Unyleya
EDUCACIONAL

A biblioteca Scikit-learn

A biblioteca Scikit-learn

Scikit-learn é uma biblioteca da linguagem Python criada para facilitar o trabalho com recursos para o aprendizado de máquina. Dentro dessa biblioteca estão várias implementações de algoritmos e técnicas para analisar dados.

Utilizando Scikit-learn

A biblioteca Scikit-learn precisa ser previamente instalada, então é possível utilizá-la para construção de aplicações, começando importando:

```
import sklearn
```

Utilizando Scikit-learn

Dentro da própria biblioteca existem alguns datasets que podem ser utilizados para criação de aplicações variadas. A importação é feita do seguinte modo:

```
from sklearn.datasets import load_breast_cancer
#Importando dataset
data = load_breast_cancer()
```

Utilizando Scikit-learn

Método para carregar	Descrição	Tamanho
load_boston()	Preços de casa em Boston	506
load_breast_cancer()	Câncer de mama Wisconsin	569
load_diabetes()	Diabetes	442
load_digits(n_class)	Dígitos	1797
load_iris()	Iris	150

Utilizando Scikit-learn

Se for necessário é possível separar os dados do dataset em variáveis diferentes e específicas.

```
nomesRotulos = data['target_names']
rotulos= data['target']
nomesAtributos = data['feature_names']
atributos = data['data']
```

Utilizando Scikit-learn

Em muitas situações é interessante dividir o dataset em duas partes diferentes: uma para treinamento e outra para testes. A biblioteca Scikit-learn possui uma função específica para isso, chamada de *train_test_split()*.

```
from sklearn.model_selection import train_test_split
```

Utilizando Scikit-learn

No parâmetro test_size é possível determinar o tamanho do conjunto de teste, ficando o de treinamento com o restante dos dados. No exemplo a seguir, o conjunto de teste terá 30% dos dados do dataset.

```
#Seperando os dados
treinamento, teste, rotulosTreinamento, rotulosTeste = train_test_split(atributos, rotulos, test_size=0.30, random_state=42)
```

Criando classificador Naive Bayes

É necessário importar o algoritmo GaussianNB, inicializá-lo e depois treiná-los com os dados que foram separados para a etapa de treinamento.

```
from sklearn.naive_bayes import GaussianNB

#Inicializando o classificador
gnb = GaussianNB()

#Treinando o classificador
modelo = gnb.fit(treinamento, rotulosTreinamento)
```

Criando classificador Naive Bayes

Depois de treinado, o modelo pode ser utilizado para predição de resultado, então, pode-se utilizar os dados que ficaram reservados para a realização de testes. Para isso utiliza-se a função *predict* com os dados de teste.

```
resultados = gnb.predict(teste)
print(resultados)
```

Avaliando o modelo

O modelo poder ser avaliado para verificar como foi o seu desempenho, utilizando a função *accuracy_score*.

```
from sklearn.metrics import accuracy_score

# Avaliar a precisão
print(accuracy_score(rotulosTeste, resultados))
```

Obrigada!

Ana Laurentino