

>>> Um seminário sobre aprendizagem

>>> Seminário de Coisas Legais

Name: Lucas Giraldi Almeida Coimbra

Date: 29 de setembro de 2023

Grafos

```
>>> Grafos
```

```
Grafos:
```

```
>>> Grafos
```

```
Grafos:
```

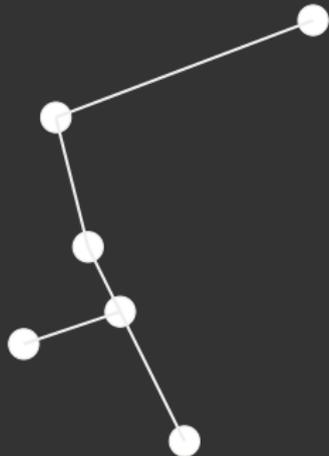
```
* vértices
```



>>> Grafos

Grafos:

* vértices e arestas;

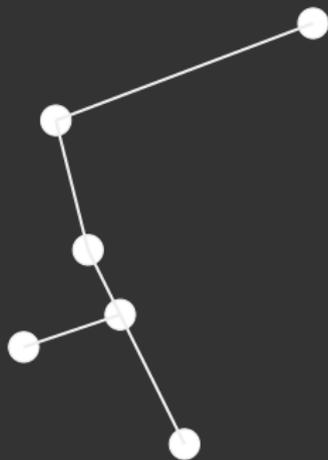


>>> Grafos

Grafos:

- * vértices e arestas;

Podem representar:



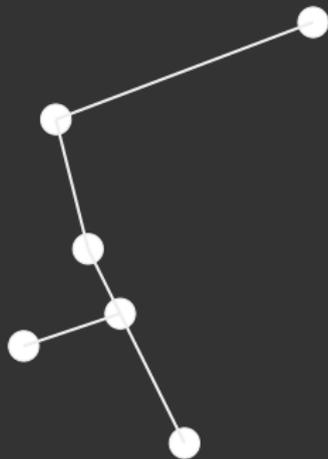
>>> Grafos

Grafos:

- * vértices e arestas;

Podem representar:

- * Substâncias químicas;



>>> Grafos

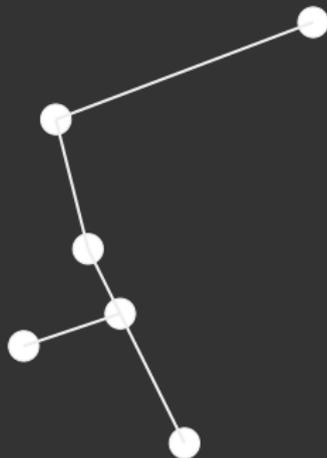
Grafos:

- * vértices e arestas;

Podem representar:

- * Substâncias químicas;

- * Redes sociais;



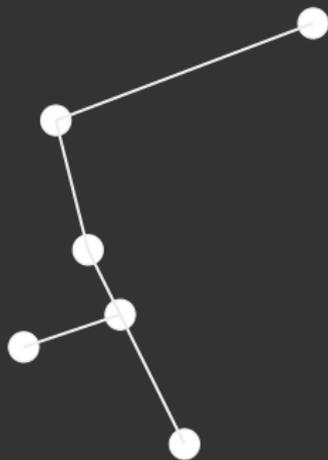
>>> Grafos

Grafos:

- * vértices e arestas;

Podem representar:

- * Substâncias químicas;
- * Redes sociais;
- * Cidades interligadas;



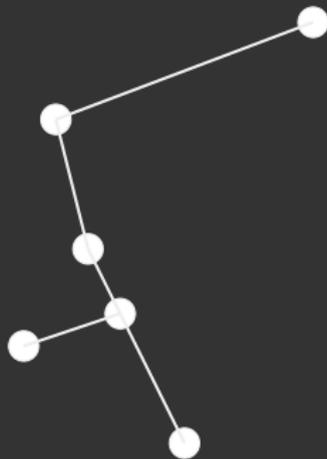
>>> Grafos

Grafos:

- * vértices e arestas;

Podem representar:

- * Substâncias químicas;
- * Redes sociais;
- * Cidades interligadas;
- * Artigos que referenciam um ao outro;



Redes Neurais


```
>>> Neurônios
```

```
Neurônios:
```

>>> Neurônios

Neurônios:

- * Função: recebe coisas e devolve coisas;

>>> Neurônios

Neurônios:

- * Função: recebe coisas e devolve coisas;
- * Recebe: sinal elétrico (número) de outros neurônios;

>>> Neurônios

Neurônios:

- * Função: recebe coisas e devolve coisas;
- * Recebe: sinal elétrico (número) de outros neurônios;
- * Devolve: um sinal elétrico (número).

>>> Neurônios

Neurônios:

- * Função: recebe coisas e devolve coisas;
- * Recebe: sinal elétrico (número) de outros neurônios;
- * Devolve: um sinal elétrico (número).

IMPORTANTE

>>> Neurônios

Neurônios:

- * Função: recebe coisas e devolve coisas;
- * Recebe: sinal elétrico (número) de outros neurônios;
- * Devolve: um sinal elétrico (número).

IMPORTANTE

Algumas ligações são mais importantes do que outras.

>>> Ligações



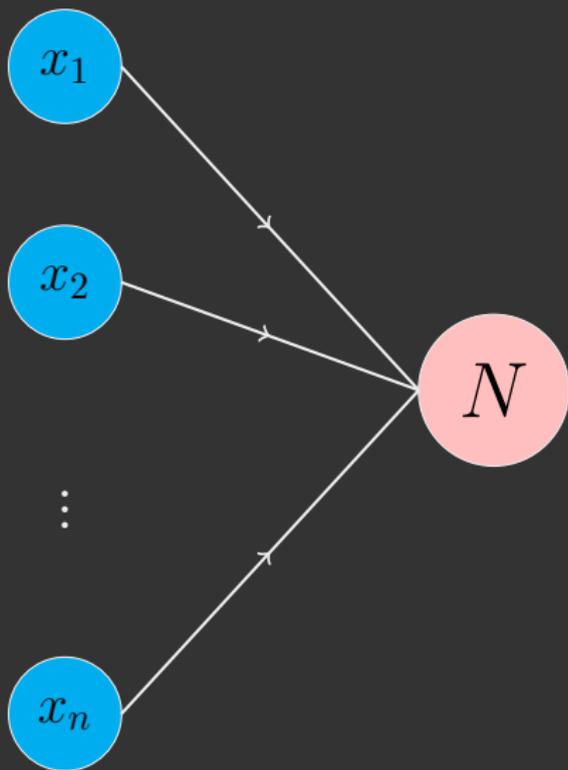
>>> Ligações



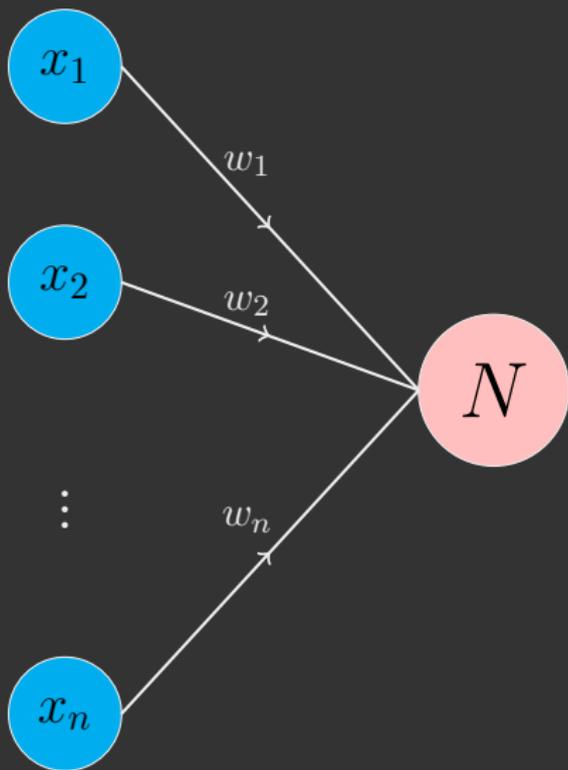
⋮



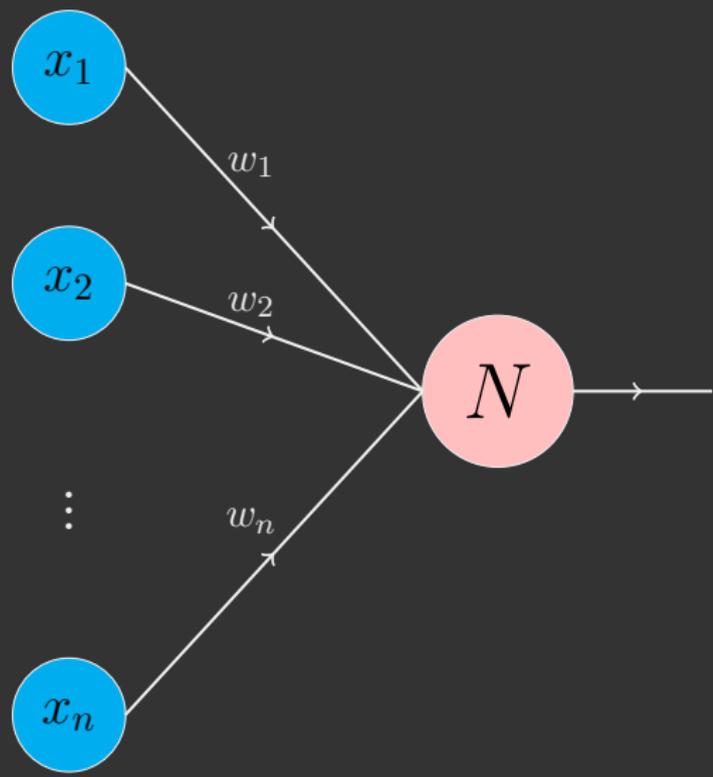
>>> Ligações



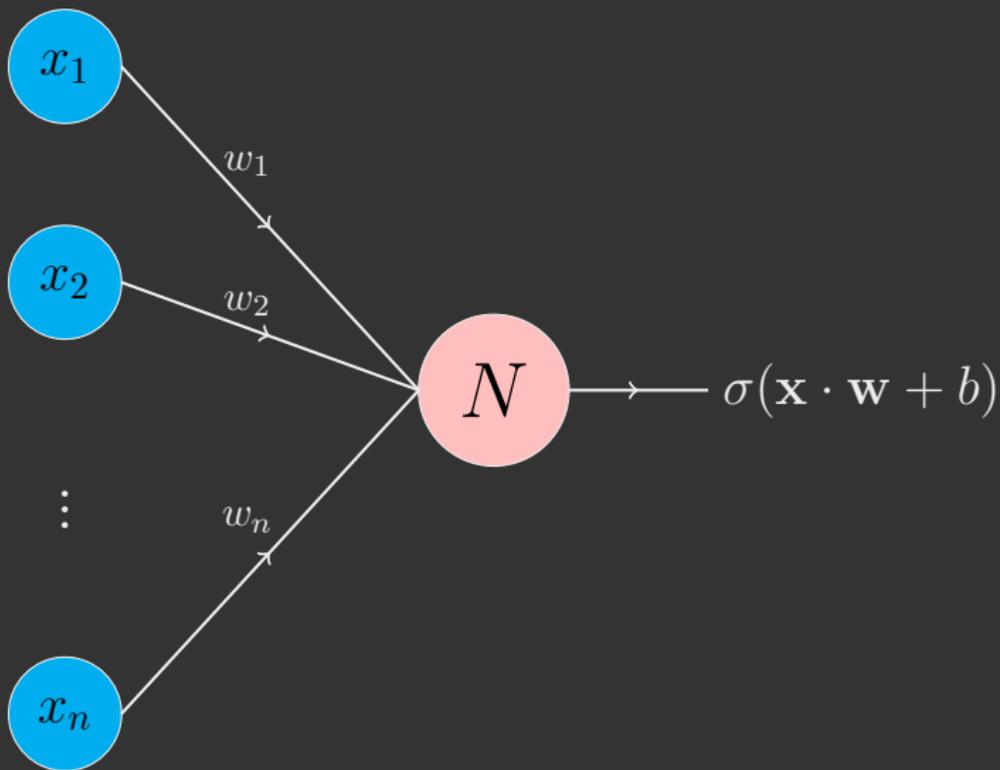
>>> Ligações



>>> Ligações



>>> Ligações



>>> Explicando

$$\sigma(\mathbf{x} \cdot \mathbf{w} + b)$$

>>> Explicando

$$\sigma(\mathbf{x} \cdot \mathbf{w} + b)$$

* $\mathbf{x} = (x_1, x_2, \dots, x_n)$ - sinais;

>>> Explicando

$$\sigma(\mathbf{x} \cdot \mathbf{w} + b)$$

* $\mathbf{x} = (x_1, x_2, \dots, x_n)$ - sinais;

* $\mathbf{w} = (w_1, w_2, \dots, w_n)$ - pesos;

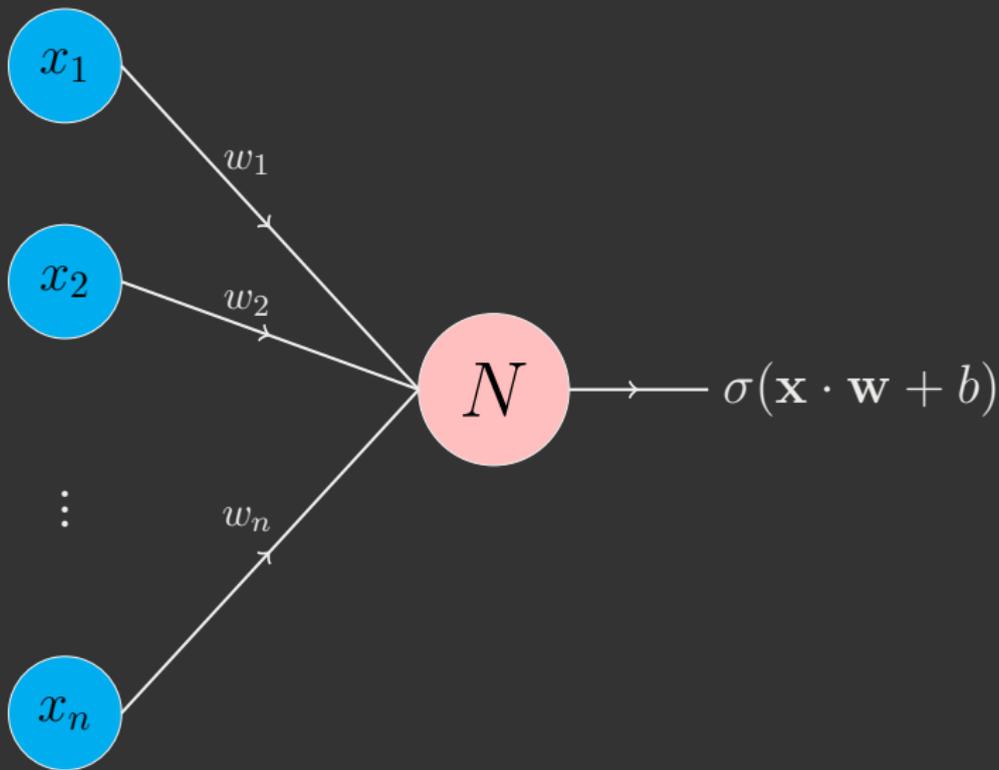
$$\sigma(\mathbf{x} \cdot \mathbf{w} + b)$$

- * $\mathbf{x} = (x_1, x_2, \dots, x_n)$ - sinais;
- * $\mathbf{w} = (w_1, w_2, \dots, w_n)$ - pesos;
- * $b \in \mathbb{R}$ - bias;

$$\sigma(\mathbf{x} \cdot \mathbf{w} + b)$$

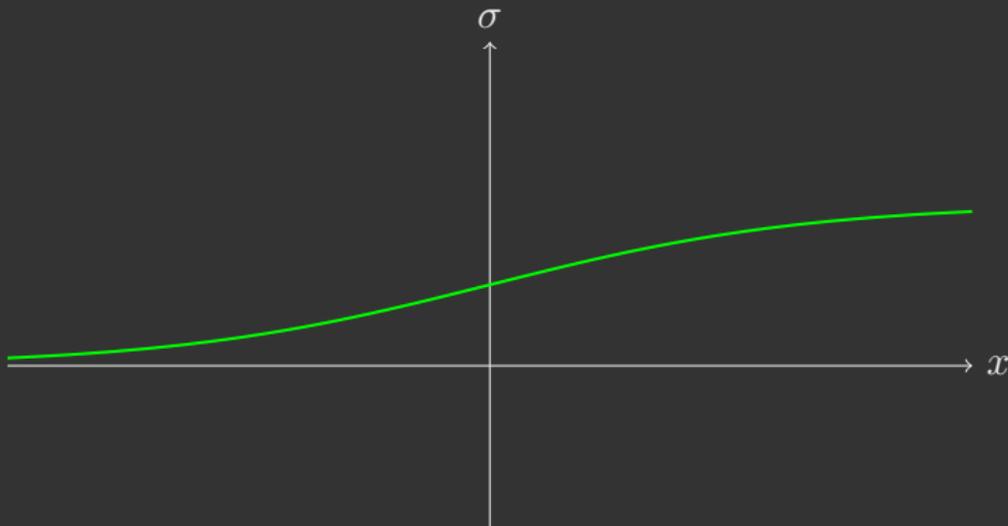
- * $\mathbf{x} = (x_1, x_2, \dots, x_n)$ - sinais;
- * $\mathbf{w} = (w_1, w_2, \dots, w_n)$ - pesos;
- * $b \in \mathbb{R}$ - *bias*;
- * $\sigma: \mathbb{R} \rightarrow \mathbb{R}$ - função de ativação.

>>> Neurônio (de novo)



>>> Função de ativação

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



>>> Aprendendo

Por que aprender?

>>> Aprendendo

Por que aprender?

- * Dado x , queremos que o neurônio transmita um valor t ;

>>> Aprendendo

Por que aprender?

- * Dado x , queremos que o neurônio transmita um valor t ;
- * Com os pesos w e o *bias* b , o neurônio está transmitindo um valor y .

>>> Aprendendo

Por que aprender?

- * Dado x , queremos que o neurônio transmita um valor t ;
- * Com os pesos w e o *bias* b , o neurônio está transmitindo um valor y .

O que é aprender?

>>> Aprendendo

Por que aprender?

- * Dado x , queremos que o neurônio transmita um valor t ;
- * Com os pesos w e o *bias* b , o neurônio está transmitindo um valor y .

O que é aprender?

- * Calcular erro: $E(y) = |y - t|$;

>>> Aprendendo

Por que aprender?

- * Dado x , queremos que o neurônio transmita um valor t ;
- * Com os pesos w e o *bias* b , o neurônio está transmitindo um valor y .

O que é aprender?

- * Calcular erro: $E(y) = |y - t|$;
- * Mudar w e b para diminuir $E(y)$.

>>> Aprendi, e agora?

- * Situação: queremos ordenar pessoas para atendimento em um pronto-socorro;

>>> Aprendi, e agora?

- * Situação: queremos ordenar pessoas para atendimento em um pronto-socorro;
- * Cada pessoa é representada por um vetor x que contém dados numéricos da ficha médica, como pressão, temperatura e peso;

>>> Aprendi, e agora?

- * Situação: queremos ordenar pessoas para atendimento em um pronto-socorro;
- * Cada pessoa é representada por um vetor x que contém dados numéricos da ficha médica, como pressão, temperatura e peso;
- * Urgência: número entre 0 e 1.

>>> Aprendi, e agora?

- * Situação: queremos ordenar pessoas para atendimento em um pronto-socorro;
- * Cada pessoa é representada por um vetor x que contém dados numéricos da ficha médica, como pressão, temperatura e peso;
- * Urgência: número entre 0 e 1. Se for 0, não é urgente, se for 1, precisa de atendimento rápido;

>>> Aprendi, e agora?

- * Situação: queremos ordenar pessoas para atendimento em um pronto-socorro;
- * Cada pessoa é representada por um vetor x que contém dados numéricos da ficha médica, como pressão, temperatura e peso;
- * Urgência: número entre 0 e 1. Se for 0, não é urgente, se for 1, precisa de atendimento rápido;
- * Sabemos: os dados da pessoa x e qual a urgência dela;

>>> Aprendi, e agora?

- * Situação: queremos ordenar pessoas para atendimento em um pronto-socorro;
- * Cada pessoa é representada por um vetor x que contém dados numéricos da ficha médica, como pressão, temperatura e peso;
- * Urgência: número entre 0 e 1. Se for 0, não é urgente, se for 1, precisa de atendimento rápido;
- * Sabemos: os dados da pessoa x e qual a urgência dela;
- * Treinamos o neurônio para que, ao receber x , retorne a sua urgência;

>>> Aprendi, e agora?

- * Situação: queremos ordenar pessoas para atendimento em um pronto-socorro;
- * Cada pessoa é representada por um vetor x que contém dados numéricos da ficha médica, como pressão, temperatura e peso;
- * Urgência: número entre 0 e 1. Se for 0, não é urgente, se for 1, precisa de atendimento rápido;
- * Sabemos: os dados da pessoa x e qual a urgência dela;
- * Treinamos o neurônio para que, ao receber x , retorne a sua urgência;
- * Esperamos que, ao receber uma pessoa x' , o neurônio retorne como seu sinal, mais ou menos corretamente, a urgência de x' .

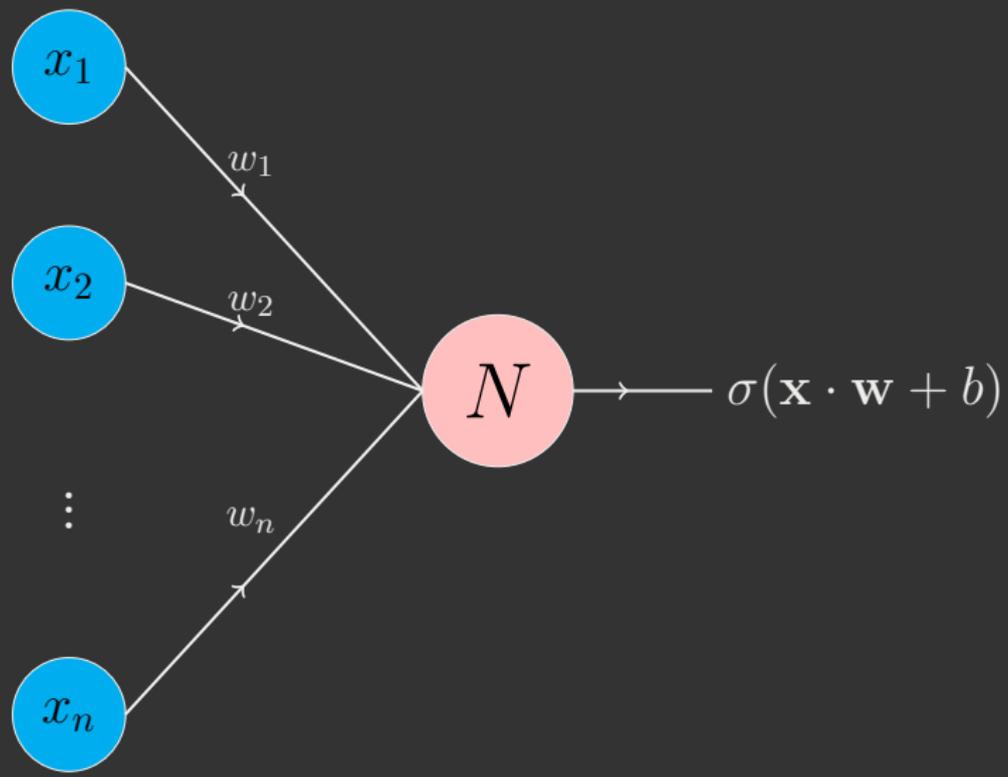
>>> Só um?

Neurônios são poderosos sozinhos.

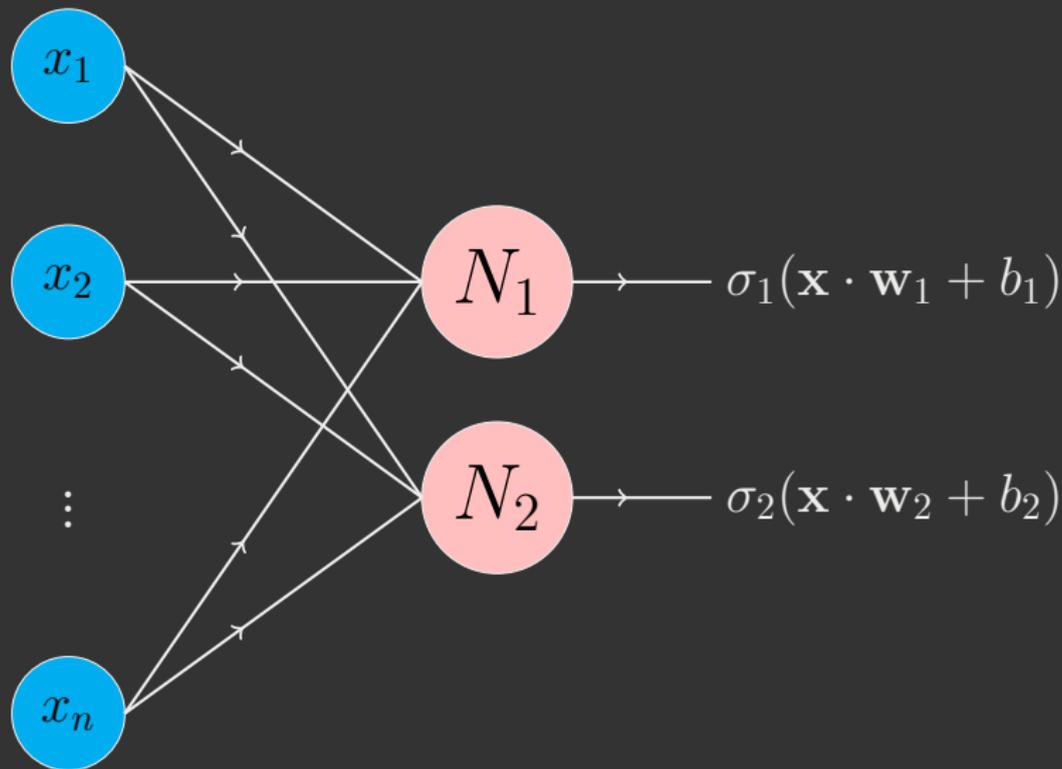
>>> Só um?

Neurônios são poderosos sozinhos.
E se pegarmos vários?

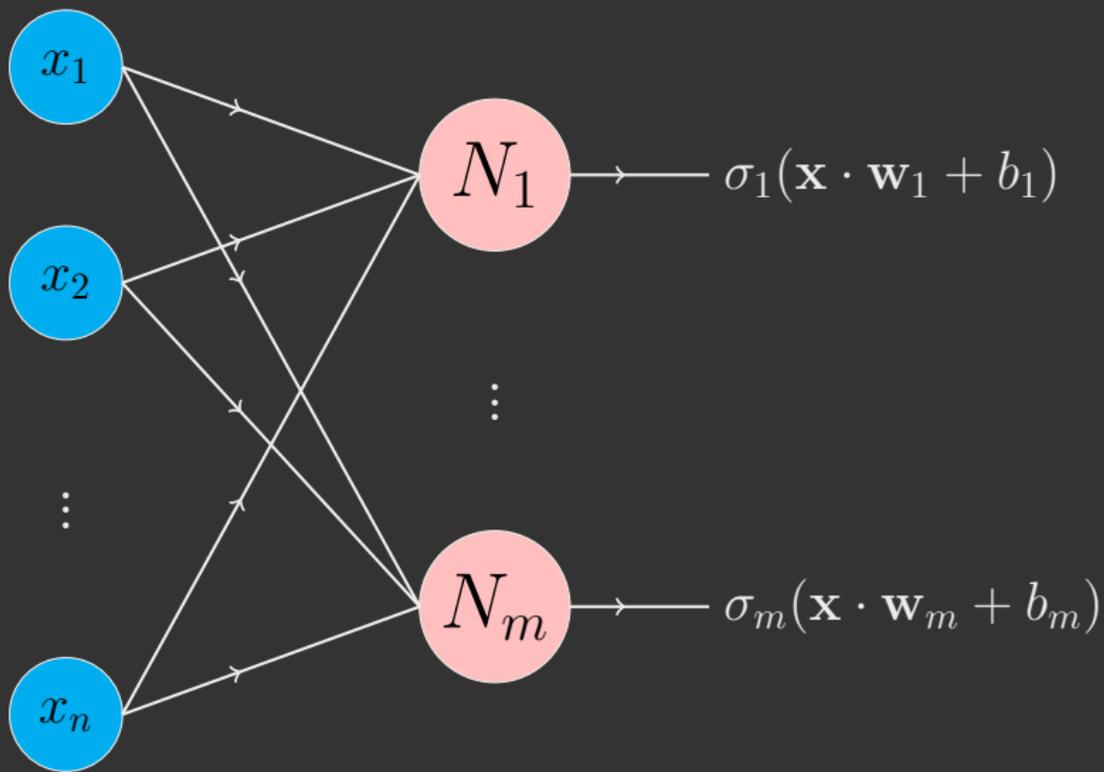
>>> Rede Neural



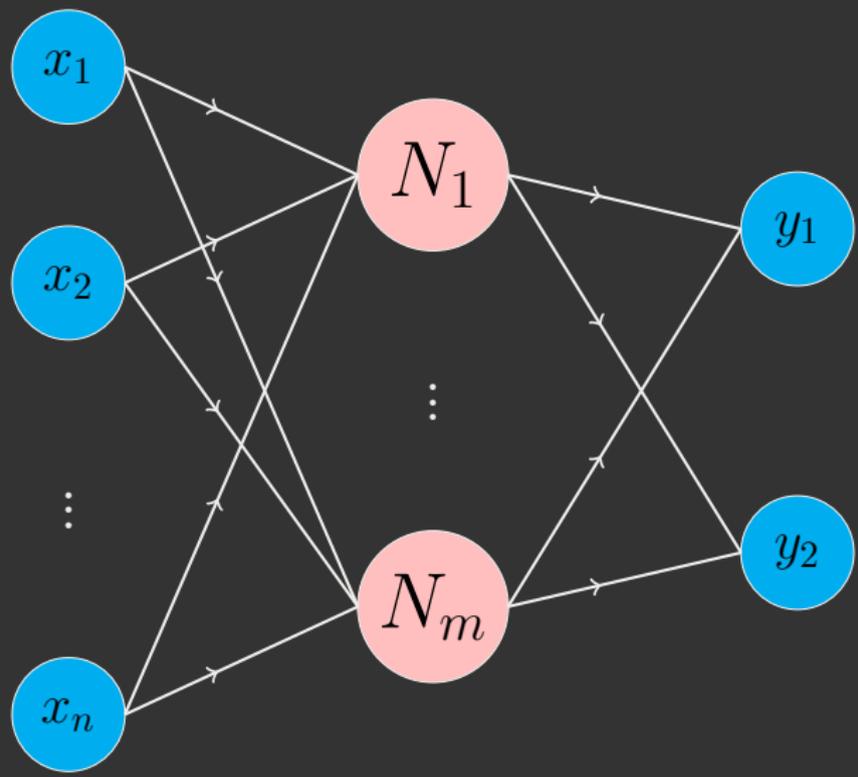
>>> Rede Neural



>>> Rede Neural



>>> Rede Neural



>>> Aprendendo (de novo)

A ideia é a mesma, mas agora temos $\mathbf{y} = (y_1, y_2)$, $\mathbf{t} = (t_1, t_2)$ e o erro

$$E(\mathbf{y}) = \|\mathbf{y} - \mathbf{t}\|.$$

>>> Aprendendo (de novo)

A ideia é a mesma, mas agora temos $\mathbf{y} = (y_1, y_2)$, $\mathbf{t} = (t_1, t_2)$ e o erro

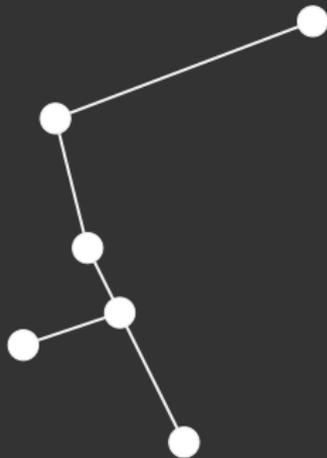
$$E(\mathbf{y}) = \|\mathbf{y} - \mathbf{t}\|.$$

PROBLEMA

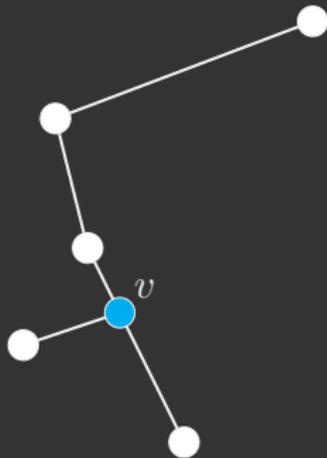
Ligações de uma camada interferem no valor de neurônios de outras camadas: treinamento é mais difícil.

Convolução

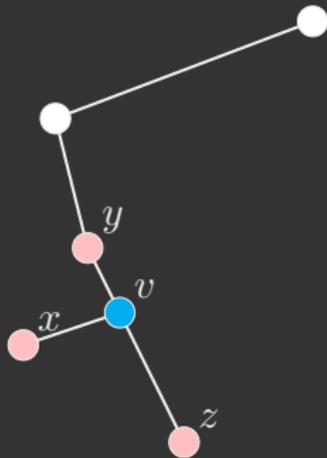
```
>>> Vizinhos e Grau
```



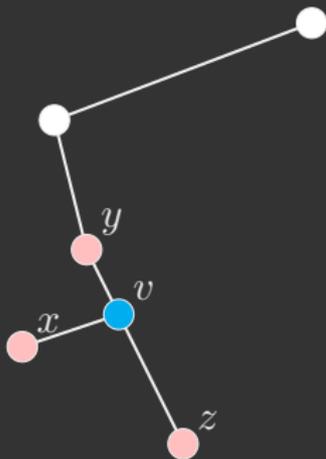
```
>>> Vizinhos e Grau
```



```
>>> Vizinhos e Grau
```

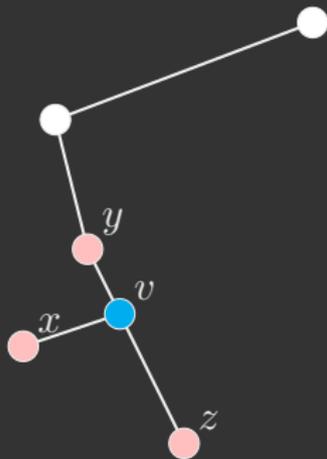


```
>>> Vizinhos e Grau
```



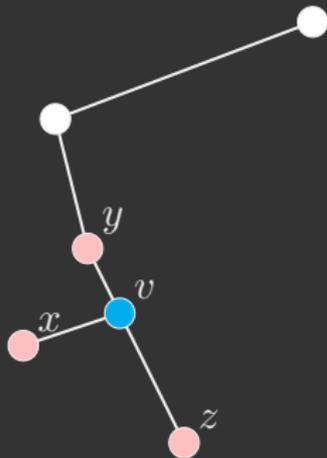
Vizinhança: pontos rosas.

>>> Vizinhos e Grau



Vizinhança: pontos rosas. Nesse caso, $\mathcal{N}(v) = \{x, y, z\}$.

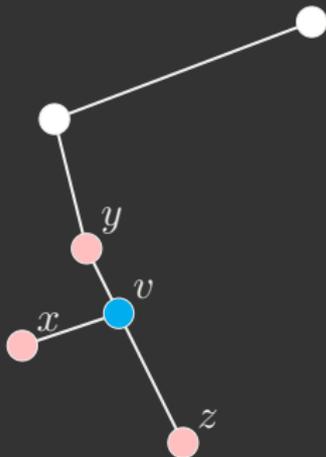
>>> Vizinhos e Grau



Vizinhança: pontos rosas. Nesse caso, $\mathcal{N}(v) = \{x, y, z\}$.

Grau: número de vizinhos.

>>> Vizinhos e Grau



Vizinhança: pontos rosas. Nesse caso, $\mathcal{N}(v) = \{x, y, z\}$.

Grau: número de vizinhos. Nesse caso, $\deg(v) = 3$.

>>> Convolução

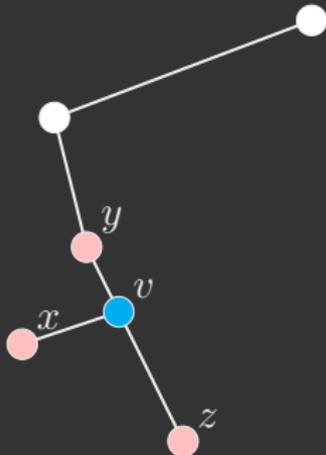
PROBLEMA

O número $\text{deg}(v)$ só contém informação sobre o vértice v .

>>> Convolução

PROBLEMA

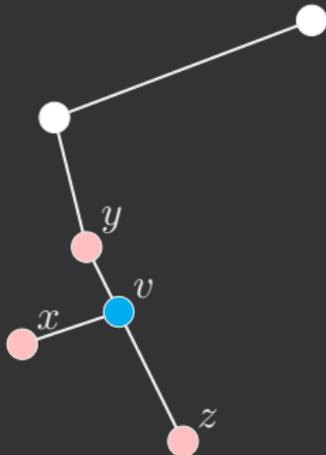
O número $\text{deg}(v)$ só contém informação sobre o vértice v .



>>> Convolução

PROBLEMA

O número $\text{deg}(v)$ só contém informação sobre o vértice v .

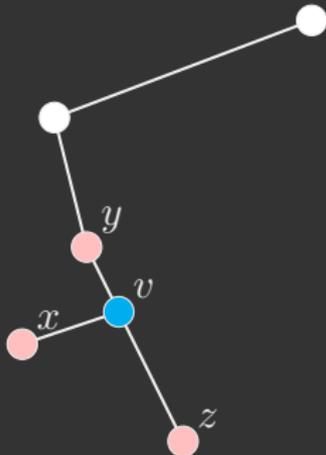


$$\text{deg}_2(v) = \frac{\text{deg}(v) + \text{deg}(x) + \text{deg}(y) + \text{deg}(z)}{4}$$

>>> Convolução

PROBLEMA

O número $\text{deg}(v)$ só contém informação sobre o vértice v .



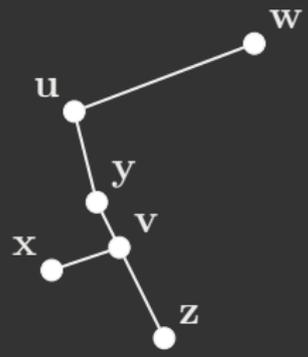
$$\text{deg}_2(v) = \frac{\text{deg}(v) + \text{deg}(x) + \text{deg}(y) + \text{deg}(z)}{4}$$

O número $\text{deg}_2(v)$ contém informação sobre v e seus vizinhos.

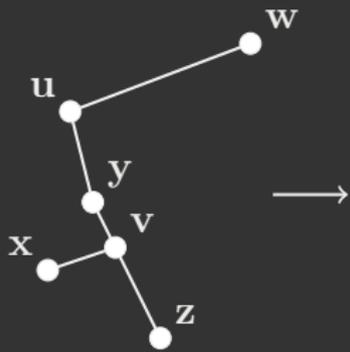
GNNs – Graph Neural Networks

Redes Neurais em Grafos

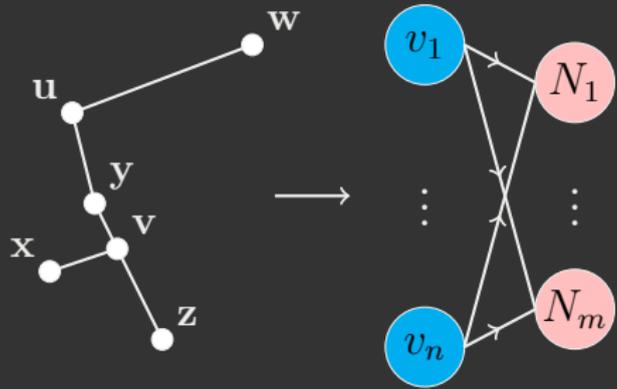
>>> GNNs - Primeiro passo



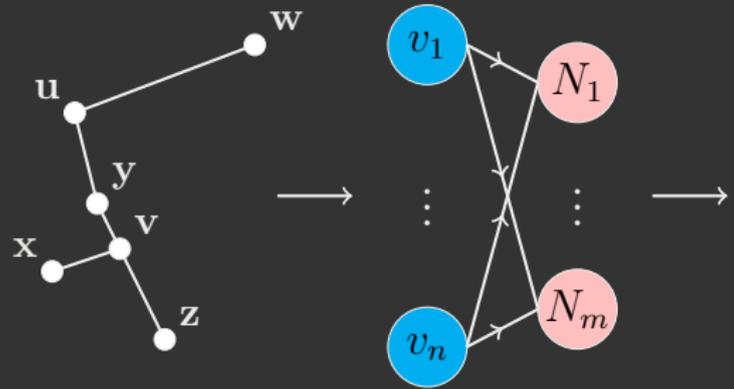
>>> GNNs - Primeiro passo



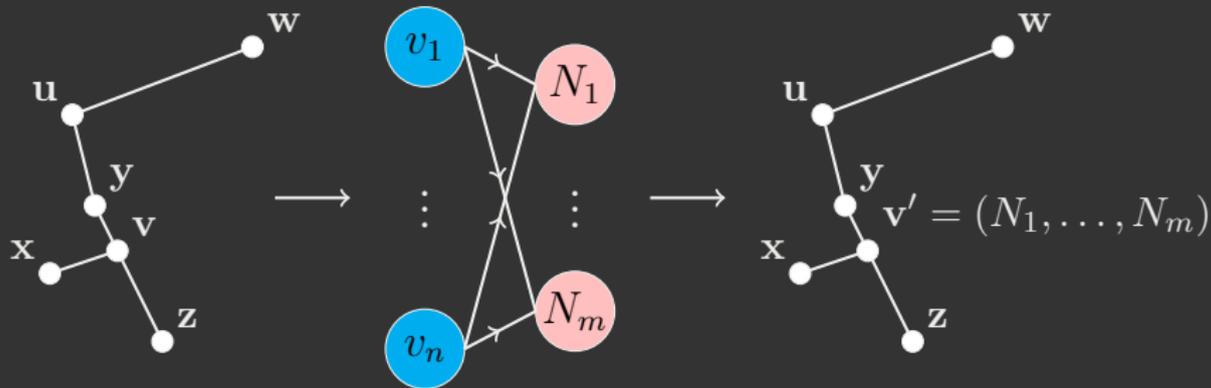
>>> GNNs - Primeiro passo



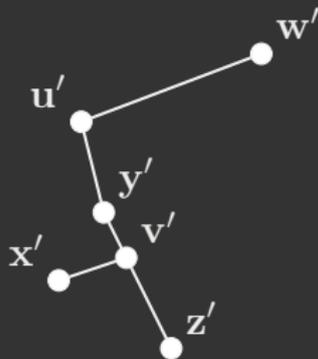
>>> GNNs - Primeiro passo



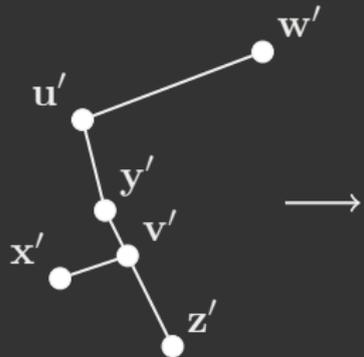
>>> GNNs - Primeiro passo



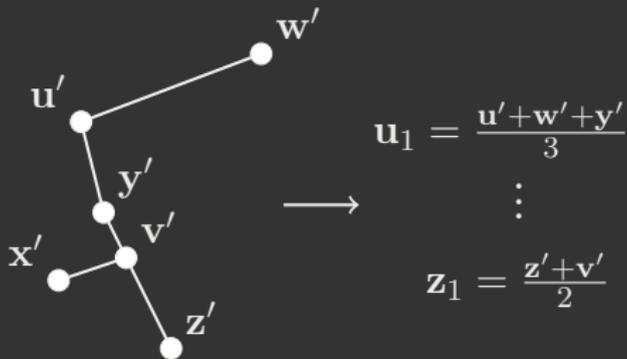
>>> GNNs - Segundo passo



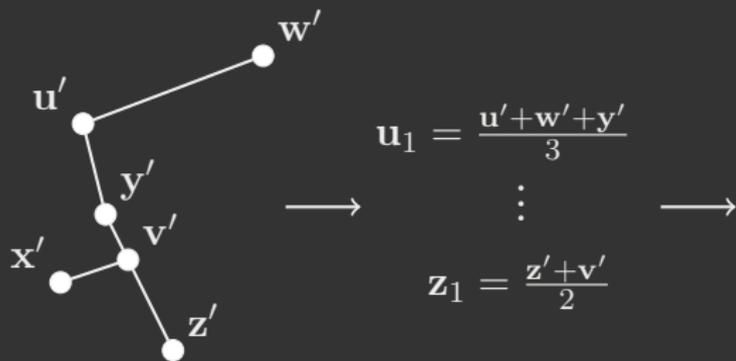
>>> GNNs - Segundo passo



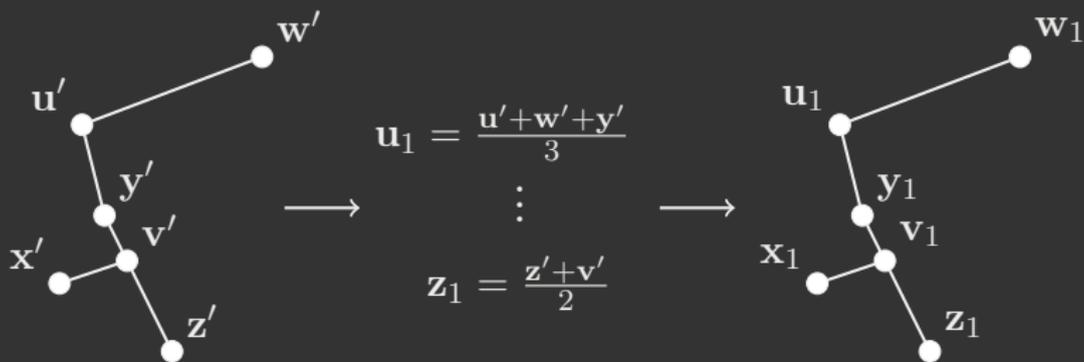
>>> GNNs - Segundo passo



>>> GNNs - Segundo passo



>>> GNNs - Segundo passo

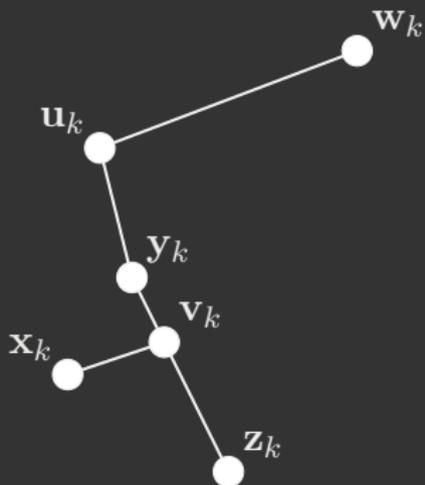


>>> Conclusão

Repetindo o processo k vezes, obtemos

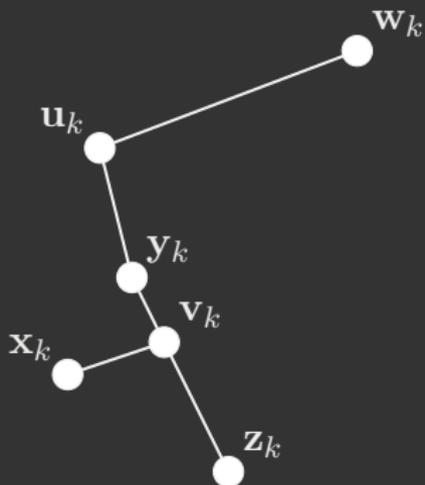
>>> Conclusão

Repetindo o processo k vezes, obtemos



>>> Conclusão

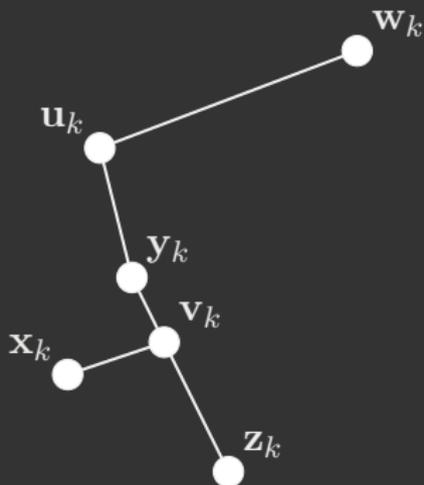
Repetindo o processo k vezes, obtemos



* Todas as listas tem o mesmo tamanho;

>>> Conclusão

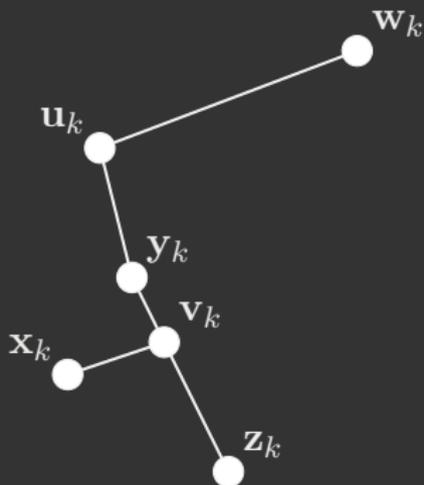
Repetindo o processo k vezes, obtemos



- * Todas as listas tem o mesmo tamanho;
- * São listas de números;

>>> Conclusão

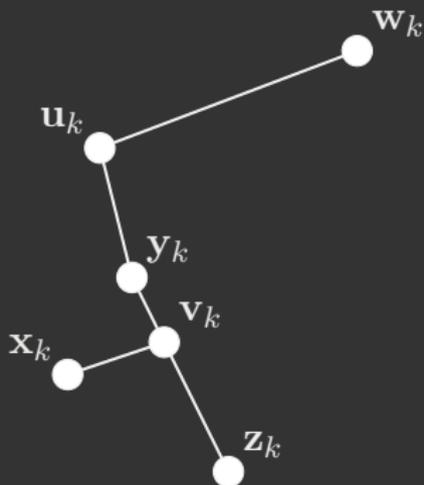
Repetindo o processo k vezes, obtemos



- * Todas as listas tem o mesmo tamanho;
- * São listas de números;
- * Podem ser interpretadas como elementos de algum \mathbb{R}^d !

>>> Conclusão

Repetindo o processo k vezes, obtemos



- * Todas as listas tem o mesmo tamanho;
- * São listas de números;
- * Podem ser interpretadas como elementos de algum \mathbb{R}^d !
- * Mergulhamos o grafo!!!!

>>> Aplicações

Com o poder da álgebra linear, podemos:

>>> Aplicações

Com o poder da álgebra linear, podemos:

- * Classificar vértices;

>>> Aplicações

Com o poder da álgebra linear, podemos:

- * Classificar vértices;
- * Novas arestas;

>>> Aplicações

Com o poder da álgebra linear, podemos:

- * Classificar vértices;
- * Novas arestas;
- * Classificar grafos;

>>> Aplicações

Com o poder da álgebra linear, podemos:

- * Classificar vértices;
- * Novas arestas;
- * Classificar grafos;
- * Agrupamento.

>>> Referências I

- [1] *Graph Neural Networks - a perspective from the ground up - YouTube*. Acessado: 29-09-2023. URL: <https://www.youtube.com/watch?v=GXhBEj1ZtE8>.
- [2] William L. Hamilton. *Graph Representation Learning (Synthesis Lectures on Artificial Intelligence and Machine Learning, 46)*. Morgan & Claypool Publishers, 2020. ISBN: 9781681739649.
- [3] *Introduction to Machine Learning | MIT Open Learning Library*. Acessado: 29-09-2023. URL: <https://openlearninglibrary.mit.edu/courses/course-v1:MITx+6.036+1T2019/about>.

>>> Referências II

- [4] *Perceptron Learning Algorithm: A Graphical Explanation Of Why It Works* | by Akshay L Chandra | Towards Data Science. Acessado: 25-09-2023. URL: <https://towardsdatascience.com/perceptron-learning-algorithm-d5db0deab975>.
- [5] *Training Deep Neural Networks. Deep Learning Accessories* | by Ravindra Parmar | Towards Data Science. Acessado: 25-09-2023. URL: <https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964>.