AtividadeMNIST-DatasetDecisionTree

December 8, 2020

1 Lendo dados do MNIST

[1]:
```
from sklearn.datasets import fetch_openml
mnist = fetch_openml('mnist_784', version=1)
mnist.keys()
```

[1]:
```
dict_keys(['data', 'target', 'frame', 'categories', 'feature_names',
'target_names', 'DESCR', 'details', 'url'])
```

[2]:
```
mnist['data']
```

[2]:
```
array([[0., 0., 0., …, 0., 0., 0.],
       [0., 0., 0., …, 0., 0., 0.],
       [0., 0., 0., …, 0., 0., 0.],
       …,
       [0., 0., 0., …, 0., 0., 0.],
       [0., 0., 0., …, 0., 0., 0.],
       [0., 0., 0., …, 0., 0., 0.]])
```

2 Conjunto de dados

[3]:
```
X, y = mnist["data"], mnist["target"]
X.shape, y.shape
```

[3]:
```
((70000, 784), (70000,))
```

3 Observando um caractere do MNIST

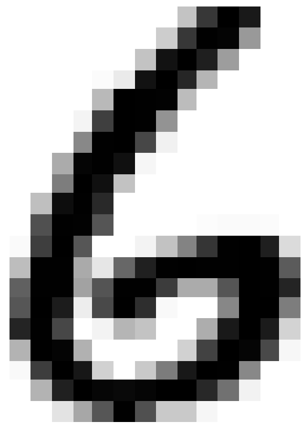
[4]:
```
import matplotlib as mpl
import matplotlib.pyplot as plt

some_digit = X[6000]
some_digit_image = some_digit.reshape(28, 28)

plt.imshow(some_digit_image, cmap = mpl.cm.binary, interpolation="nearest")
plt.axis("off")
```

```
plt.show()

y[6000]
```



[4]: '6'

[5]:
```
some_digit.reshape(28, 28)
```

[5]:
```
array([[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.],
       [  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.],
       [  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.],
       [  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,  59., 200., 255., 229.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.],
       [  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,  50., 200., 253., 251., 102.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.],
       [  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
          0.,   0.,   0.,  59., 236., 254., 224.,  98.,   0.,   0.,   0.,
          0.,   0.,   0.,   0.,   0.,   0.],
       [  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
```

```
   0.,   4.,  21., 236., 254., 217.,  74.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,  70., 254., 253., 252.,  67.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
  11., 192., 254., 253., 108.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
 119., 253., 254., 184.,  14.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  84.,
 250., 253., 239.,   9.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., 130.,
 254., 239.,  61.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  64., 242.,
 253., 213.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., 175., 253.,
 252., 170.,   0.,   0.,   0.,   0.,   3.,   5.,   5.,   4.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   5., 193., 254.,
 169.,   0.,   0.,  12.,  61., 123., 202., 253., 254., 224.,  36.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  66., 253., 253.,
  85.,  25., 152., 222., 253., 254., 253., 253., 254., 253., 164.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., 159., 253., 253.,
  93., 171., 253., 253., 195.,  84.,  84., 170., 254., 253., 219.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., 169., 254., 227.,
  18., 182., 254., 209.,   6.,   0.,   0., 120., 254., 254., 130.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., 219., 253., 184.,
   0.,  12.,  74.,  60.,   0.,   0.,  82., 229., 254., 228.,  43.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  75., 254., 251.,
  78.,   0.,   0.,   0.,   0.,  46., 188., 254., 250., 181.,   8.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   6., 194., 254.,
 232.,  46.,   0.,  54., 131., 230., 254., 253., 119.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  76., 222.,
 254., 249., 245., 250., 254., 254., 215., 142.,  13.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
```

```
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  28.,
  88., 169., 253., 175.,  54.,  54.,   9.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.],
[  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
   0.,   0.,   0.,   0.,   0.,   0.]])
```

4 Divida o dataset entre conjunto de treinamento e teste

Utilize 33% para o conjunto de teste

[6]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.33)
```

5 Construa uma arvore de decisão e verifique a acurácia do modelo.

Utilize o algoritmos tradicional do `sklearn`

[7]:
```
from sklearn.tree import DecisionTreeClassifier
```

[8]:
```
decTree = DecisionTreeClassifier(random_state = 0)
decTree.fit(X_train, y_train)
```

[8]:
```
DecisionTreeClassifier(random_state=0)
```

[9]:
```
decTree.score(X_test, y_test)
```

[9]:
```
0.8672727272727273
```

6 Compare a acurácia do modelo de árvore de decisão e um modelo de regressão logística (sklearn)

[10]:
```
from sklearn.linear_model import LogisticRegression
```

[13]:
```
# LogReg = LogisticRegression()
# LogReg.fit(X_train, y_train)
```

[12]:
```
LogReg.score(X_test, y_test)
```

[12]:
```
0.9192640692640692
```

7 Desafio

Implemente o algoritmo CART utilizando apenas o numpy e pandas e aplique no dataset MNIST

[ ]:
```

```