Perceptrón multicapa regresión

Para esta implementación vamos a utilizar la función

```
sklearn.neural_network.MLPRegressor(
hidden_layer_sizes=100, activation='relu', *, solver='adam',
alpha=0.0001, batch_size='auto', learning_rate='constant',
learning_rate_init=0.001, power_t=0.5, max_iter=200,
shuffle=True, random_state=None, tol=0.0001,
verbose=False, warm_start=False, momentum=0.9,
nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1,
beta_1=0.9, beta_2=0.999, epsilon=1e-08,
n_iter_no_change=10, max_fun=15000)
```

de la biblioteca de `sklearn`

https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html

(Añadir enlace a la bibliografía más adelantes)

Además utilizaremos los siguientes argumentos:

- `hidden_layer_sizes` número de unidades por capa en el rango 50-100, que afinaremos por validación cruzada.
- `activation`: `logistic` la función de activación logística NO TENGO ARGUMENTO PARA ELEGIR ESTA U OTRA.
- `solver` la técnica para minimizar `adam` ya que según la documentación este método es el que funciona mejor con miles datos como es nuestro caso.
- `alpha` método de regularización.
- `learning_rate: {'constant', 'invscaling', 'adaptative'}`.
- `learning_rate_init` aquí si hay que utilizarl

Explicación del método de minimización de `adam`

Bibliografía: Kingma, Diederik, and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

Es un método basado en la optimización de gradiente descendiente. Requiere de gradeinte de primer orden.

Las ventajas que supone son

`Our method is designed to combine the advantagesof two recently popular methods: AdaGrad (Du`

TODO : redactar mejor

Además com heurística en la propia documentación del sklearn se recomendaba para tamños de entrenamiento de miles.

El algoritmo indicado en el artículo de 2015 donde se publicó es el siguiente:

Algorithm 1: *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  while θ_t not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  end while
  return θ_t (Resulting parameters)

Figura 1: Descripción del algoritmo de minimización estocástico de Adam

.