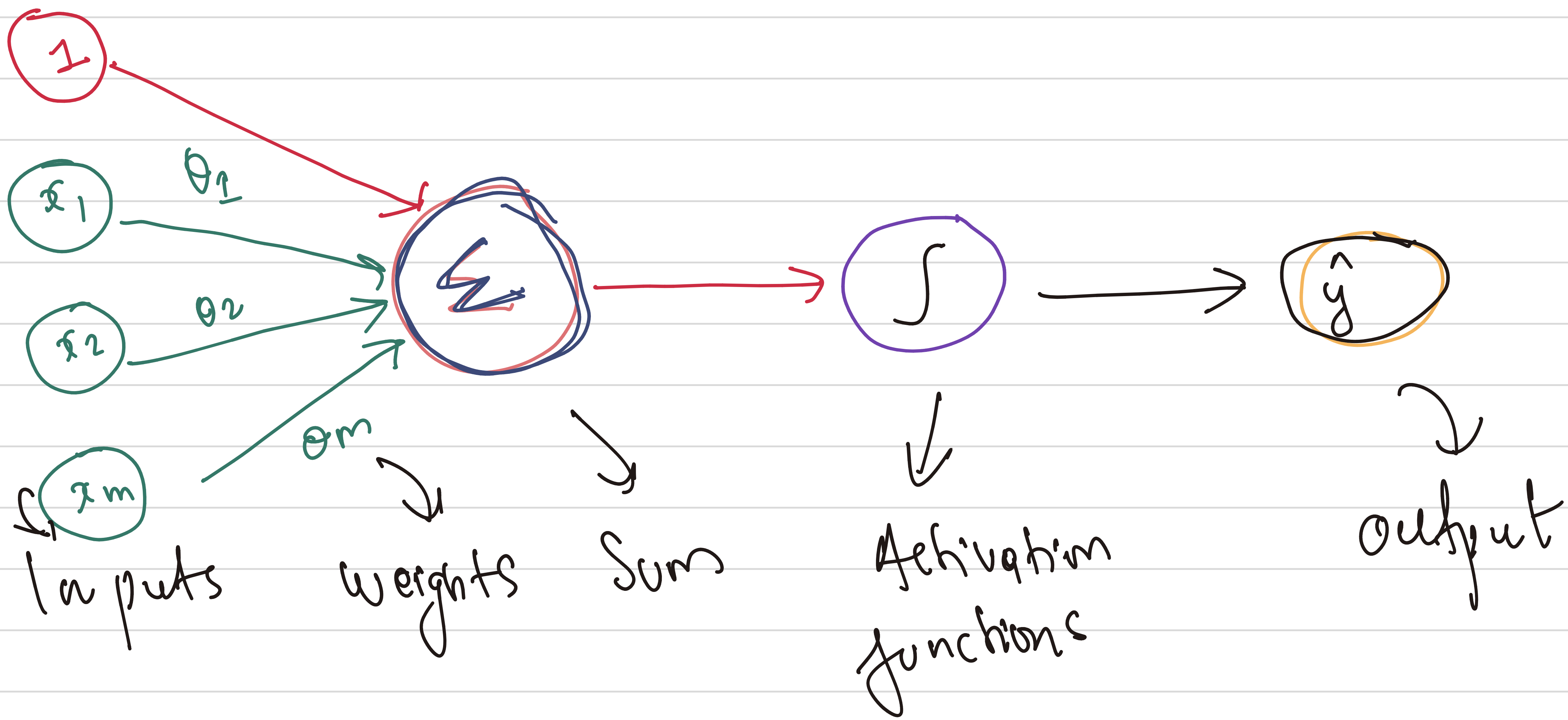
Day 7, Oct-9, 2024.

Dot products for Neural Networks

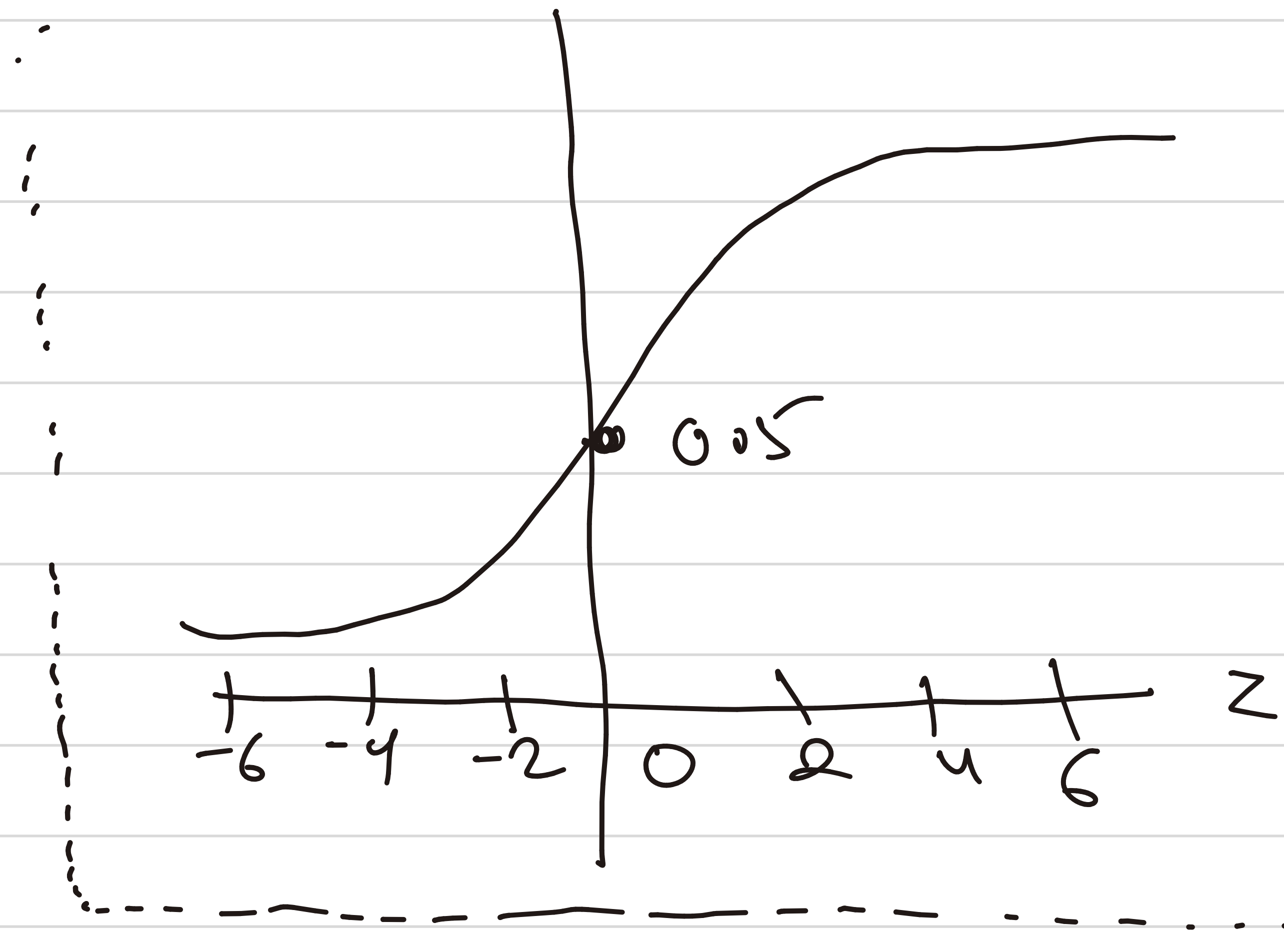


Activation functions

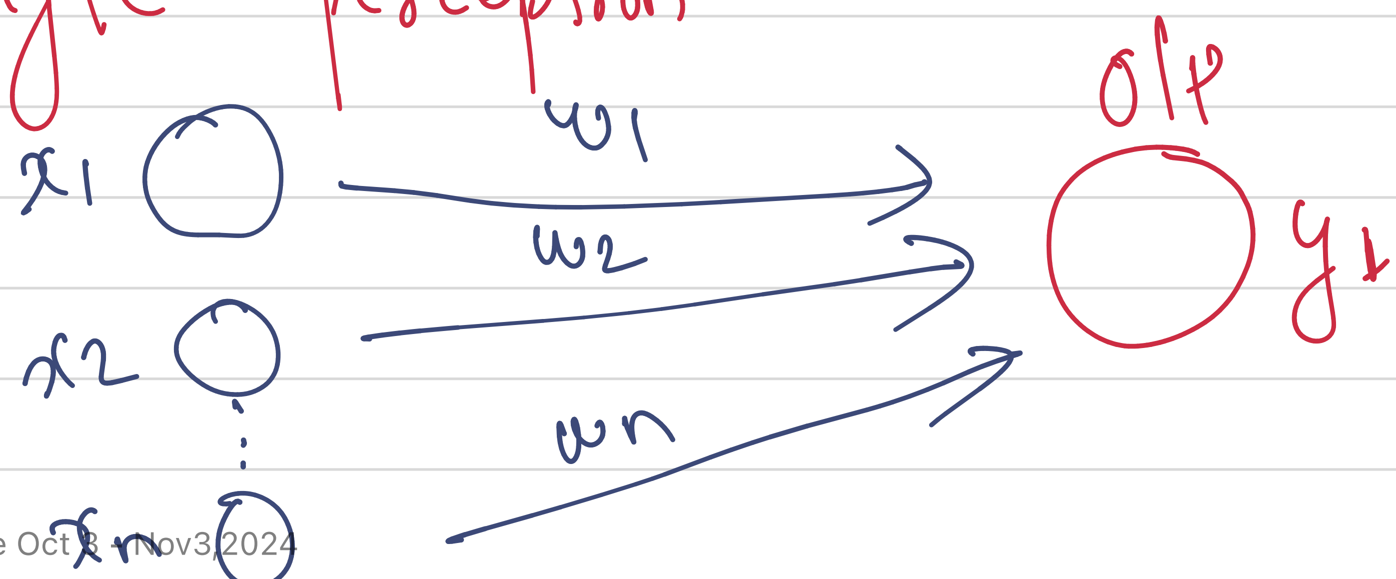
$$\hat{y} = g(\theta_0 + X^T\theta)$$

- Example: Sigmoid function

$$g(z) = \sigma(z) \Rightarrow \frac{1}{1+e^{-x}}$$



Single Perceptron



$$y_1 = \sum_{j=1}^{n} w_j x_j$$

$$\Rightarrow (w_1 \cdots\cdots w_n) \begin{pmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{pmatrix}$$

$$\Rightarrow Wx$$

→ Inputs are Col Vector each multiplied and add together.

→ using dot product

$y = w_1 x_1 + w_2 x_2 + \cdots + w_n$

Then apply Activation function then the o/p is $\hat{y}$ ...

if vector input $\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ are close or parallel to eachother

then for a given input magnitude, the response is maximized when the input is parallel to the weight vector

weight vector can be features, 'x' is picture and 'w' are the eyes inside the picture if we give model to find eyes / detect eyes for a given chair Image then the model activation functions are less likely to get fired because of

no features matched or neurons weights are so low it doesn't get fired or activated.

Matrix Multiplication

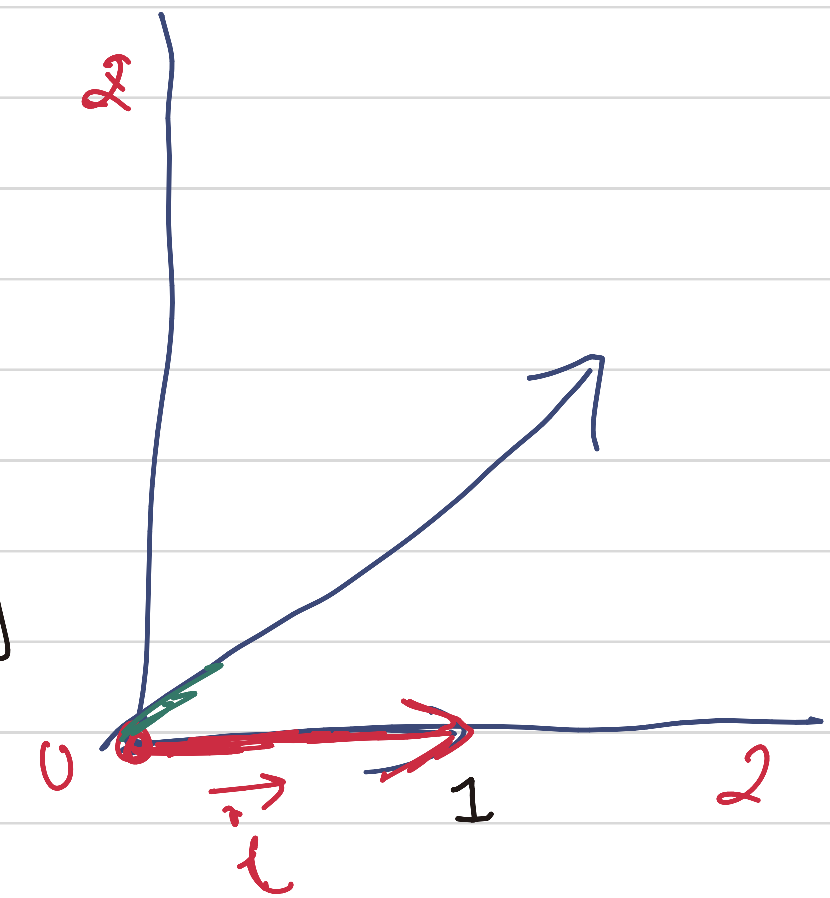
$$\begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 7 \\ 6 \end{bmatrix}$$

$$3 \begin{bmatrix} 2 \\ 1 \end{bmatrix} + 1 \begin{bmatrix} 1 \\ 3 \end{bmatrix} \Rightarrow \begin{bmatrix} 7 \\ 6 \end{bmatrix}$$

dot product.

Matrix Multiplication is Scaling for an example

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



$\vec{i}$ vector ends up at $\begin{bmatrix} 2 \\ 1 \end{bmatrix}$

$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$ would be the vector transformed by $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and if

$$\begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 3 \end{bmatrix} = \begin{bmatrix} 7 \\ 6 \end{bmatrix}$$

Here vector $\begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}$ is transformed by $\begin{bmatrix} 1 \\ 3 \end{bmatrix}$ and o/p is $\begin{bmatrix} 7 \\ 6 \end{bmatrix}$

The idea is that where $\vec{i}$ and $\vec{j}$ is transformed to using $\begin{bmatrix} x \\ y \end{bmatrix}$ to multiply them.

$$\begin{array}{cc} \vec{i} & \vec{j} \\ \end{array}$$
$$\begin{bmatrix} 2 & 3 \\ 1 & 4 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow T \rightarrow T \text{ vector (transformed vector)}$$

$$\begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 7 \\ 6 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \cdot \left(3 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) \Rightarrow 3 \begin{bmatrix} 2 \\ 1 \end{bmatrix} + 1 \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

Complexity Arises with increasing 3D Vector Dot product.

where $\vec{i}$ ends $\begin{bmatrix} 1 & 3 \\ -2 & 0 \end{bmatrix}$ → where $\vec{j}$ ends up

$j \Rightarrow \begin{bmatrix} 3 \\ 0 \end{bmatrix}$ $i \Rightarrow \begin{bmatrix} 1 \\ -2 \end{bmatrix}$ using $\begin{bmatrix} x \\ y \end{bmatrix}$ we get Transformed vector

Matrices developed to solve linear Equations

$$A\vec{x} = \vec{V}$$

$2x + 5y + 3z = -3$

$4x + 0y + 8z = 0$

$1x + 3y + 0z = 2$

Extracts the coefficients of x, y & z.

$$\overset{A}{\begin{bmatrix} 2 & 5 & 3 \\ 4 & 0 & 8 \\ 1 & 3 & 0 \end{bmatrix}} \overset{\vec{x}}{\begin{bmatrix} x \\ y \\ z \end{bmatrix}} \Rightarrow \overset{\vec{v}}{\begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix}}$$

Now we can find $\vec{x}$ such that when transformed by matrix A it lands on $\vec{v}$

We use matrices as linear equations it gives unique solutions, Consistency, efficient Computing, easy to solve, gives solution or not.

Eigenvectors & Eigenvalues

Eigen is German for "proper", 'Special', 'Characteristics'

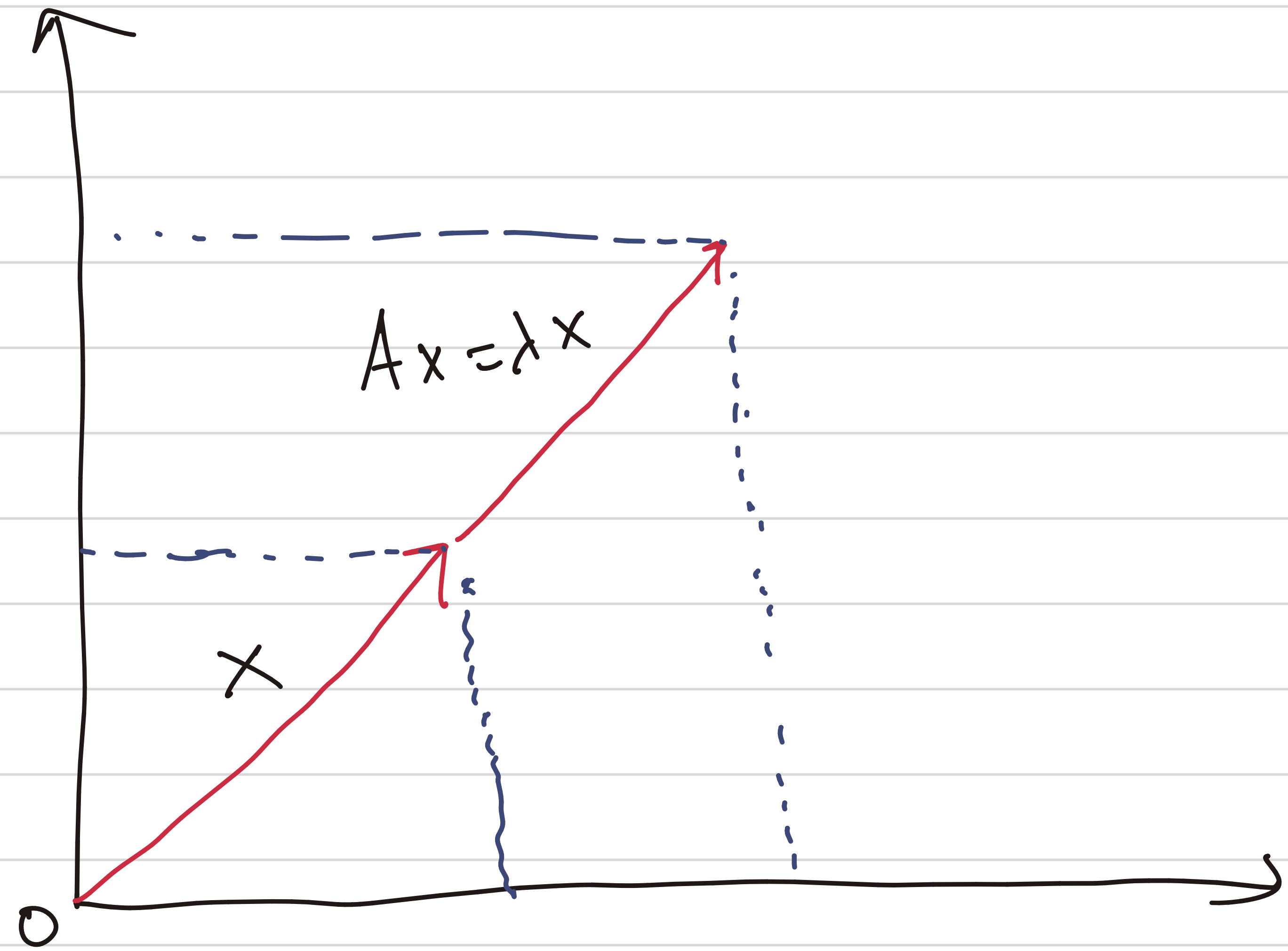
The eigenvector x of matrix A is a vector that satisfies the equation

$$Ax = \lambda x$$

where λ, called eigenvalue, is a number

Graphically, this means that under operation A, the vector x doesn't change direction, just magnitude.

So vector 'x' doesnot change direction, just magnitude.



So, $Ax = \lambda x$

Matrices transform the vector,

Some matrices when transform the vector then the resulting vector is only scaled. And the scaled vector 'x' is eigen vectors.

Eigen vectors and Eigen vectors

As we Know that Matrices transform (rotate, scale, shear) and also direction of the vector gets changed when the matrices is transformed.

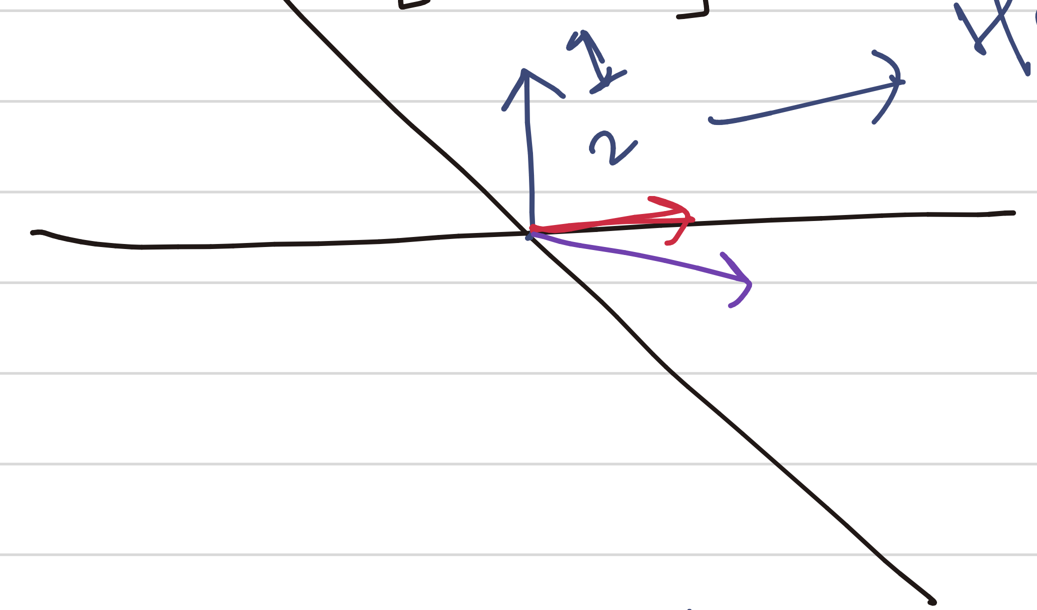
But what if only vector is Scaled without change in direction. Then the transformed vector (x) is eigen vector and the amount by which vector Scaled is eigen value.

So the eqn

$A x = \lambda x$ must be Satisfied to be

(x eigen vector and λ = eigen value

$$\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix}$$

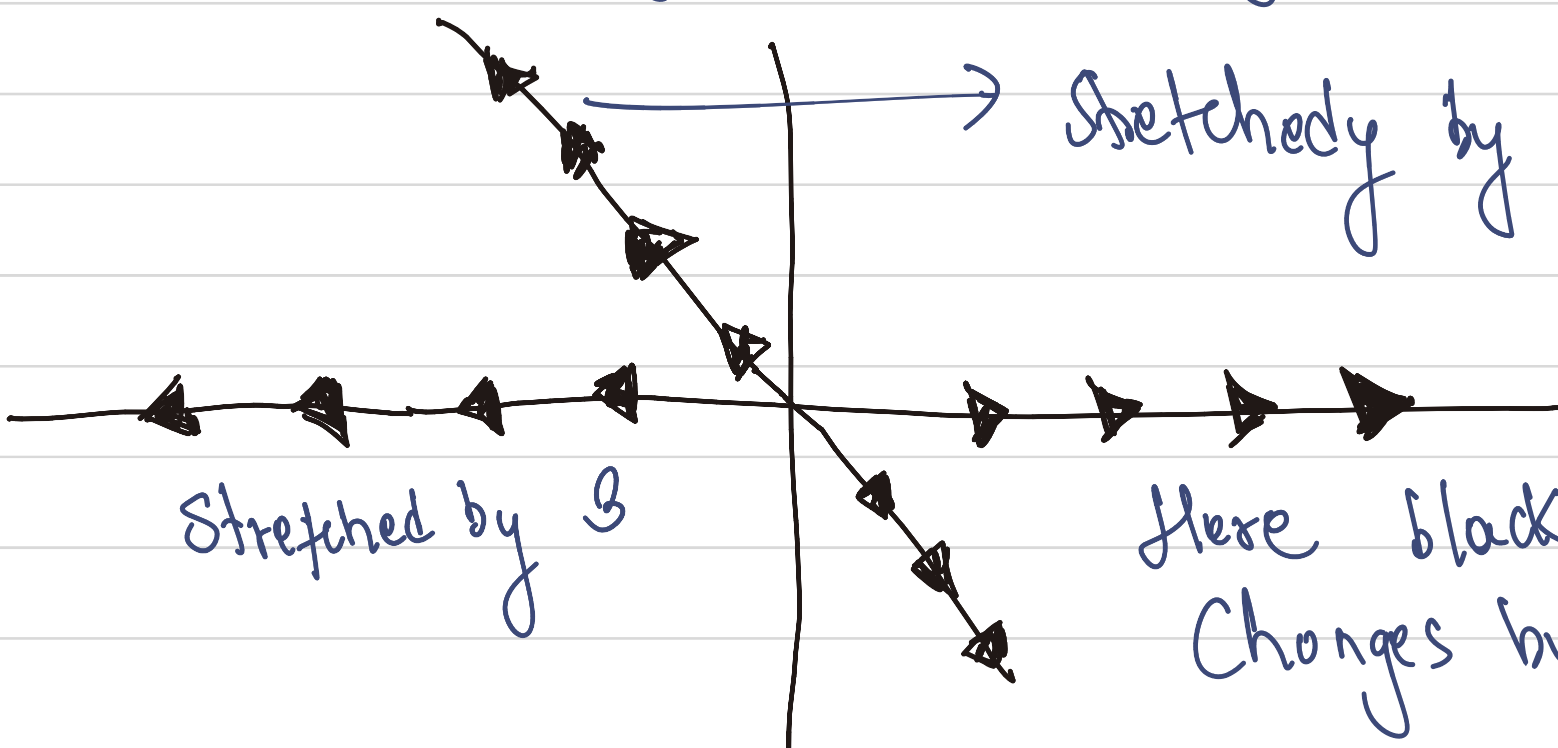


Here $\begin{bmatrix} 3 \\ 0 \end{bmatrix}$ doesnot change ∡, remains in Span

Here $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$ Magnitude & ∡ both changes.

⇒ Eigen vector: vector that only Scales

Eigenvalue: The amount by which the eigenvector Scales (stretched)



Stretched by 2

Stretched by 3

Here black vector magnitude Changes but not ∡

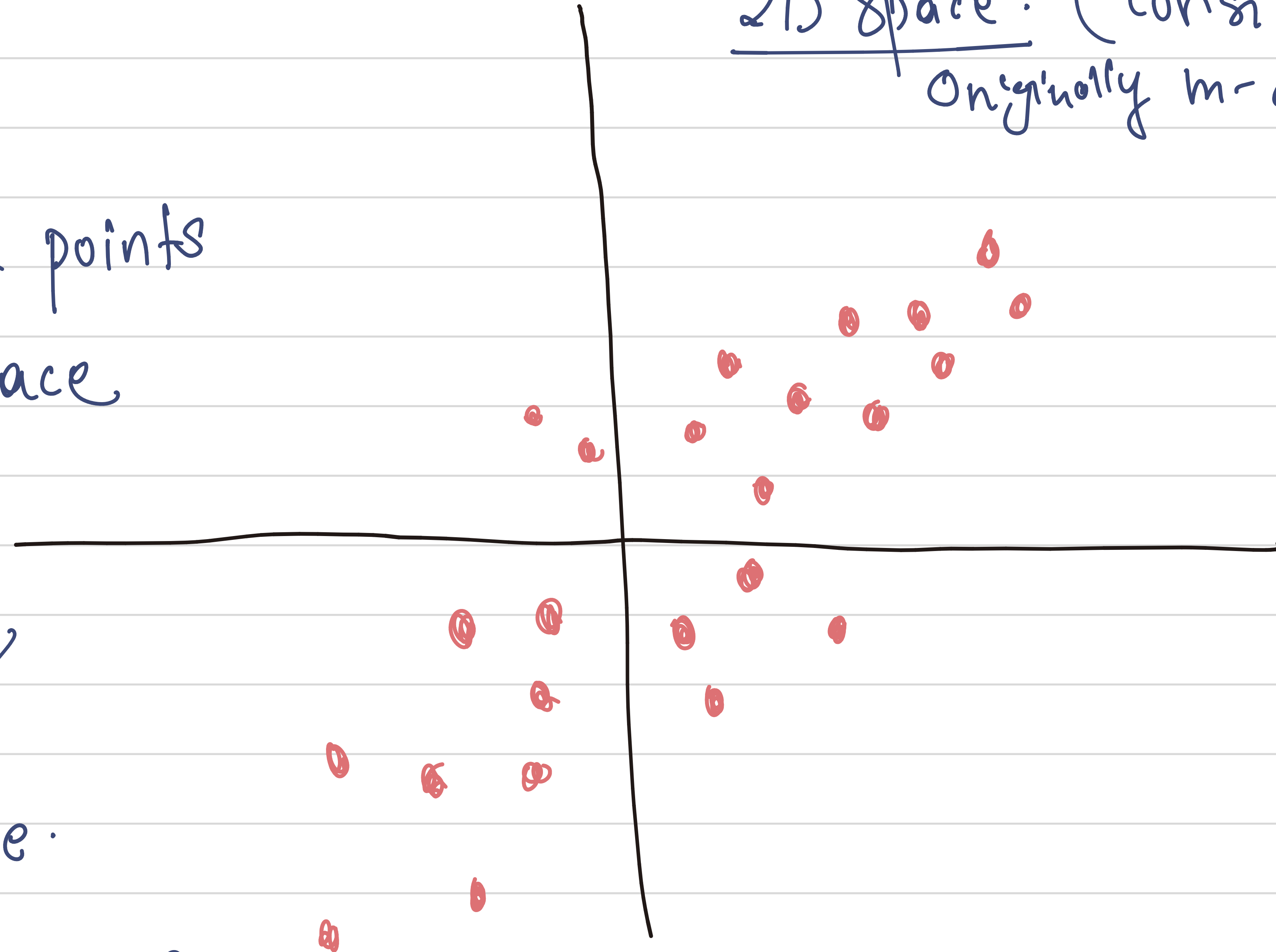
Why Eigen?

Suppose we have n data points in m-dimensional space

oftentimes, these points will be clustered along a line or another low-dimensional subspace.

'2D Space' (Consider)

Originally m-dimensional.



what is that line or subspace?

finding slope 'm'

$y = mx + c$

we can find the straight line.

(Clustering) (Intuitive)

finding the best-fit straight line that contains almost all shown points in a figure.

We can compress the data, suppose in the figure if n=20 (x,y) we need 40 numbers to represent all states of (x,y)-cordinates. 40 numbers can be compressed into two groups without losing the data.

We can squeez the large dimensional data into smaller space without losing the data information. So, the lower-dimensional subspace is required and possibly possible using Eigen vector / Eigen value.

Why Eigen? • Represent the data points as a matrix

To find the line, first organize the data points into an $m \times n$ matrix A, and then compute the eigenvectors of $AA^T / n-1$.

$$A = \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccc} \text{Student 1} & \text{Student 2} & \text{Student 3} & \text{Student 4} & \text{Student 5} & \text{Student 6} \end{array}$$

$$A = \left[\begin{array}{cccccc} 3 & -4 & 7 & -1 & -4 & -3 \\ 7 & -6 & 8 & -1 & -1 & -7 \end{array}\right] \begin{array}{l} \text{Math} \\ \text{History} \end{array}$$

There's little more to this, we first centered the data points by subtracting the mean of each row. The matrix $AA^T / n-1$ is called a Sample Covariance Matrix.

Compression keeps the Semantics losing the redundancy in the data.

$A \times A^T$ Converts non-Square matrix into Square Matrix

when $m \times n$ matrix when multiplied by its $n \times m$ (A^T).

Above $A = [\quad]$ is co-variance, take this A vector and we find the eigenvector (for A is not any linear transformation) (n-1 is for centering purpose) then

$A \cdot A^T$ are Covariance vector represents how these data represents in m-dimensional space. Represent all the information of data in m-dimensional space.

Eigen Decomposition:

It means decomposing the matrix into it's eigenvectors and finding their Scale vectors.

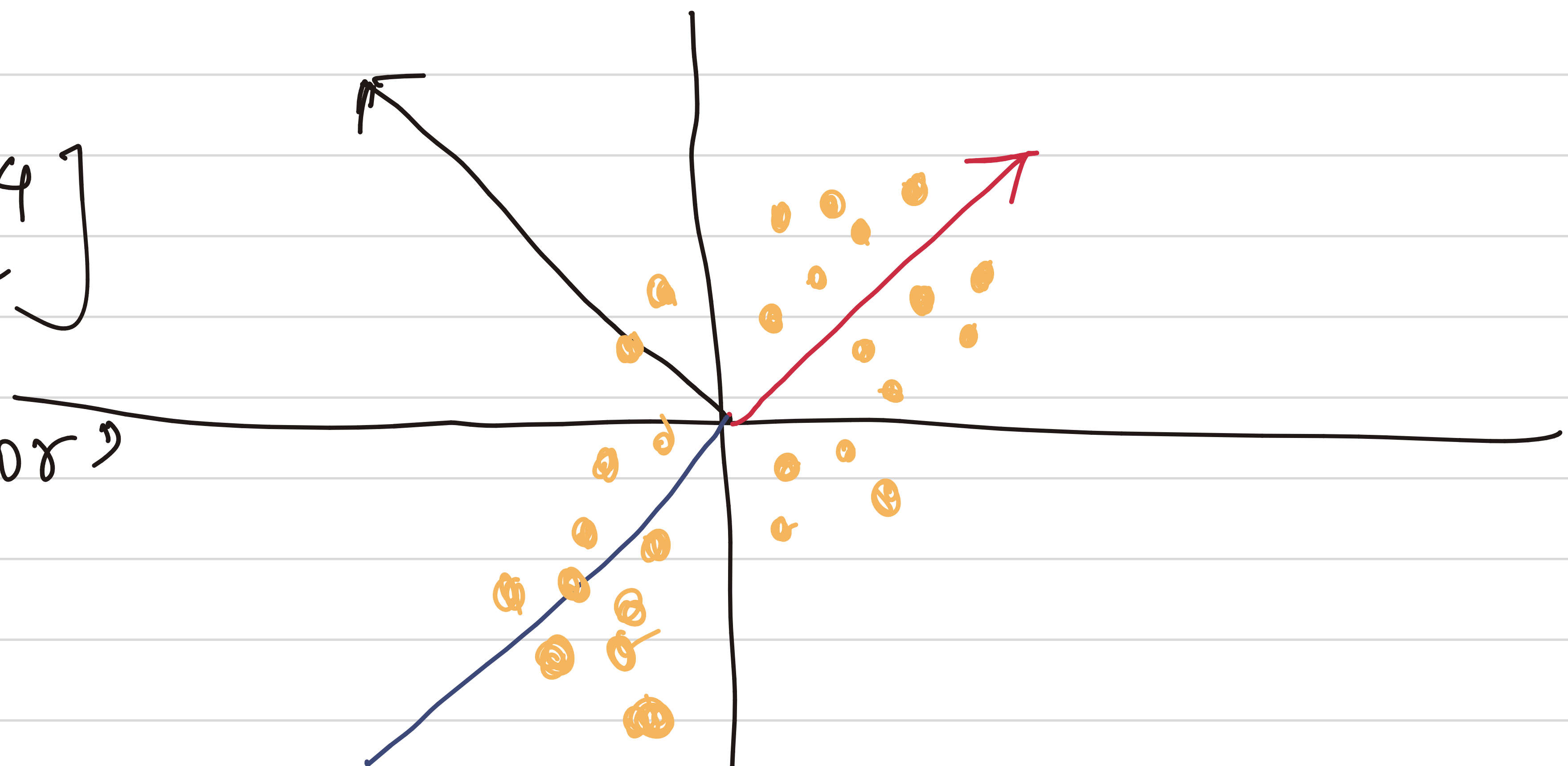
Eigenvectors represent the Principal direction of data which basically means that the line best corresponds to how the data's can be clustered together and we can view.

Eigen Vector → Principal direction of data

In this example, the matrix $AA^T/5$ has two eigenvectors (eigen vectors may have imaginary & real numbers as eigen value).

$e_2 = \begin{bmatrix} -1.4 \\ 1 \end{bmatrix}$

"Scale factor" is 3



$e_1 = \begin{bmatrix} .6 \\ .8 \end{bmatrix}$

"Scale factor" is 57

One with higher Scaling factor gives the PDA (clustering of data)

Above figure can be for Maths & History, Customers & Brand, Recommendation for users and many.

Eigen vector & value represents the matrices itself.

Eigen Decomposition gives logarithm of Matrices (which is not easy to compute of course).

What we studied ?

1. Eigen vectors & values
2. Matrix transformation (Intuition)
3. Dot product as Neural Networks

4. Eigen Decomposition
5. Why Eigen?
6. Matrices As System of Linear Equation Solving
7. PDA (Principal Direction of Data)

External Resources

- Gilbert Strang's MIT Courseware
- Grant Sanderson 3Blue1Brown

8. Concept of Compression and usage of Eigen Vectors & Value.

Summory

① Dot product can be used in Neural networks. for a Single Percoptor

$$y = wx + b$$ where b is bias

w = weight

$x \Rightarrow$ input

$e \Rightarrow$ eigen value constant

* Activation function $\sigma(z) \Rightarrow \dfrac{1}{1+e^{-z}}$

$$z = y = wx + b.$$

② Matrices can be used to solve the System of linear Equations and when we have two $\vec{i}$ and $\vec{j}$ and Scaling them means by what factor $\begin{bmatrix} x \\ y \end{bmatrix}$ if is transformed.

③ Eigen Vector and Eigen value for any vector transformation,

$A\vec{x} = \vec{v}$ (Normal Vector Linear Transformation)
$A \Rightarrow$ vector

When,

$A\vec{x} = \lambda\vec{x}$ where $\vec{x} \Rightarrow$ vector (Eigen vector)
$\lambda \Rightarrow$ Eigen value

Eigen value or Vector does only scaling doesnot change the direction. Changes only magnitude.

④ Eigen Decomposition: Decomposing Matrix into Eigen Vectors and finding their Scalar vector.

⑤ PDA allows to locate the clustering of data points and Eigen Vectors & values allow to compute logarithms. Eigen values can be real or imaginary numbers.

Foundations of AI

Since, AI is interdisciplinary field and the creation of intelligent machine requires knowledge, ideas, research from various sources. So, what are the foundations of AI?

1. Linguistic → Language generation, understanding, communication
2. Economics → game theory, rational utility, usage & cost
3. Computer Engineering → Artificats & Robotics, Automation + CE
4. Mathematics → Logics, propositional, Knowledge Representation
5. Philosophy → perceive, questioning, Logical argument, reasoning
6. Pyschology → mind and act like Human behaviour, thinking, acting
7. Neuro Science → Studies of neurons, brain to mimic brain in mod
8. Cognitive Science → Studies broad field including Cognitive functionalit
9. Quantum Computing (Emerging) → Quibits Computing, Promising Pat
10. AI ethics & Society → Structuring, Morale, Ethics, Society, Privacy, Country & States

History of AI.

① 1940s - 1950s: Early foundations

1943: Mc Colloch & Pitts developed first mathematical model of neural network

1950: Alan Turing Proposed Turing Test

1956: AI term coined during Dartmouth Conference, birth of AI as a field of study

② 1960s: Exploration of Symbolic AI

① 1961: first industrial robot installed

2 1965: ELIZA, a chatbot developed

3 1966: First Machine Translation such as SYSTRAN developed focusing on Russian-English Translations (Cold-war Era).

1970s: AI Winter & Challenge

1972: PROLOG, a logic programming for AI

1973: First AI Winter began due to disappointing result from Early AI projects (Reduced funding)

1979: Standford Cart, Autonomous vehicle

1980s: Expert Systems and Revival

1982: John Hopfield revived ANN introduced Hopfield network

1986: Geoffrey introduced backpropagation, improving ANN

1990s: Machine Learning & data:

1997 : IBM's blue Chip defeat world chess champion

1999: AI in e-commerce, recommender.

2000s: Data Explosion and Neural Network

2006: Geoffrey Hinton popularized the term

Deep Learning (Multi-layer ANN)

2008: AI powered devices, Apple's Siri better at NLP

2010's: AI Revolution & Deep Learning

2012: AlexNet, CNN won ImageNet Competition, ground-breaking Computer Vision

2014: Google's AlphaGo defeated professional Go player

2016: OpenAI was founded

2017: Transformers in NLP revolutioned, Paper is "Attention is All you Need", LLM's developed

- # 2020's. (AI Hype and Open AI Revolutionatized)

2021: GPT-3, first LLM's with 175-billion parameters best LLM's till the date

2022: DALLE-2 and Stable Diffusion as Generative AI

2023-2024: GPT-4, Ongoing work in AGI by OpenAI and DeepMind's Projects.