

Introdução às Redes Neurais Artificiais

Scikit Learn

Prof. João Marcos Meirelles da Silva
jmarcos@id.uff.br

Universidade Federal Fluminense

www.latelco.uff.br

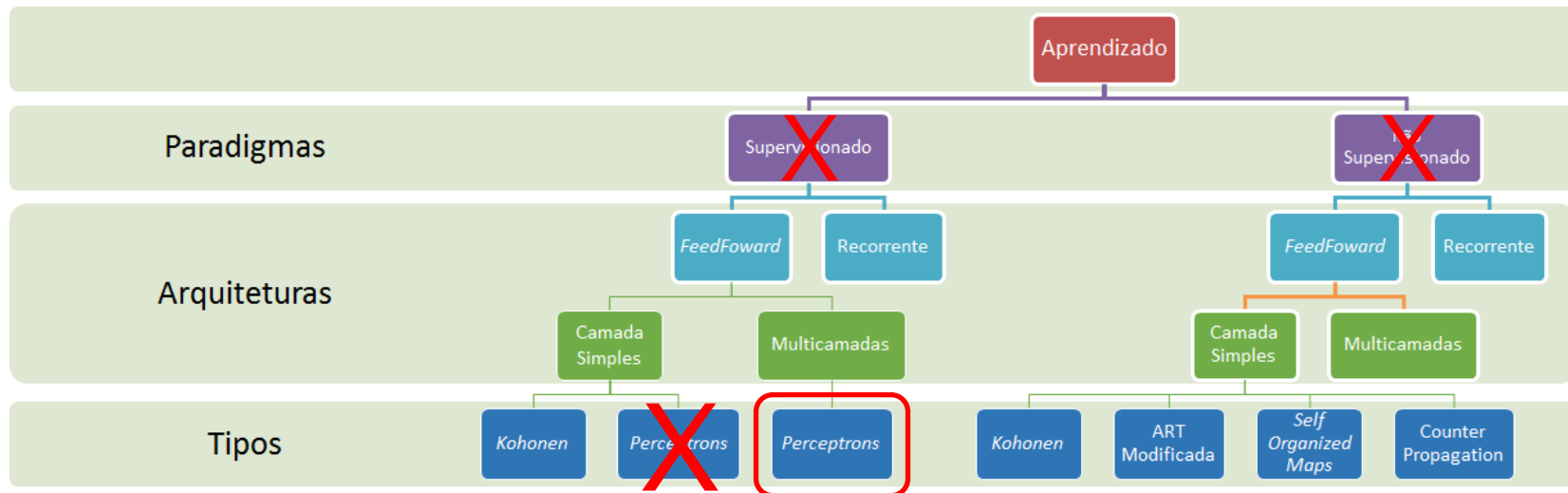
www.professores.uff.br/jmarcos

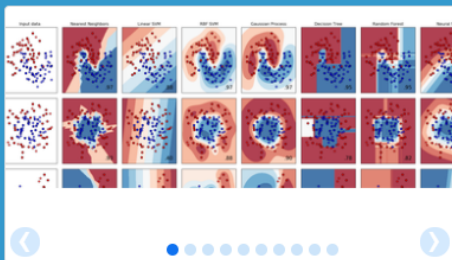
Mapa de Aprendizado

Redução de Dimensionalidade

Tipos de Dados

Deteccção de Anomalias





scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Mapa de Aprendizaje

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.

Algorithms: SVM, nearest neighbors, random forest, ... — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: SVR, ridge regression, Lasso, ... — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift, ... — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: grid search, cross validation, metrics. — Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: preprocessing, feature extraction. — Examples

Classificador MLP

Classe `sklearn.neural_network.MLPClassifier`

Implementa uma rede Perceptron Multicamadas usando o algoritmo de treinamento *Backpropagation*.

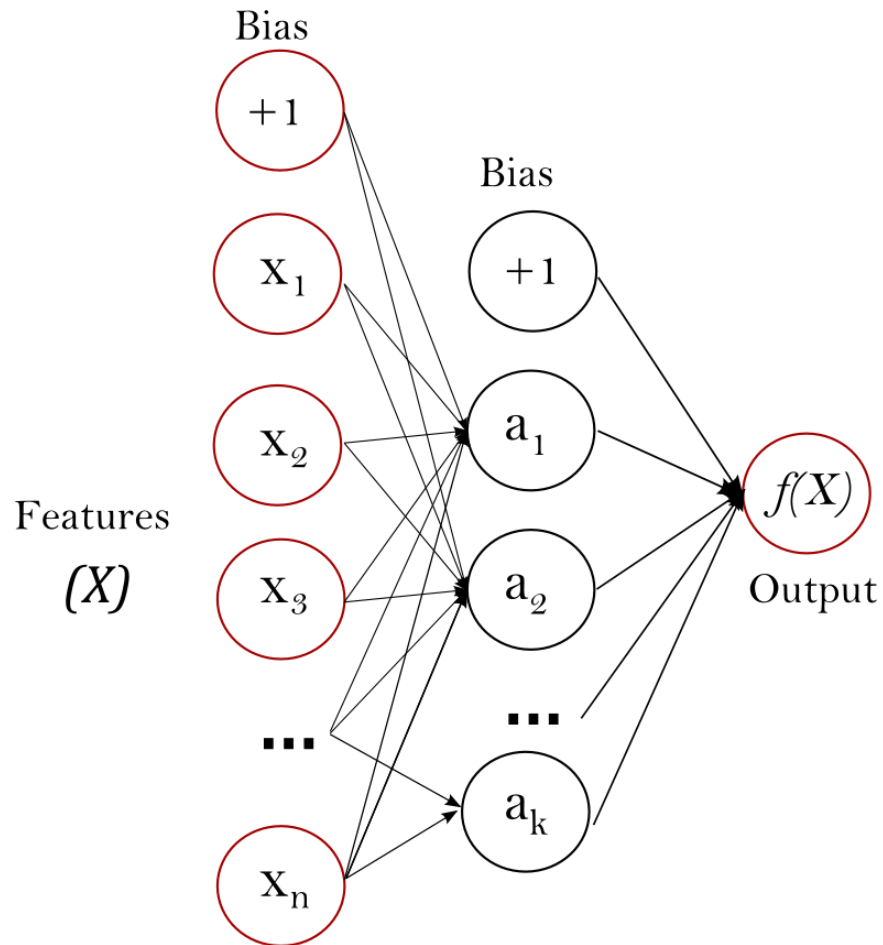
Entrada: Array $X_{(n_amostras, n_atributos)}$

Saída: Array $Y_{(n_amostras)} \rightarrow$ Rótulos de classes

Classificador MLP

Atributos públicos:

- **coefs_**: Lista de matrizes de pesos, onde a i -ésima matriz de pesos representa os pesos entre a camada i e $i+1$;
- **intercepts_**: Lista dos vetores de bias, onde o i -ésimo vetor representa o bias adicionado à camada $i+1$.



Classificador MLP

```
>>> from sklearn.neural_network import MLPClassifier
>>> X = [[0., 0.], [1., 1.]]
>>> y = [0, 1]
>>> clf = MLPClassifier(solver='lbfgs', alpha=1e-5,
...                      hidden_layer_sizes=(5, 2), random_state=1)
...
>>> clf.fit(X, y)
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=(5, 2), learning_rate='constant',
              learning_rate_init=0.001, max_iter=200, momentum=0.9,
              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)
```

Classificador MLP

Após o comando `clf.fit(X,y)`, a rede está pronta para classificar novas entradas:

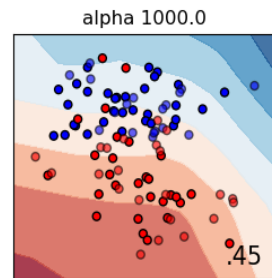
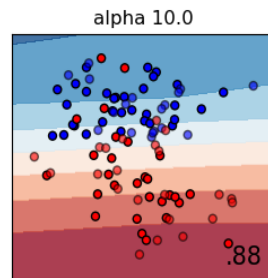
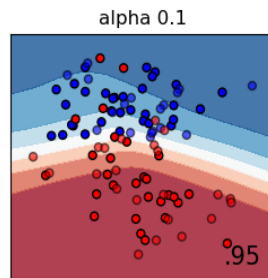
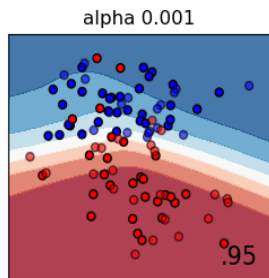
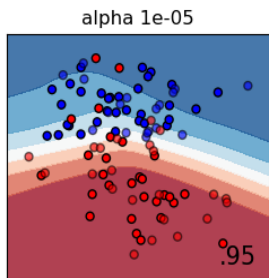
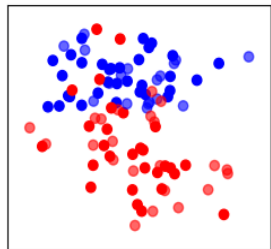
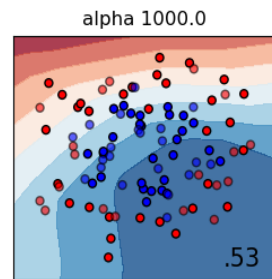
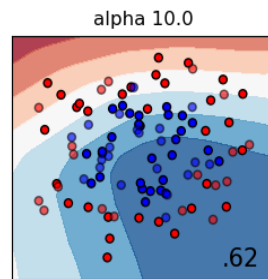
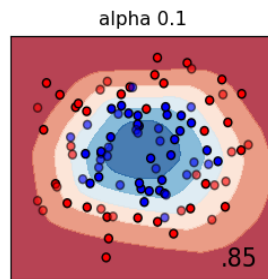
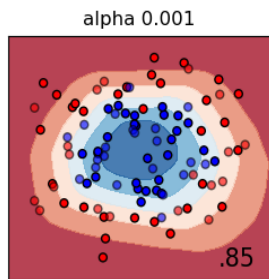
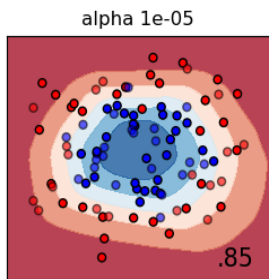
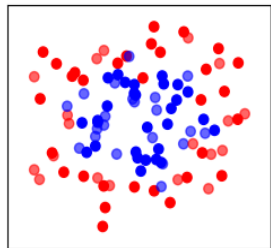
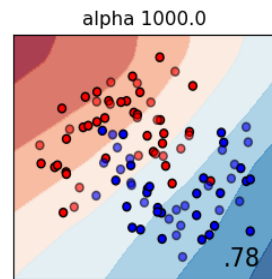
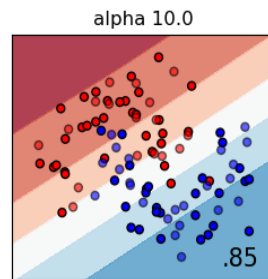
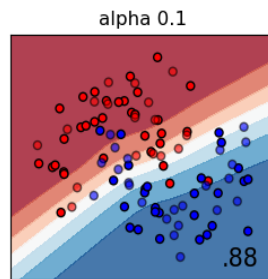
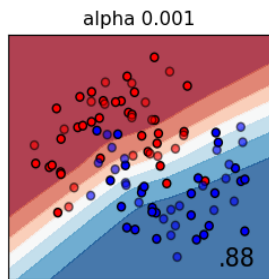
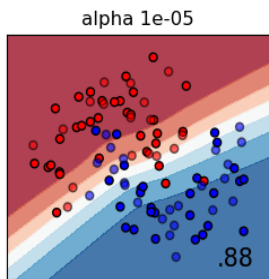
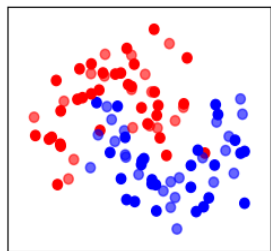
```
>>> clf.predict([[2., 2.], [-1., -2.]])  
array([1, 0])
```

O atributo `clf.coefs_` contém as matrizes de pesos:

```
>>> [coef.shape for coef in clf.coefs_]  
[(2, 5), (5, 2), (2, 1)]
```

Regularização

A classe `MLPClassifier` usa o parâmetro α para a regularização, o que ajuda a evitar o problema de *overfitting* através da penalização dos pesos com grandes magnitudes.



Algoritmos

A classe `MLPClassifier` usa os seguintes algoritmos para treinamento:

1. *SGD (Stochastic Gradient Descent)* → online e mini-batch
2. *Adam (Adaptive Moment Estimation)* → online e mini-batch (melhor que SGD)
3. *L-BFGS*

Dicas

1. MLP é altamente sensível à escala dos atributos. Normalize cada um dos atributos do vetor de entrada para $[0,1]$ ou $[-1, +1]$;
2. Padronize seus dados de modo que a média seja 0 e variância unitária;
3. Use o método *StandardScaler* (geralmente na escala $10.0**np.arange(1,7)$);
4. Tente encontrar um bom valor para o parâmetro de regularização α (método *GridSearchCV*).
5. Use o método:
L-BFGS para pequenas bases de dados
ADAMS para grandes bases de dados

Dicas

```
>>> from sklearn.preprocessing import StandardScaler
>>> scaler = StandardScaler()
>>> # Don't cheat - fit only on training data
>>> scaler.fit(X_train)
>>> X_train = scaler.transform(X_train)
>>> # apply same transformation to test data
>>> X_test = scaler.transform(X_test)
```

PRÁTICA

1) Visitar o site:

www.scikit-learn.org

2) Clicar em "Classification"

3) Clicar no item "1.17 - Neural Network Models (Supervised)"

4) Leia a página e, por fim, clique em "Visualization of MLP weights on MNIST"

5) Leia a página e execute o código exemplo disponível nela.

PRÁTICA

1. Visitar o site
2. Leia sobre a base de dados íris dataset

sklearn.neural_network: Visualization of MLP we: UCI Machine Learning Re: <https://archive.ics.uci.edu/ml/index.php>

UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

About Citation Policy Donate a Data Set Contact

Repository Web

View ALL Data Sets

Welcome to the UC Irvine Machine Learning Repository!


We currently maintain 426 data sets as a service to the machine learning community. You may [view all data sets](#) through our searchable interface. Our [old web site](#) is still available, for those who prefer the old format. For a general overview of the Repository, please visit our [About page](#). For information about citing data sets in publications, please read our [citation policy](#). If you wish to donate a data set, please consult our [donation policy](#). For any other questions, feel free to [contact the Repository librarians](#). We have also set up a [mirror site](#) for the Repository.

Supported By: In Collaboration With: [Rexa.info](#)
Research • People • Connections

Latest News:

- 04-04-2013: Welcome to the new Repository admins Kevin Bache and Moshe Lichman!
- 03-01-2010: [Note](#) from donor regarding Netflix data
- 10-16-2009: Two new data sets have been added.
- 09-14-2009: Several data sets have been added.
- 07-23-2008: [Repository mirror](#) has been set up.
- 03-24-2008: New data sets have been added!
- 06-25-2007: Two new data sets have been added: UJI Pen Characters, MAGIC Gamma Telescope

Featured Data Set: [Coil 1999 Competition Data](#)


Data Type: Multivariate
Attributes: 17
Instances: 340

This data set is from the 1999 Computational Intelligence and Learning (COIL) competition. The data contains measurements of river chemical concentrations and class description.

Newest Data Sets:

- 03-22-2018: [UCI Repeat Consumption Matrices](#)
- 02-27-2018: [UCI SGEMM GPU kernel performance](#)
- 02-21-2018: [UCI chipsaq](#)
- 02-20-2018: [UCI News Popularity in Multiple Social Media Platforms](#)
- 02-19-2018: [UCI Residential Building Data Set](#)
- 02-19-2018: [UCI ICMLA 2014 Accepted Papers Data Set](#)
- 02-19-2018: [UCI Health News in Twitter](#)
- 02-19-2018: [UCI RLE RSSI Dataset for Indoor localization and](#)

Most Popular Data Sets (hits since 2007):

- 1796097: [Iris](#)
- 1128960: [Adult](#)
- 859211: [Wine](#)
- 739890: [Car Evaluation](#)
- 671445: [Breast Cancer Wisconsin \(Diagnostic\)](#)
- 624420: [Heart Disease](#)
- 620641: [UCI Human Activity Recognition Using Smartphones](#)

plot_mlp_alpha.py plot_mnist_filters.py anatomia do siste....pdf

Exibir todos

PRÁTICA

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  2 07:49:10 2018
4
5 @author: jmarcos
6 """
7
8 from matplotlib import pyplot as plt
9 from sklearn.datasets import load_iris
10 import numpy as np
11
12 # We load the data with load_iris from sklearn
13 data = load_iris()
14 features = data['data']
15 feature_names = data['feature_names']
16 target = data['target']
17 for t,marker,c in zip(range(3),">ox","rgb"):
18     # We plot each class on its own to get different colored markers
19     plt.scatter(features[target == t,0],
20                 features[target == t,1],
21                 marker=marker, c=c)
```

Crie um classificador usando a classe `MLPClassifier` para a base da Iris!

Referências

1. *Neural Networks and Learning Machines*, 3rd. Edition, Simon Haykin
2. *Fundamental of Neural Networks - Architectures, Algorithms and Applications*, Laurene Fausett
3. *Pattern Classification*, Richard O. Duda, Peter E. Hart, David G. Stork