**Deep Learning Specialization**

**+bài học :**

**-mạng neural và deep learning**

**🡺ví dụ nhận dạng con mèo**

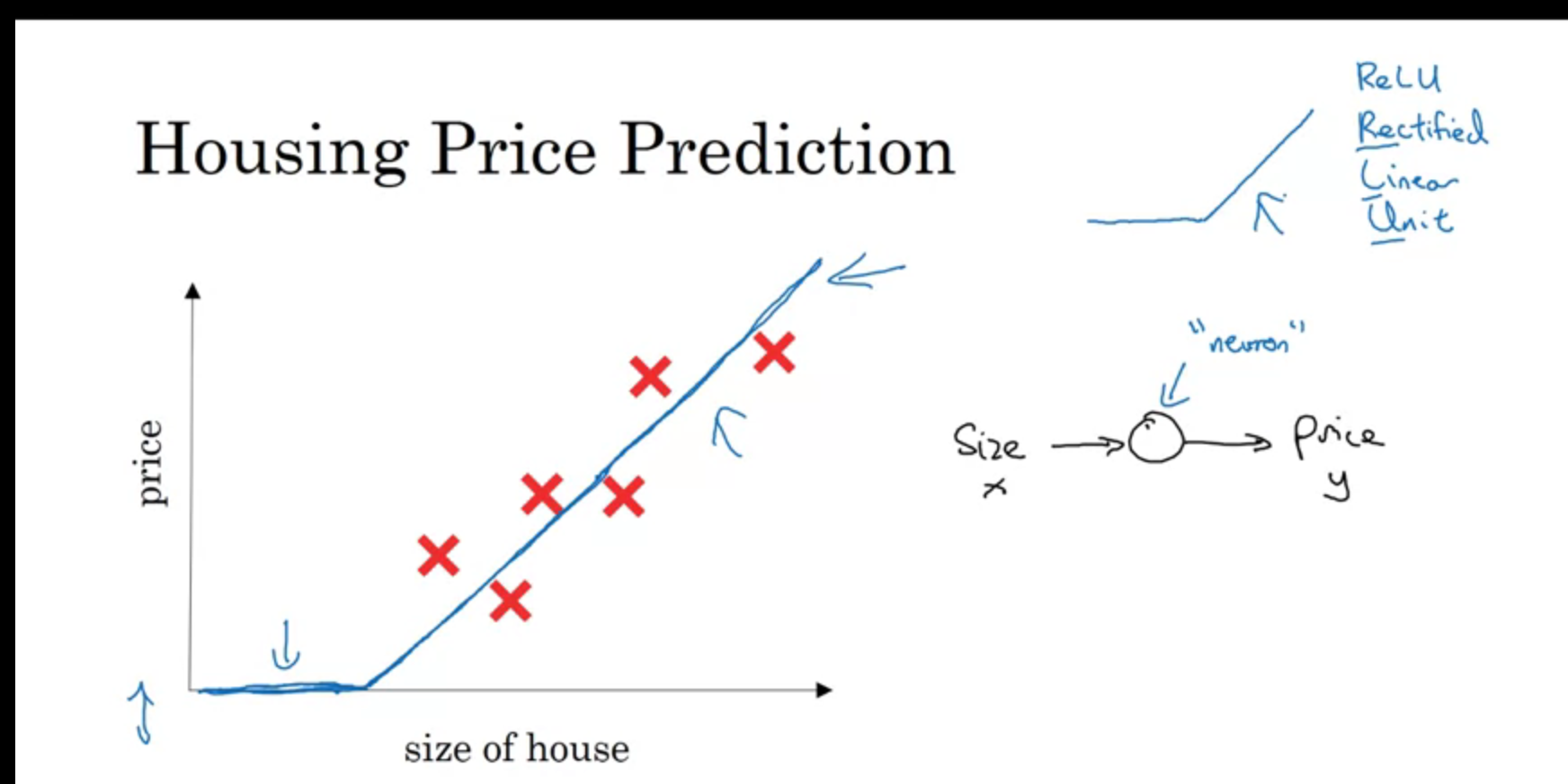
**-Cải thiện deep neural networks :hyperparameter tuning,regularization and optimazation**

**- cấu trúc 1 dự án machine learning –cách lên kế hoạch:phát triển :đào tạo,(sẽ có 1 số bài bản quyền)**

**-mạng convolutional neural (CNN)-dùng cho ảnh**

**-natural language processing :building sequence models –xử lí ngôn ngữ tự nhiên (mạng thần kinh viết tắt là RNs-các model LSTM-viết tắt model bộ nhớ ngắn hạn và dài hạn**

**1.** **What is a Neural Network?**

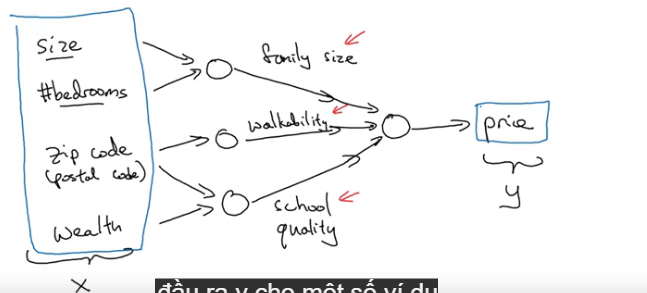
****

**Giải sử ta có 6 ngôi nhà X đỏ**

**Một đường thẳng biểu thị cho giá nhà ,hàm số gọi là ReLu (rectified linear unit ) với tối đa là 0 - (bằng 0 vì giá nhà không có giá trị âm**

**-Hàm này nhận x (size ) => qua neural tính toán => y (price)**

**-size được coi như là 1 nerual nhỏ ,lẻ**

****

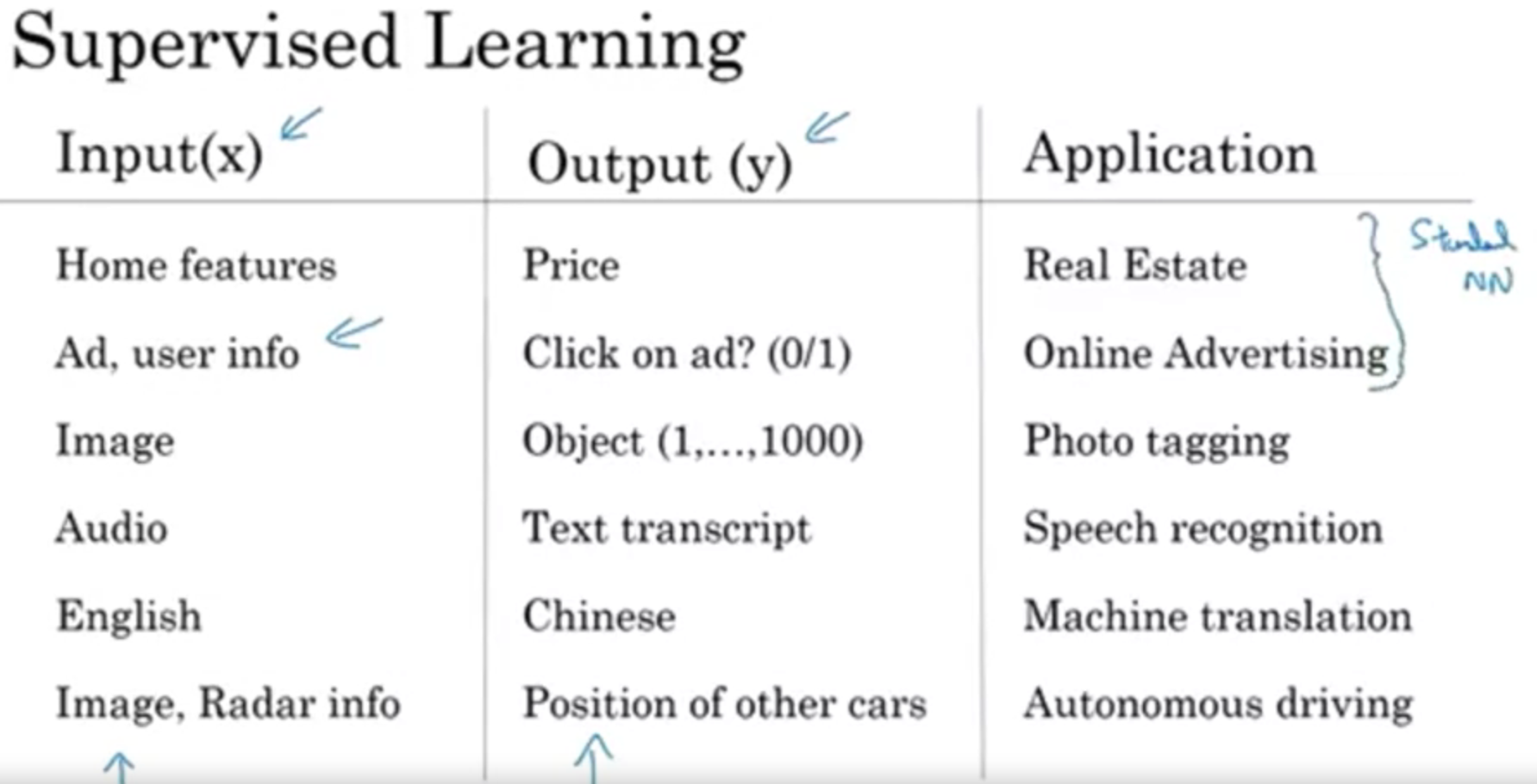
+neural lớn sẽ là sự kết hợp của nhiều neural nhỏ :phòng ngủ, mã code khu vực ,trường học ,thời tiết….

+mỗi hình tròn là 1 nơ ron nhân tạo ,mỗi hình đều link với các đầu vào x

+mỗi hình tròn là 1 ẩn

* Từ đó dự đoán giá nhà

Một số ứng dụng của mạng nơ ron áp dụng trong học sâu và học có giám sát

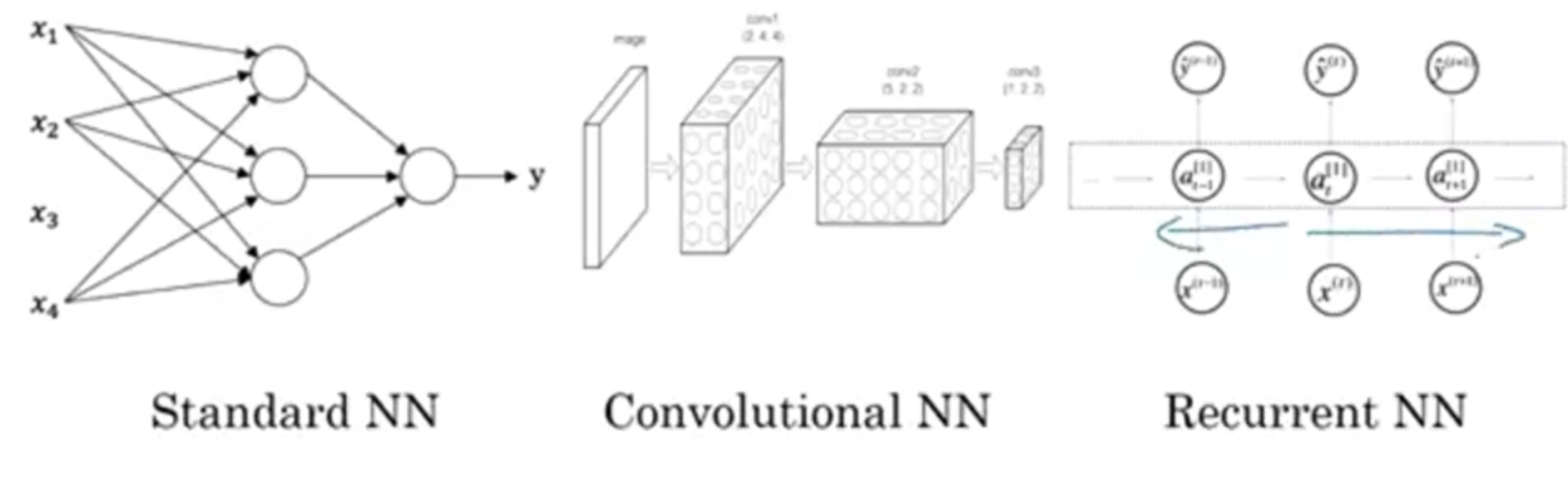


+đối với nhà ta dùng mạng nơ ron tiêu chuẩn

+Đối với ảnh ta dùng CNN tích chập

+giọng nói và ngôn ngữ ta dùng RNN hồi quy

+lái xe tự động ta dùng nơ ron custome hoặc hybrid

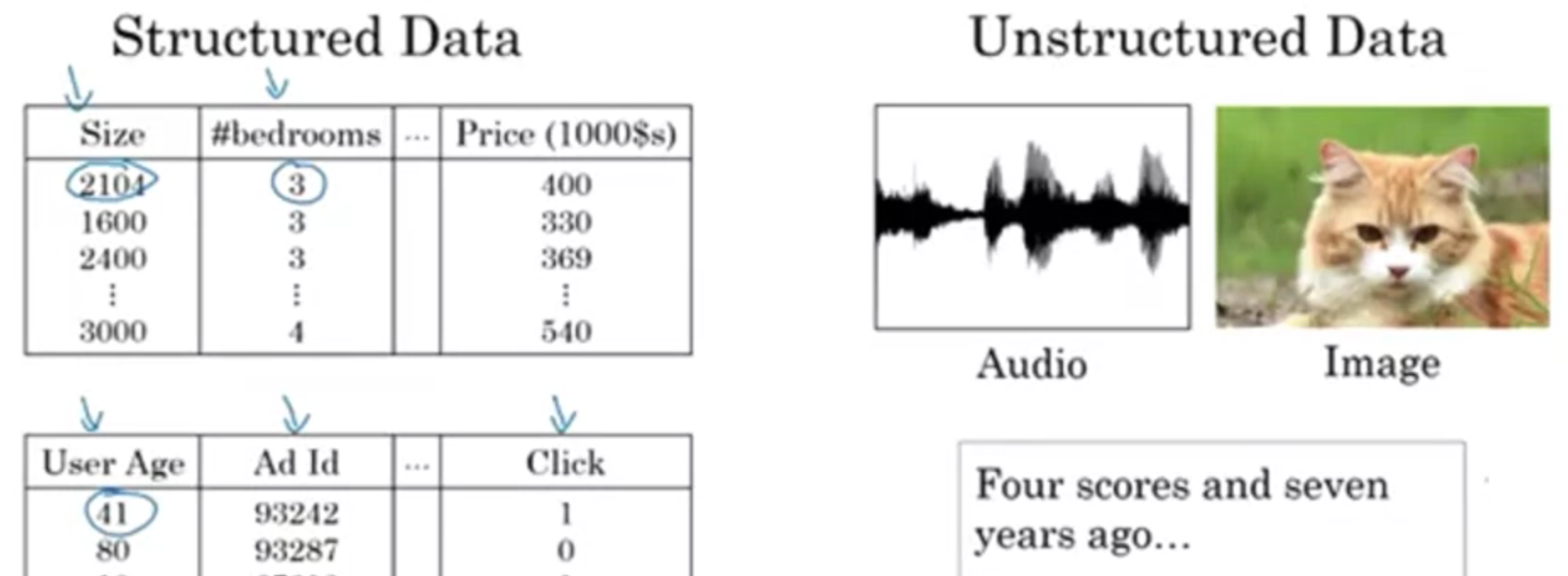
Hình ảnh các mạng :RNN dùng chuỗi 1 chiều và có thể có tính tạm thời   


Có 2 loại data :

trong đó structured là có các đặc điểm được xác định rõ rang từ đầu

Unstructured là dạng audio ,ảnh văn bản….

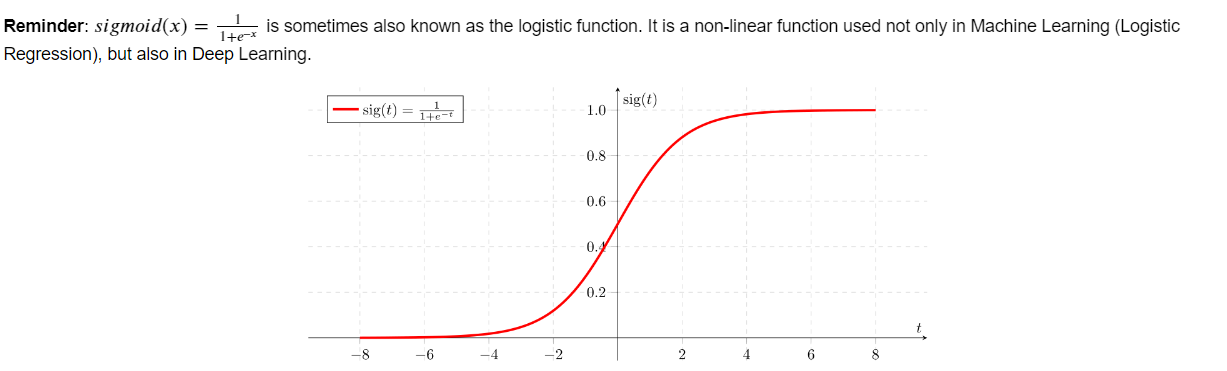
🡺nhờ mạng nơ ron máy tính đã học được data unstructed



Lập trình Numpy cơ bản :

np.exp() hữu dụng hơn math.exp() trong tạo hàm sigmoid ,còn được gọi là hàm logic,không chỉ dùng trong Machine Learning (Logistic Regression), but also in Deep Learning.

* Công thức và hình dạng của hàm



Chúng ta ít khi nào sử dụng math trong học sâu vì int là số thực ,trong khi học sâu chỉ sử dụng matrix và vecto nên ta mới dùng np.exp

Bài tập 3:sigmoid :

X hiện giờ có thể là 1 số thực ,matrix ,vecto

Các structure sử dụng để hiển thị numpy được gọi là numpy array

s = 1 / (1 + np.exp(-x)) thay thế cho math.exp

**Sigmoid Gradient**

+được dùng để tối ưu hóa loss function bằng backpropagation.

+tính bằng = sig .(1-sig)

Reshaping arrays định dạng lại mảng

+Hai hàm được sử dụng trong deep là  [np.shape](https://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.shape.html" \t "_blank) and [np.reshape()](https://docs.scipy.org/doc/numpy/reference/generated/numpy.reshape.html" \t "_blank).

[np.shape](https://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.shape.html" \t "_blank) để lấy ra  shape (dimension) của matrix hoặc vecto  
 [np.reshape()](https://docs.scipy.org/doc/numpy/reference/generated/numpy.reshape.html" \t "_blank).dùng để định hình lại X thành 1 chiều khác ( other dimension.)

🡺ví dụ 1 ảnh được hiển thị 3D với hình dạng  (𝑙𝑒𝑛𝑔𝑡ℎ,ℎ𝑒𝑖𝑔ℎ𝑡,𝑑𝑒𝑝𝑡ℎ=3)(length,height,depth=3).,tuy nhiên khi đọc vào thuật toán sẽ chuyển nó thành 1 D với dạng   
(𝑙𝑒𝑛𝑔𝑡ℎ∗ℎ𝑒𝑖𝑔ℎ𝑡∗3,1) ,phương pháp này gọi là unroll

Image2Vector:

+image2vector() triển khai để inp vào là 1 ảnh có hình dạng (a,b,c) tra ra là (a\*b,c)

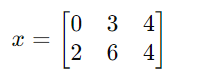
🡺v **=** v.reshape((v.shape[0] **\*** v.shape[1], v.shape[2])) *# v.shape[0] = a ; v.shape[1] = b ; v.shape[2] = c*

🡺reshape để chuyển chiều của nó lại

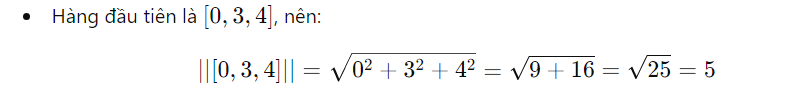
### Normalizing rows

+thường dùng trong machine và deep vì gradient descent diễn ra nhanh hơn sau khi normalizing

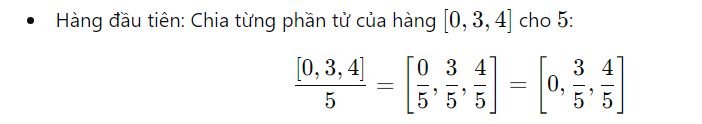
* Cách tính norm



Tính bằng công thức Euclidean: là lấy từng giá trị trong hàm để tính ,từng hàng

  
🡺 ta ra được 5/căn 56

+cuối cùng ta chia từng hàng cho phần tử tương ứng



### normalize\_rows

Sauk hi áp dụng vào một ma trận x, thì mỗi hàng của x sẽ có độ dài là 1 (độ dài đơn vị)

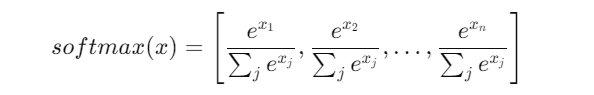
🡺code tính norm: x\_norm = np.linalg.norm(x,ord = 2,axis=1,keepdims=True)

🡺mỗi lần chạy sẽ có nhiều shape khác nhau vì nó lấy chuẩn theo hàng khác số cột

### softmax

+dùng để chuẩn hóa khi muốn phân loại 2 hay nhiều lớp

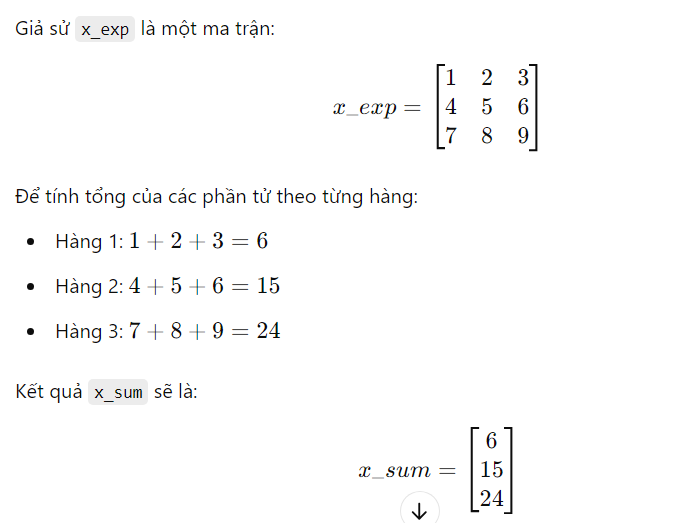
🡺công thức tính cho cả vecto và matrix



🡺tính cho vector: Ở đây, giá trị của từng phần tử xix\_ixi​ được mũ hóa bởi eee, sau đó chia cho tổng tất cả các giá trị exje^{x\_j}exj​ trong vector.

🡺tính theo từng hàng của matrix , hàm **softmax** chuẩn hóa các giá trị bằng cách biến chúng thành xác suất, làm cho tổng các giá trị trong một hàng bằng 1.

🡺là ta sẽ tính tổng theo hàng nếu axist là 1 và dọc nếu 0



## 2 - Vectorization

+thực hiện các phép toán dot pro,outer pro,,element multiplicat,general dot pro để so sánh hiệu xuất

1. lấy từng phần tử của 2 vecto nhân lần lượt cho nhau
2. tạo 1 matrix từ chứa ,trong đó mỗi vị trí là tích tương ứng của a1 và b1 ,a2 và b2…
3. lấy từng phần tử nhân với nhau
4. tạo 1 matrix ngẫu nhiên với 3x15 ,sau đó lấy từng phần tử của w để nhân với x tương ứng ,rồi sum lại cho từng hàng

🡺process time là thời gian chạy cho từng phép toán

### Implement the L1 and L2 loss functions

Loss function tính tổng sai lệch giữa dự đoán và thực thế y mũ và y ,loss càng lớn càng kém chính xác

L1= |y1 –y mũ (1)|+|y2-y mũ (2)|…..

+sai lệch lớn hay nhỏ phụ thuộc vào phạm vi y và độ sai lệch bạn mong muốn

L2: Sai số bình phương trung bình (MSE),giúp loại bỏ các giá trị âm