Funções de normalização

É muito importante para os algoritmos de machine learning e deep learning, que os dados coletados estejam na mesma escala, evitando que algumas features se sobressaiam a outras e também fazendo com que o modelo performe melhor. Nem sempre é necessário realizar a normalização, mas sim quando os parâmetros tiverem intervalos muito diferentes, como por exemplo duas variáveis idade e renda, a faixa etária varia de 0 a 100 anos, já a renda de 0.00 até 20.000,00 ou mais, ou seja, a renda é cerca de 1000 vezes maior que a idade, podendo influenciar mais do que o necessário no resultado final, mesmo que não seja tão importante ao modelo.

MinMaxScaler

Essa função padroniza os dados entre dois parâmetros estipulados (0, 1 por exemplo), da seguinte forma:

```
# Fórmula utilizada para normalização MinMaxScaler
X_std = (X - Xmin) / (Xmax - Xmin)
X_scaled = X_std * (máx - min) + min

from sklearn.preprocessing import MinMaxScaler

X = [[4, 1, 2, 2],[1, 3, 9, 3],[5, 7, 5, 1]]
normalizador = MinMaxScaler(feature_range = (0 , 1))
print(normalizador.fit_transform(X))
[[4, 1, 2, 2], [1, 3, 9, 3], [5, 7, 5, 1]]
[[0.75 0. 0. 0.5 ]
[0. 0.33333333 1. 1. ]
[1. 1. 0.42857143 0. ]]
```

StandardScaler

Normaliza os dados a partir da fórmula gaussiana:

z = (x - u) / s

Onde u é a média e “s” é o desvio padrão (standard deviation).

```
from sklearn.preprocessing import StandardScaler

X = [[4, 1, 2, 2],[1, 3, 9, 3],[5, 7, 5, 1]]
```

```
normalizador = StandardScaler()
print(normalizador.fit_transform(X))
[[ 0.39223227 -1.06904497 -1.16247639 0. ]
[-1.37281295 -0.26726124 1.27872403 1.22474487]
[ 0.98058068 1.33630621 -0.11624764 -1.22474487]]
```

MaxAbsScaler

Normaliza os dados dividindo cada elemento pelo maior valor do conjunto:

X' = X/M (onde M é o valor máximo)

```
from sklearn.preprocessing import MaxAbsScaler

X = [[4, 1, 2, 2],[1, 3, 9, 3],[5, 7, 5, 1]]
normalizador = MaxAbsScaler()
print(normalizador.fit_transform(X))
[[0.8 0.14285714 0.22222222 0.66666667]
[0.2 0.42857143 1. 1. ]
[1. 1. 0.55555556 0.33333333]]
```

Função Normalize:

Realiza a normalização de cada linha da matriz (o cálculo é feito linha por linha em vez de coluna por coluna). Possui 3 parâmetros possíveis: **'l1'**, 'l2' **ou 'max'.

- L1: $z = \|x\|_1 = \sum_{i=1}^{n} |x_i|$
- L2: $z = \|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$
- Max: $z = \max x_i$

Exemplos:

Obs: Os valores são aplicados por linhas, é necessário fornecer o eixo que deseja realizar a normalização, se é por coluna ou por linhas com o parâmetro axis.

```
from sklearn.preprocessing import Normalize

X = [[4, 1, 2, 2],[1, 3, 9, 3],[5, 7, 5, 1]]

normalizador = Normalize(X, norm = 'l1')
print(normalizador)
```

```
[[0.44444444 0.11111111 0.22222222 0.22222222]
[0.0625 0.1875 0.5625 0.1875 ]
[0.27777778 0.38888889 0.27777778 0.05555556]]

X = [[4, 1, 2, 2],[1, 3, 9, 3],[5, 7, 5, 1]]
normalizador = Normalize(X, norm = 'l2')

print(normalizador)
[[0.8 0.2 0.4 0.4]
[0.1 0.3 0.9 0.3]
[0.5 0.7 0.5 0.1]]

X = [[4, 1, 2, 2],[1, 3, 9, 3],[5, 7, 5, 1]]
normalizador = Normalize(X, norm = 'max')

print(normalizador)
[[1. 0.25 0.5 0.5 ]
[0.11111111 0.33333333 1. 0.33333333]
[0.71428571 1. 0.71428571 0.14285714]]
```