplest neural network has only one perceptron.



Weight (w) and bias (b) are the parameters which will get updated when the model gets trained.

set of input nodes are called input layer. There is an output layer consists of nodes for the stored calculated data.

Very Important

Forward propagation → gets the perceptron's output calculated.
Backward propagation → get the required corrections for parameters.