# Redes Neurais Artificiais Clássicas

Como selecionar os melhores parâmetros

# Quem sou eu

Mestranda na área de Inteligência Computacional na Engenharia Elétrica - UFMG.

Git:
https://github.com/JulianaGuama
Linkedin:
https://br.linkedin.com/in/juliana-guama

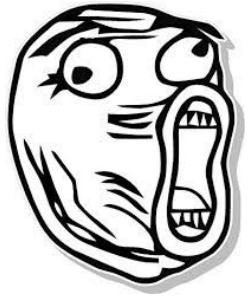
Seleciona os parâmetros por feeling

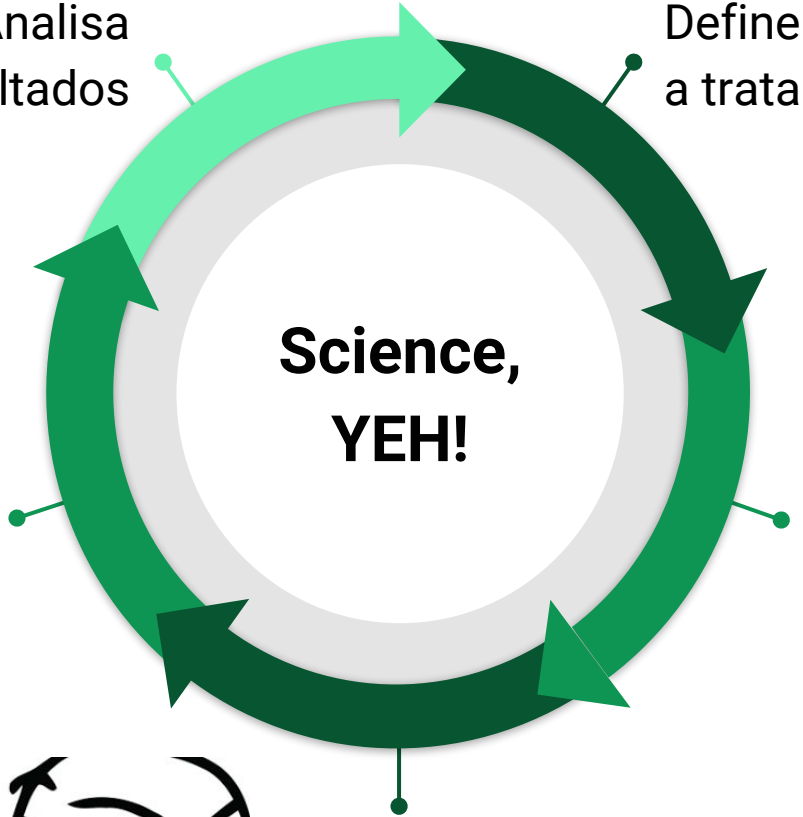
Planeja o experimento

# Ciência de dados é **CIÊNCIA!!!**


Analisa Resultados

Define problema a tratar



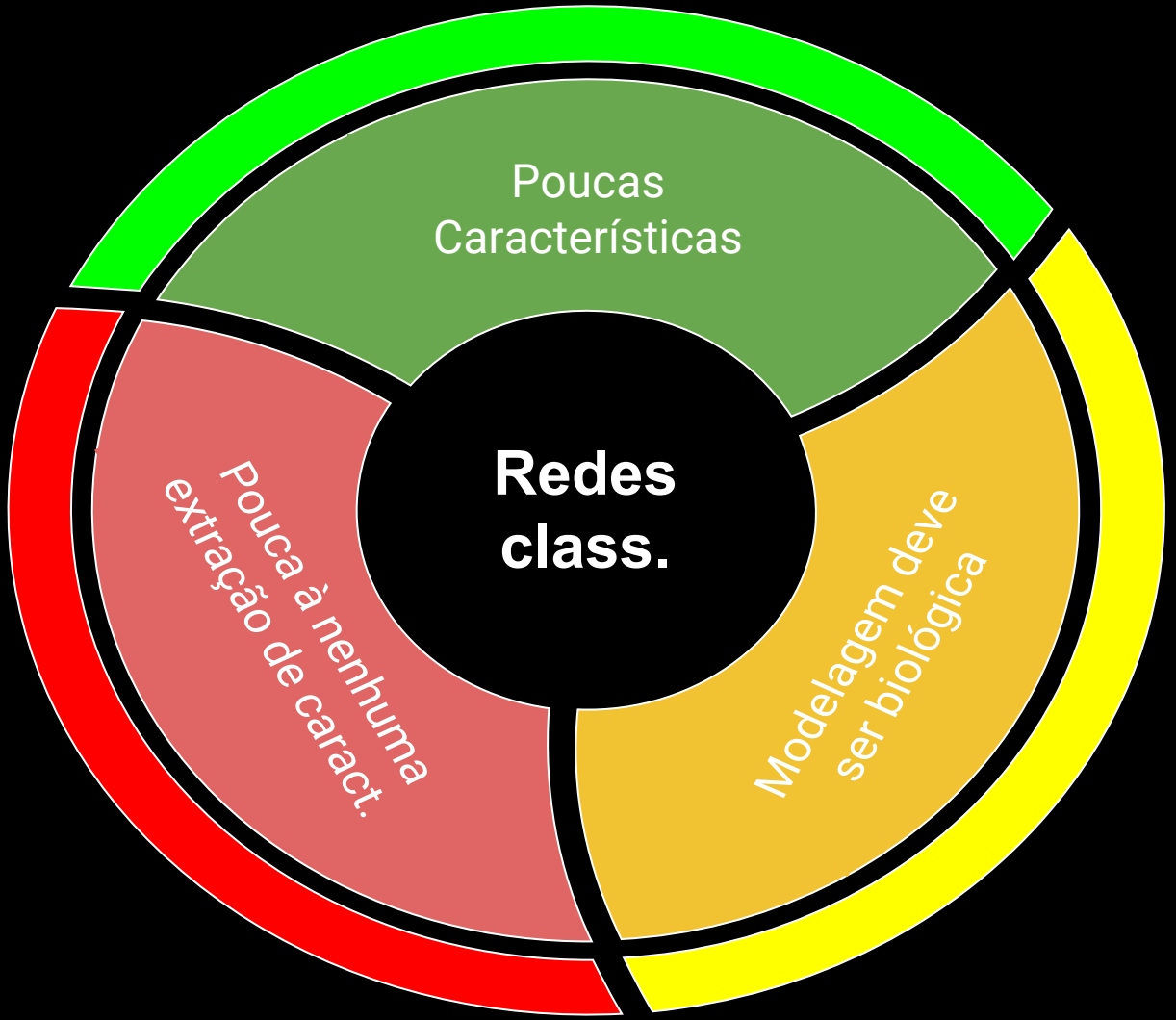



Testa Hipótese

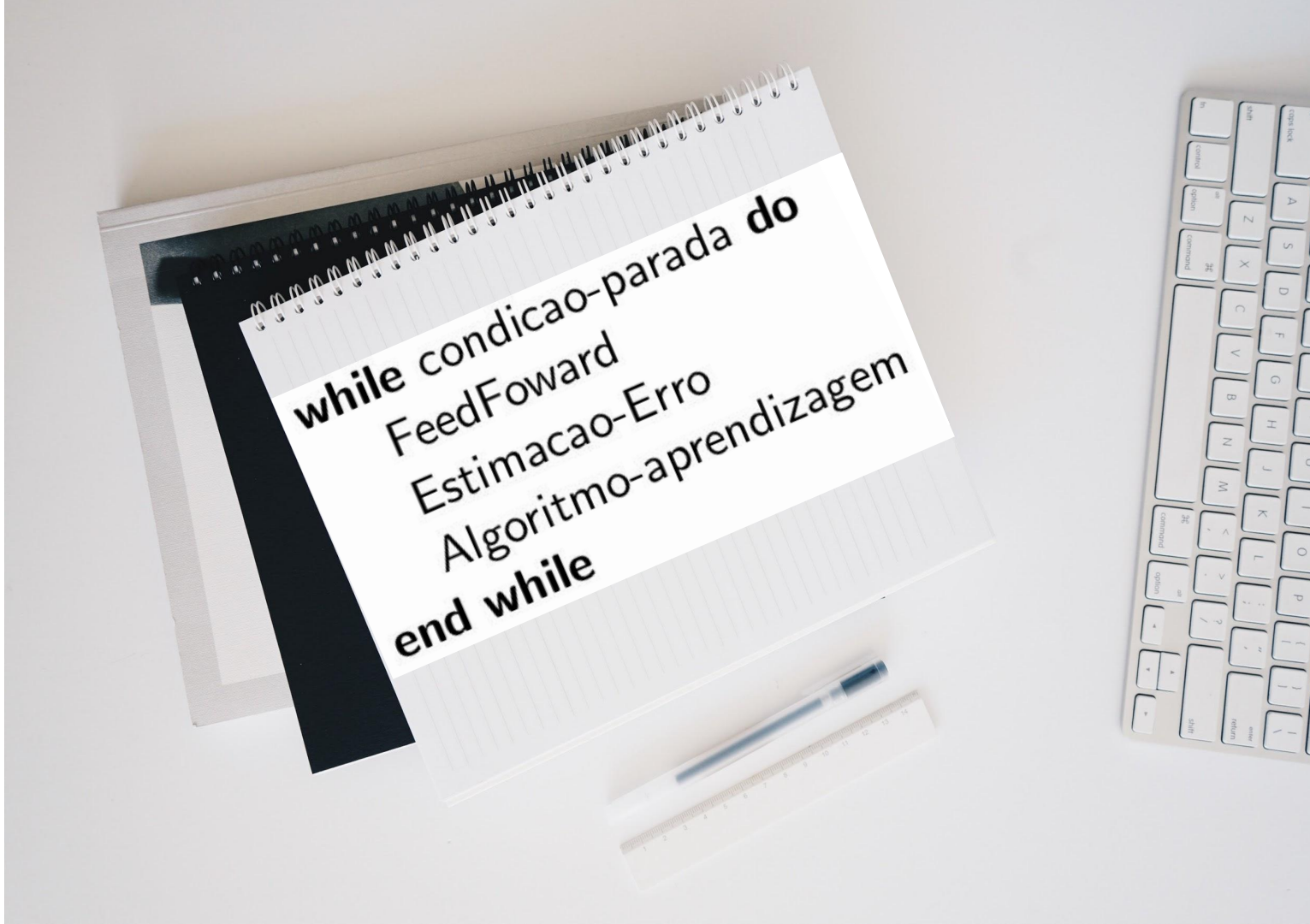
Coleta dados


Formaliza Hipótese


Por que escolher as rede clássicas?


Poucas Características
Redes class.
Pouca à nenhuma extração de caract.
Modelagem deve ser biológica

# Forma Geral das RNAs



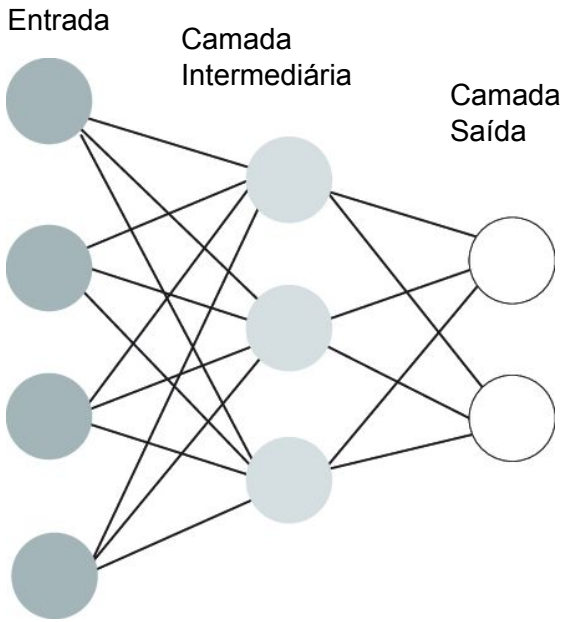
# Algumas RNAs tipo FeedFoward

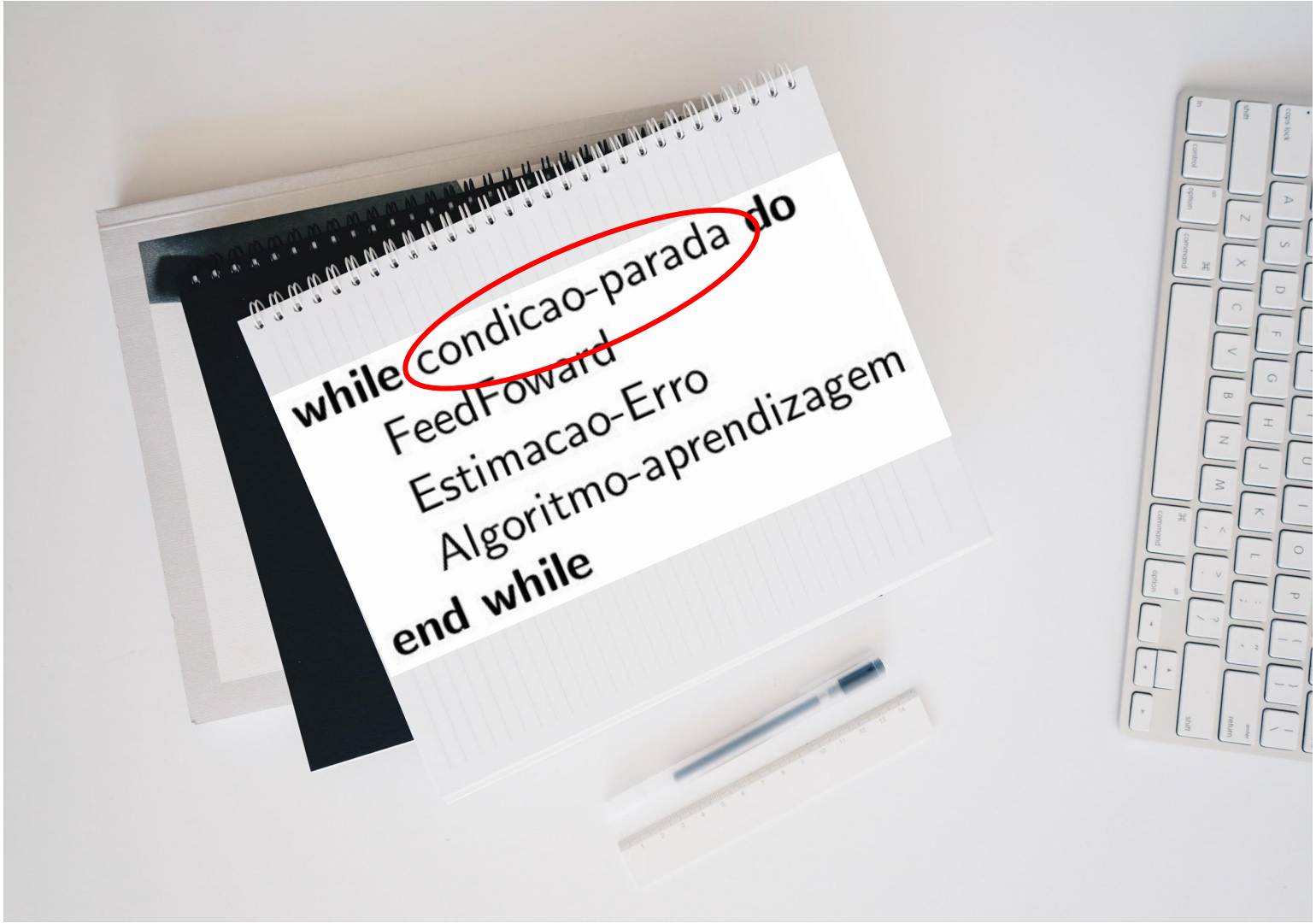
Uma camada:

- Perceptron
- Adaline

Multiplas Camadas:

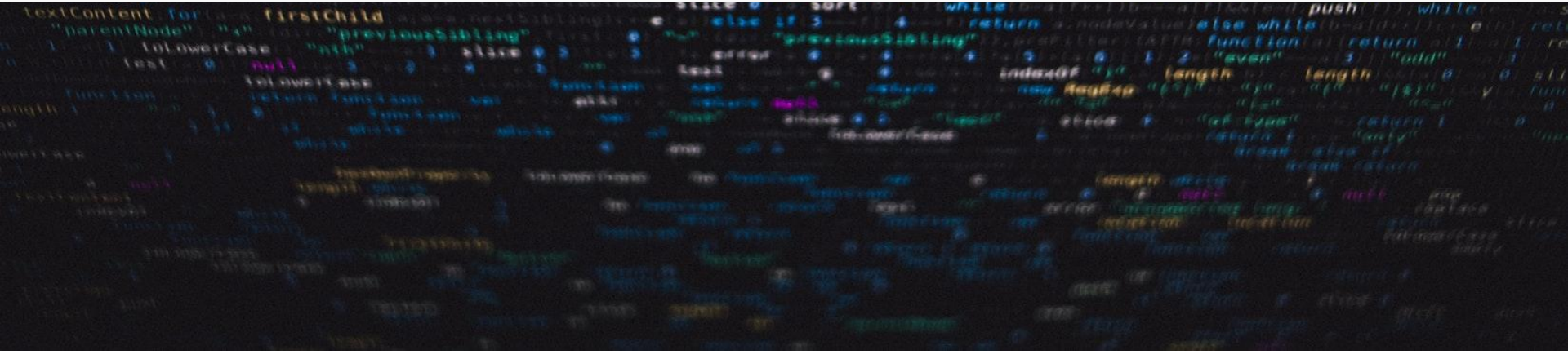
- Multi Layer Perceptron (MLP)
- Extreme Learning Machine (ELM)
- Redes de Base Radial (RBF)
- Classificador Bayesiano
- Rede Neural Probabilística (PNN)

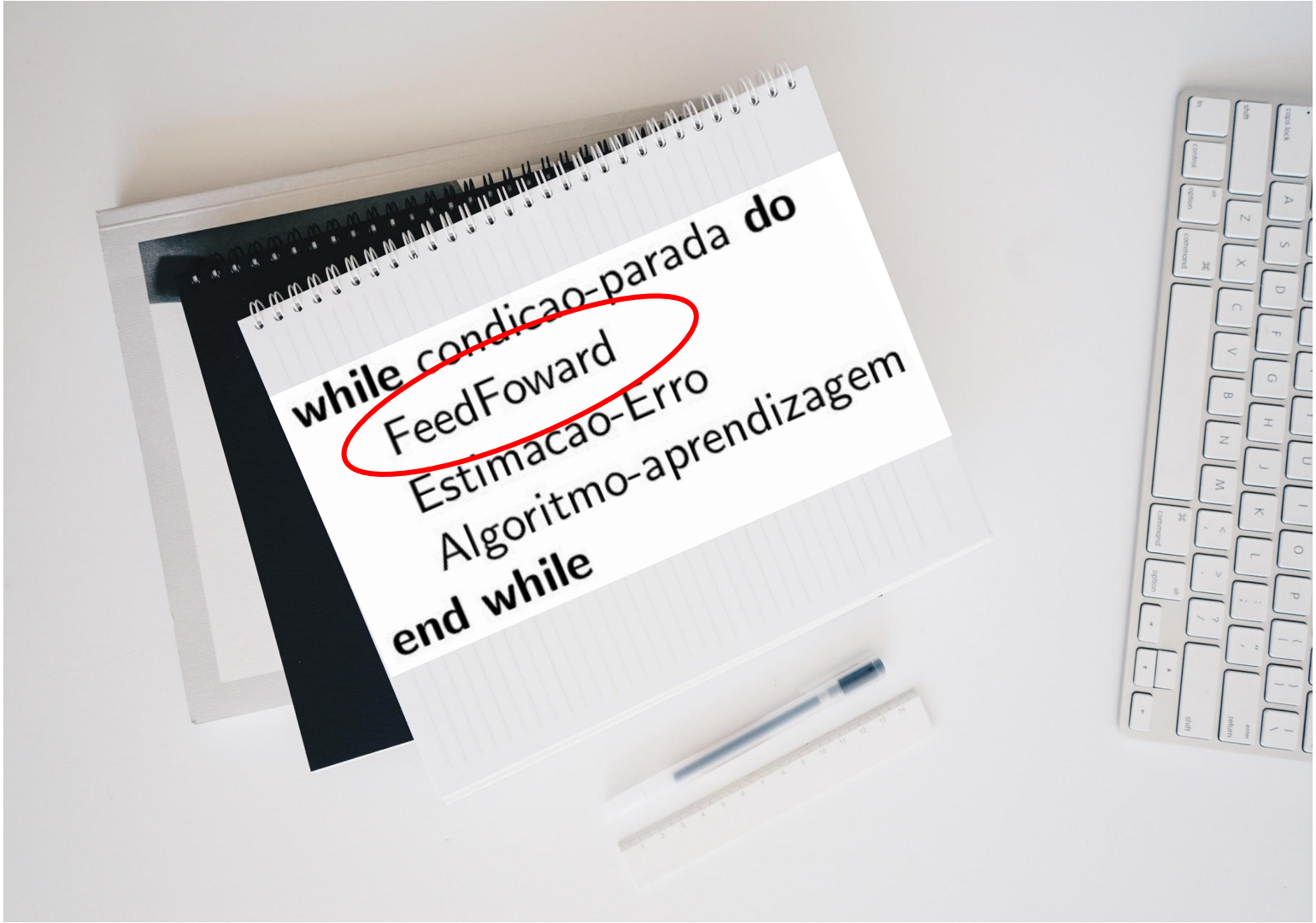



while condicao-parada do
FeedFoward
Estimacao-Erro
Algoritmo-aprendizagem
end while

# Condições de Parada

```
MLPClassifier(activation='relu', alpha=1e-05, batch_size='auto',
              beta_1=0.9, beta_2=0.999, early_stopping=False,
              epsilon=1e-08, hidden_layer_sizes=(100,) ,
              learning_rate='constant', learning_rate_init=0.001,
              max_iter=200, momentum=0.9, n_iter_no_change=10,
              nesterovs_momentum=True, power_t=0.5, random_state=1,
              shuffle=True, solver='lbfgs', tol=0.0001,
              validation_fraction=0.1, verbose=False, warm_start=False)
```




while condicao-parada do
FeedFoward
Estimacao-Erro
Algoritmo-aprendizagem
end while

# FeedFoward

-Teorema da Aproximação Universal [Geoge Cybenko 1989]

Tendo quantidade suficiente de neurônios, uma rede tipo ”feedfoward” com apenas 1 camada escondida tem capacidade de aproximar qualquer função contínua.

# FeedFoward

```
MLPClassifier(activation='relu', alpha=1e-05, batch_size='auto',
              beta_1=0.9, beta_2=0.999, early_stopping=False,
              epsilon=1e-08, hidden_layer_sizes=(100,),
              learning_rate='constant', learning_rate_init=0.001,
              max_iter=200, momentum=0.9, n_iter_no_change=10,
              nesterovs_momentum=True, power_t=0.5, random_state=1,
              shuffle=True, solver='lbfgs', tol=0.0001,
              validation_fraction=0.1, verbose=False, warm_start=False)
```

Alpha:penalização tipo L2

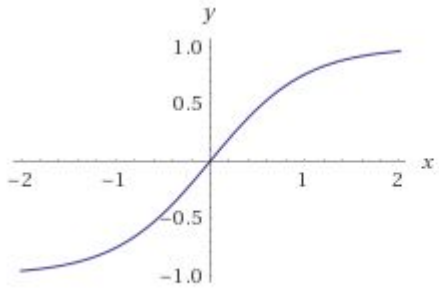
***0.0001***

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

Cost function

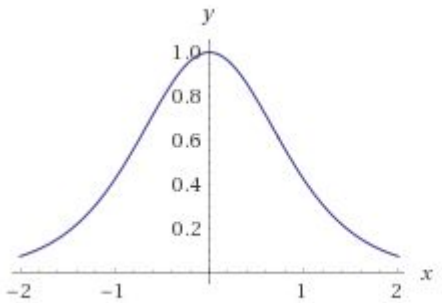
# FeedFoward

-”Vanish Gradient”



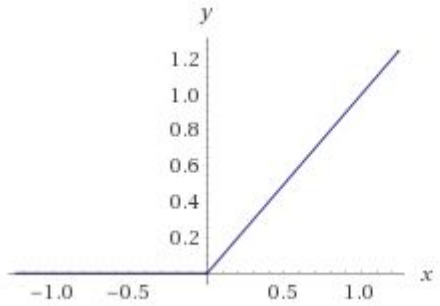

tanh (x)

(x from −2 to 2)




tanh’ (x)

(x from −2 to 2)




relu (x)

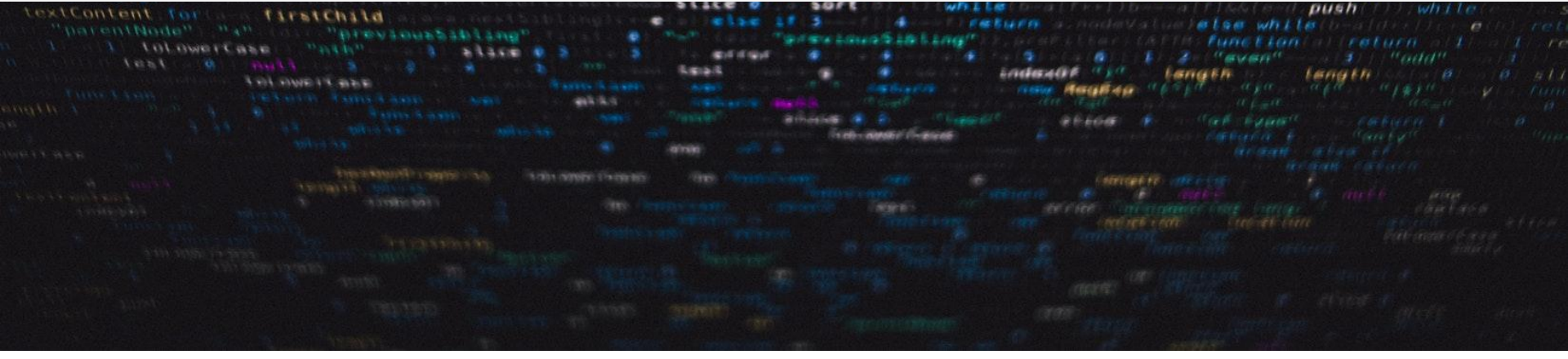
(x from −1.2 to 1.2)

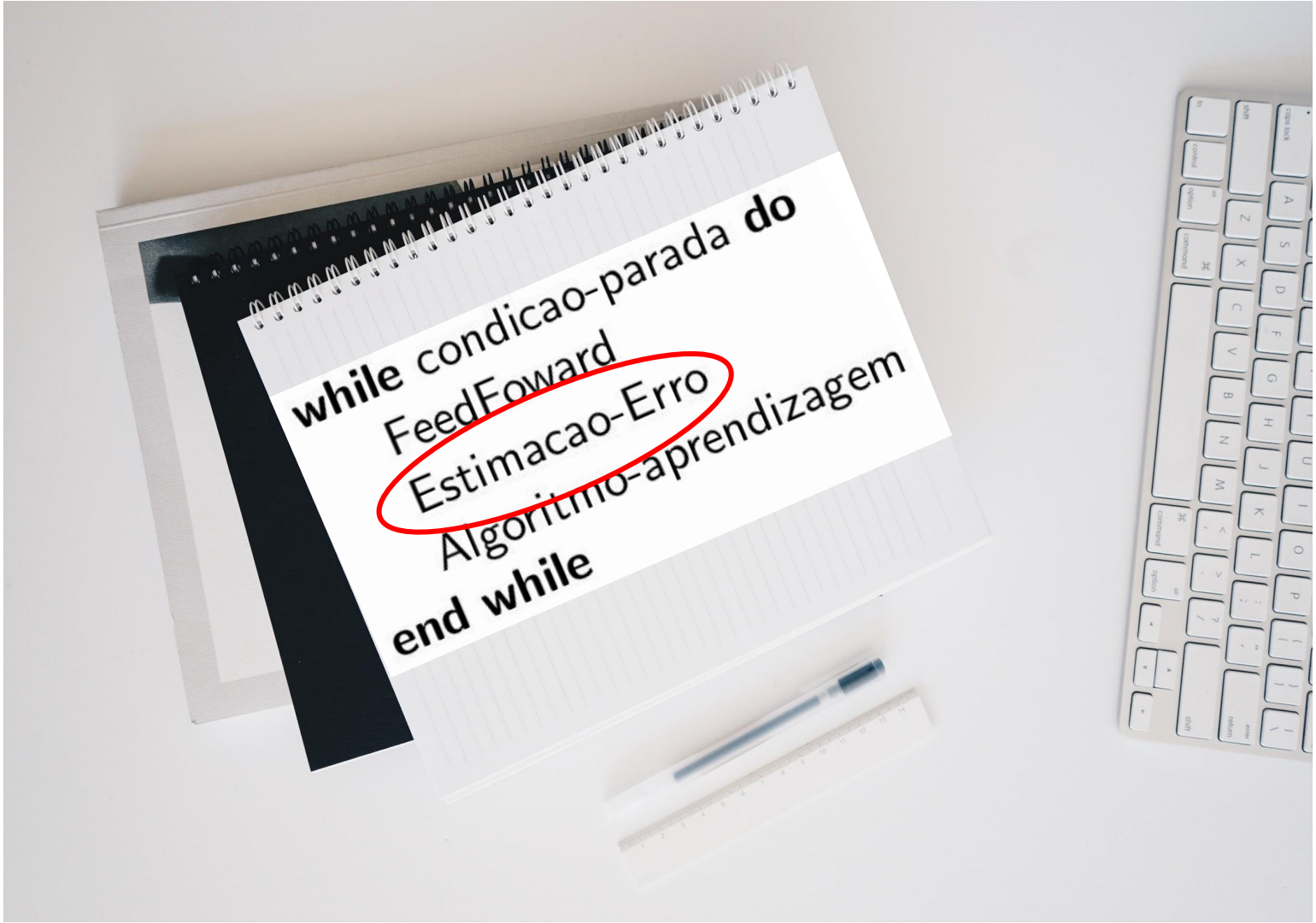
relu’ (x)

$$\frac{d}{dx}\left(f(x) = \max(0, x)\right) = \begin{cases} 0 & x < 0 \\ 1 & x > 0 \\ \text{indeterminate} & \text{(otherwise)} \end{cases}$$

# FeedFoward

```
MLPClassifier(activation='relu', alpha=1e-05, batch_size='auto',
              beta_1=0.9, beta_2=0.999, early_stopping=False,
              epsilon=1e-08, hidden_layer_sizes=(100,) ,
              learning_rate='constant', learning_rate_init=0.001,
              max_iter=200, momentum=0.9, n_iter_no_change=10,
              nesterovs_momentum=True, power_t=0.5, random_state=1,
              shuffle=True, solver='lbfgs', tol=0.0001,
              validation_fraction=0.1, verbose=False, warm_start=False)
```

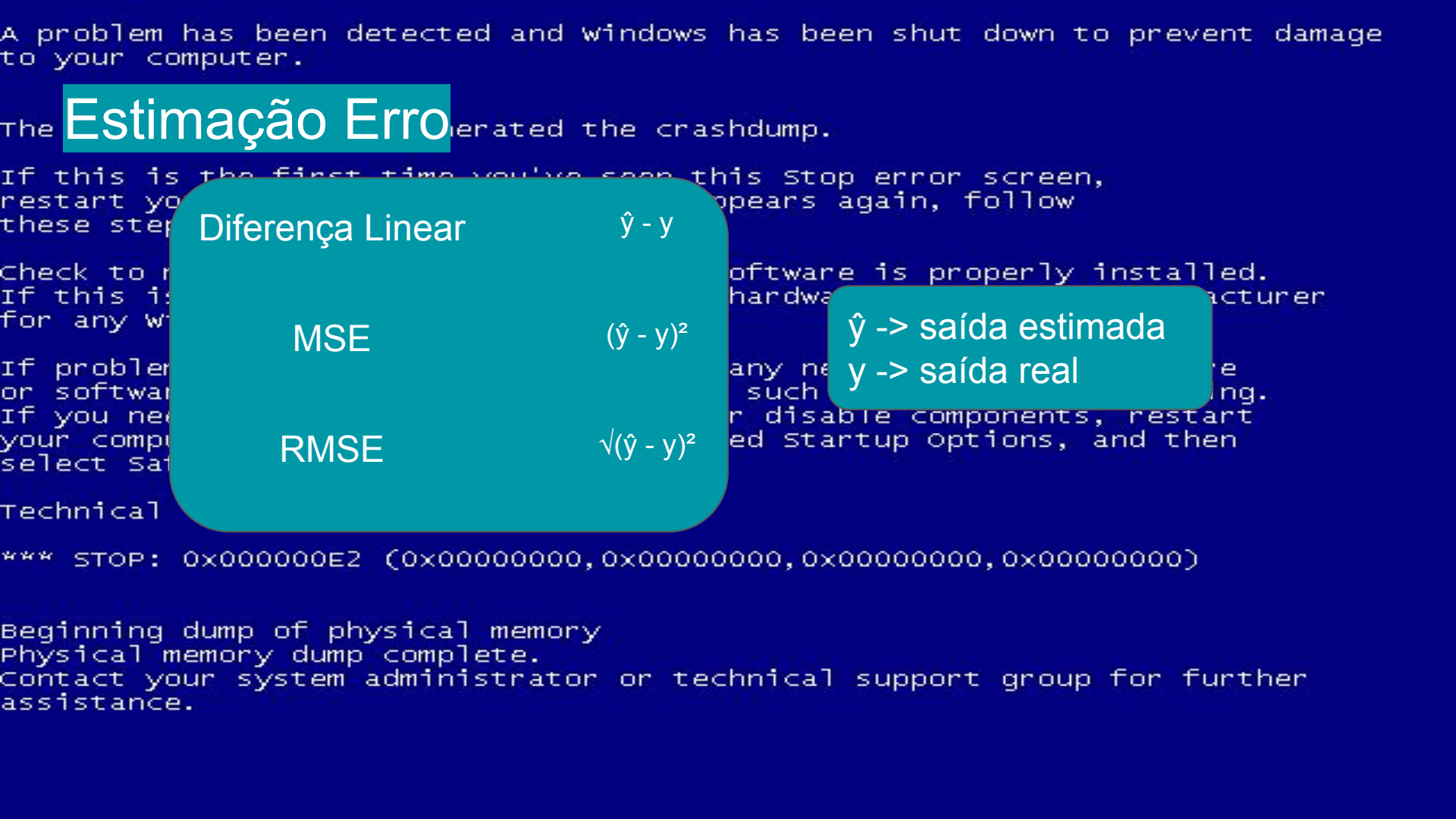



while condicao-parada do
FeedFoward
Estimacao-Erro
Algoritmo-aprendizagem
end while

# Estimação Erro

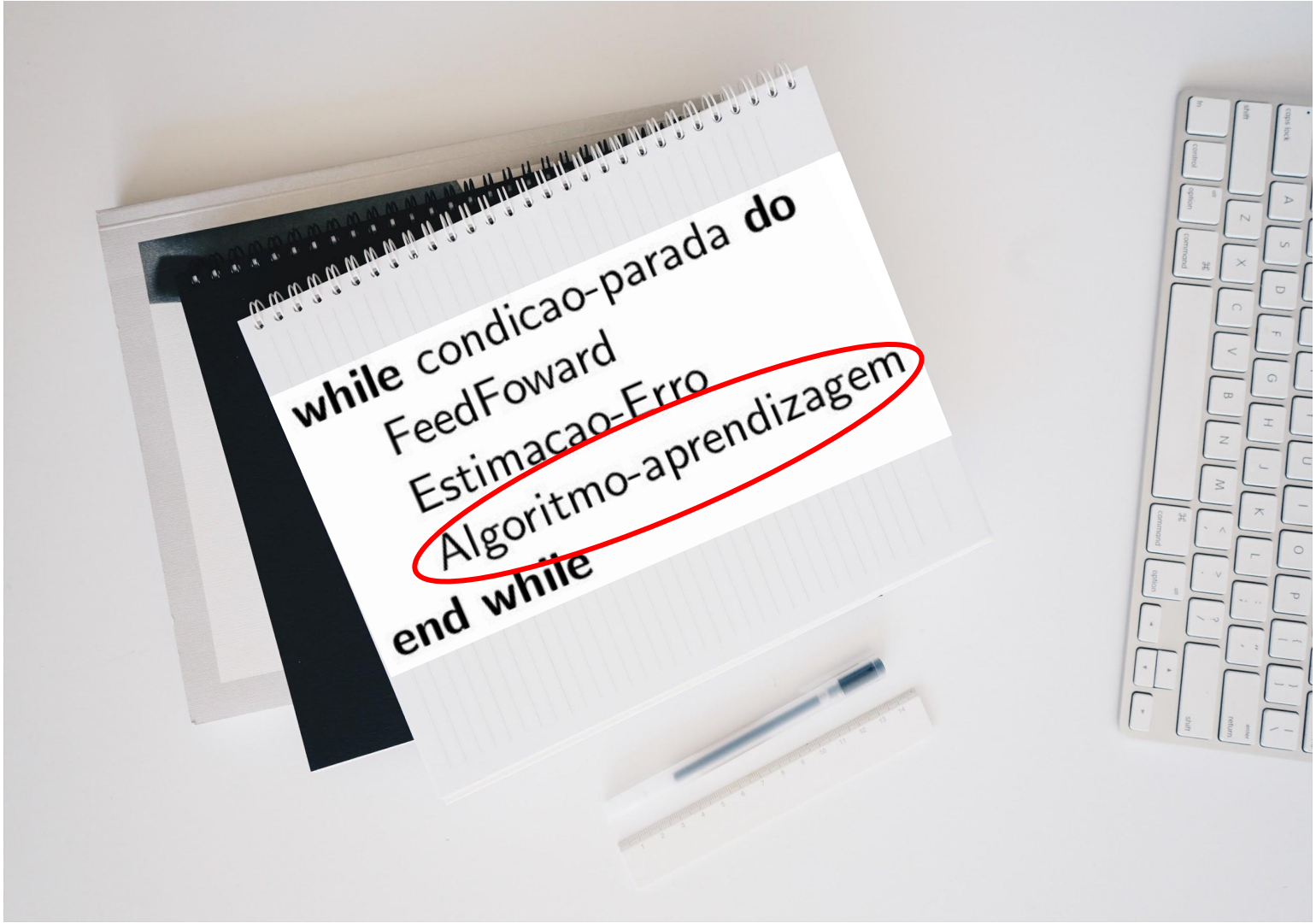
| | |
|---|---|
| Diferença Linear | $\hat{y} - y$ |
| MSE | $(\hat{y} - y)^2$ |
| RMSE | $\sqrt{(\hat{y} - y)^2}$ |

$\hat{y}$ -> saída estimada
y -> saída real

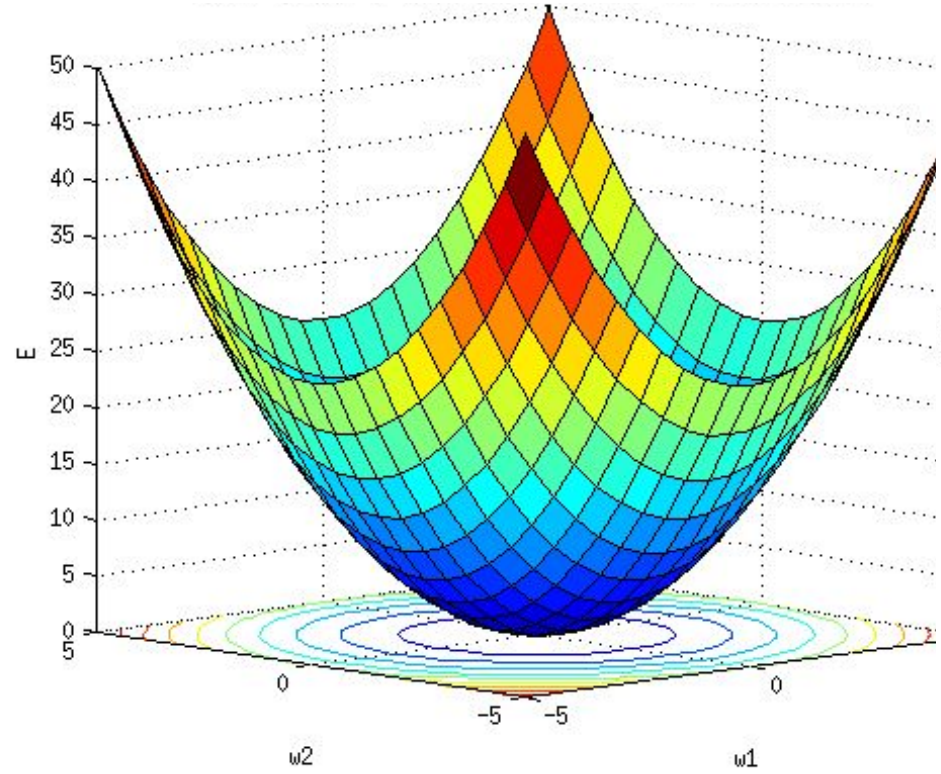


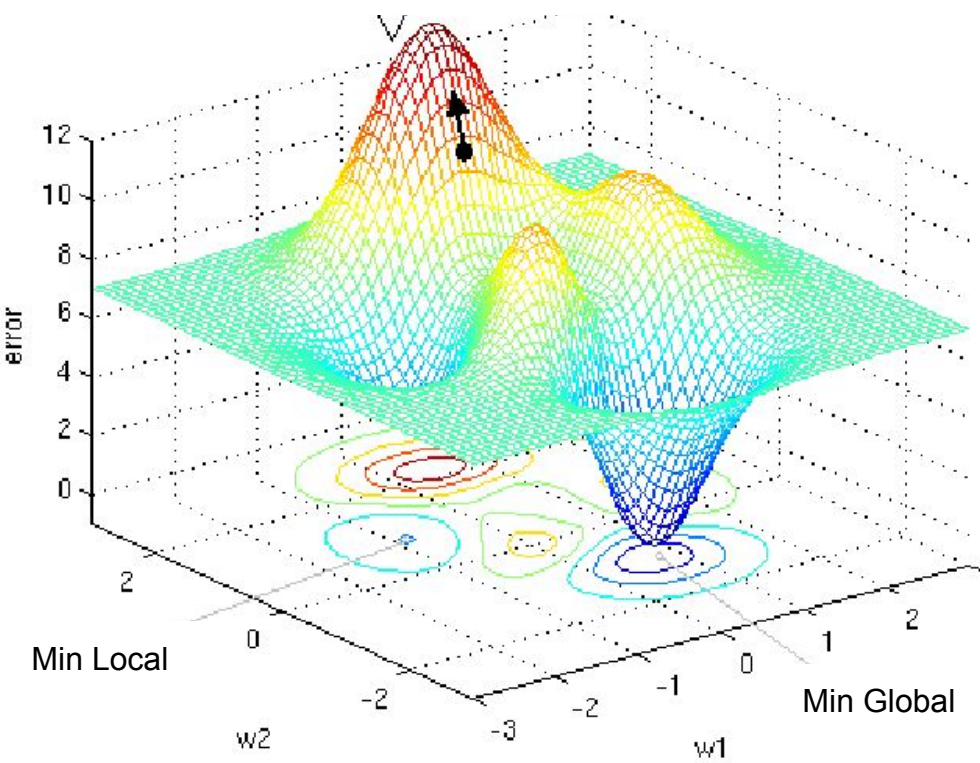
# Escolhendo a equação certa de erro

| estimado | real | linear | mse | rmse |
|---|---|---|---|---|
| $\hat{y}$ | y | $\hat{y}$ - y | $(\hat{y} - y)^2$ | $\sqrt{(\hat{y} - y)^2}$ |
| 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | -1 | 1 | 1 |
| -1 | 1 | -2 | 4 | 2 |


while condicao-parada do
FeedFoward
Estimacao-Erro
Algoritmo-aprendizagem
end while

# Superfície de Erro e algoritmos de otimização



# Algoritmo de otimização / aprendizado

```
MLPClassifier(activation='relu', alpha=1e-05, batch_size='auto',
              beta_1=0.9, beta_2=0.999, early_stopping=False,
              epsilon=1e-08, hidden_layer_sizes=(100,) ,
              learning_rate='constant', learning_rate_init=0.001,
              max_iter=200, momentum=0.9, n_iter_no_change=10,
              nesterovs_momentum=True, power_t=0.5, random_state=1,
              shuffle=True, solver='lbfgs', tol=0.0001,
              validation_fraction=0.1, verbose=False, warm_start=False)
```

Solvers e seus parâmetros de configuração:

***lbfgs -> --***

***sgd ->*** *batch_size,* learning_rate, learning_rate_init, power_t, max_iter, shuffle, momentum, nesterovs_momentum, early_stopping, n_iter_no_change

***adam ->*** *batch_size,* learning_rate_init, max_iter, shuffle, momentum, early_stopping, beta_1, beta_2, epsilon, n_iter_no_change

# Exemplos

```
mlp1 = MLPClassifier(hidden_layer_size
                     early_stopping=Tr
```

MLP 1: relu e adam

| | Média Treino | Var Treino |
|---|---|---|
| Iris | 0.685079 | 0.0481284 |
| Wine | 0.44543 | 0.0139164 |

```
mlp2 = MLPClassifier(hidden_layer_siz
                     early_stopping=T
```

MLP 2: tanh e sgd

| | Média Treino | Var Treino |
|---|---|---|
| Iris | 0.731111 | 0.0369485 |
| Wine | 0.45 | 0.0239262 |

## **CONCLUSÕES**:

- Ambos melhores com tanh e sgd
  ➔ De fato, ambos não são tão complexos
- MLP para dataset wine não está adequado!


DÚVIDAS ?

# Links úteis:

**Palestra**:
Git: https://github.com/JulianaGuama/palestra_parametros_rna.git
Colab:
https://colab.research.google.com/drive/1N0CdTD9bYtbHtZbCTW-0mT9luQqIFK5C

**Pessoais**:
Git: https://github.com/JulianaGuama
Linkedin: https://br.linkedin.com/in/juliana-guama

# Créditos:

Neural Networks supervisionadas:
https://scikit-learn.org/stable/modules/neural_networks_supervised.html

Documentação do MLP scikit-learning:
https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

Crédito imagens:
https://pixabay.com