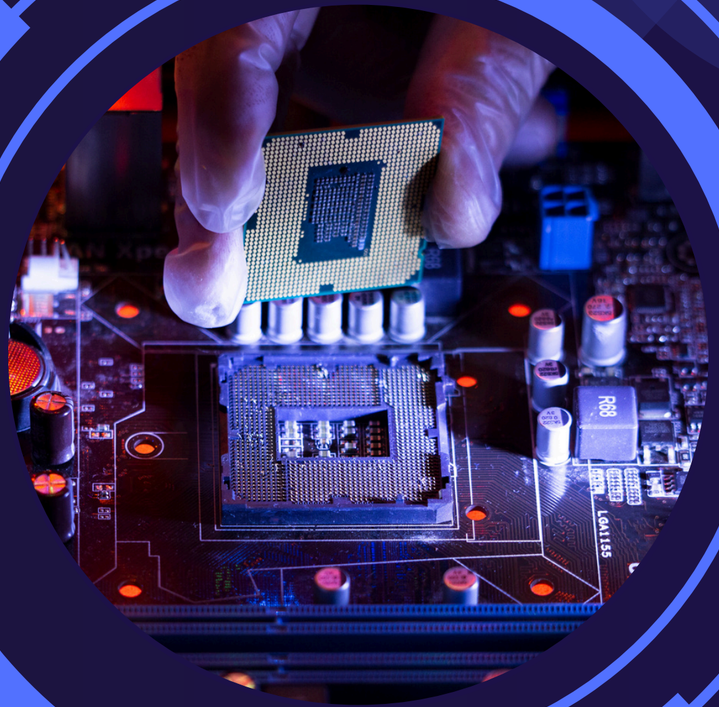
Machine Learning:

Classificação de dígitos manuscritos



Giovane Cirne

Objetivo: classificar dígitos manuscritos (0 a 9) a partir de imagens digitalizadas.

IN

Vetores de características (features) de imagens de dígitos, onde cada imagem 8×8 pixels é transformada em um vetor de 64 dimensões, com cada valor representando a intensidade de um pixel.

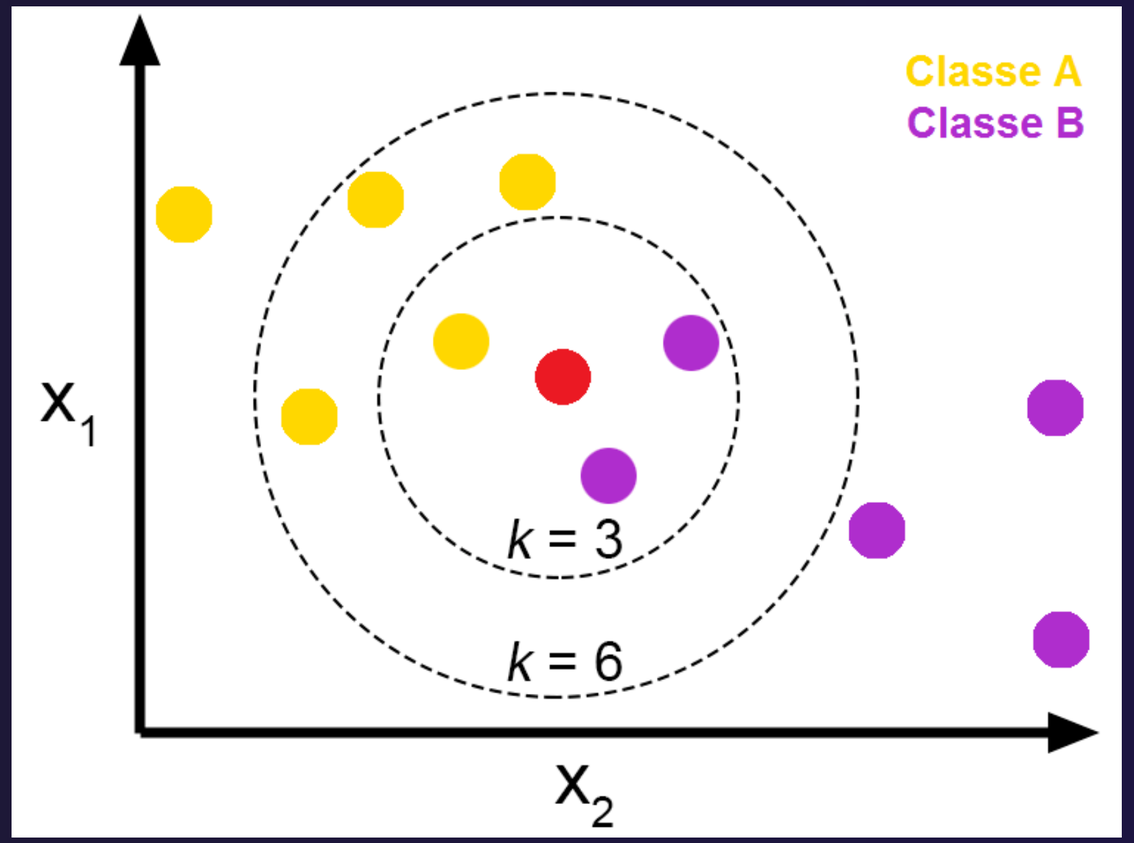
OUT

Dígitos corretos (0, 1, 2, ..., 9).

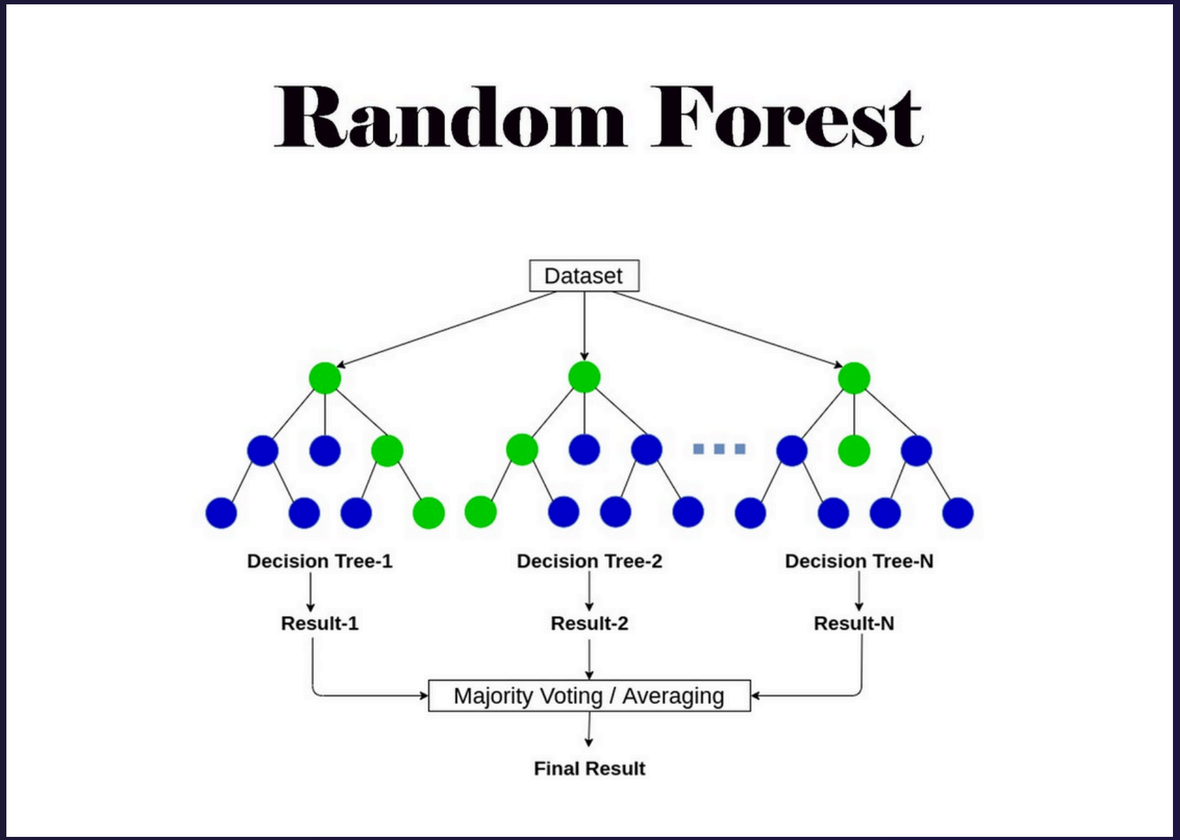
Ferramentas

- **K-Vizinhos Mais Próximos (KNeighborsClassifier)**
- **Floresta Aleatória (RandomForestClassifier)**

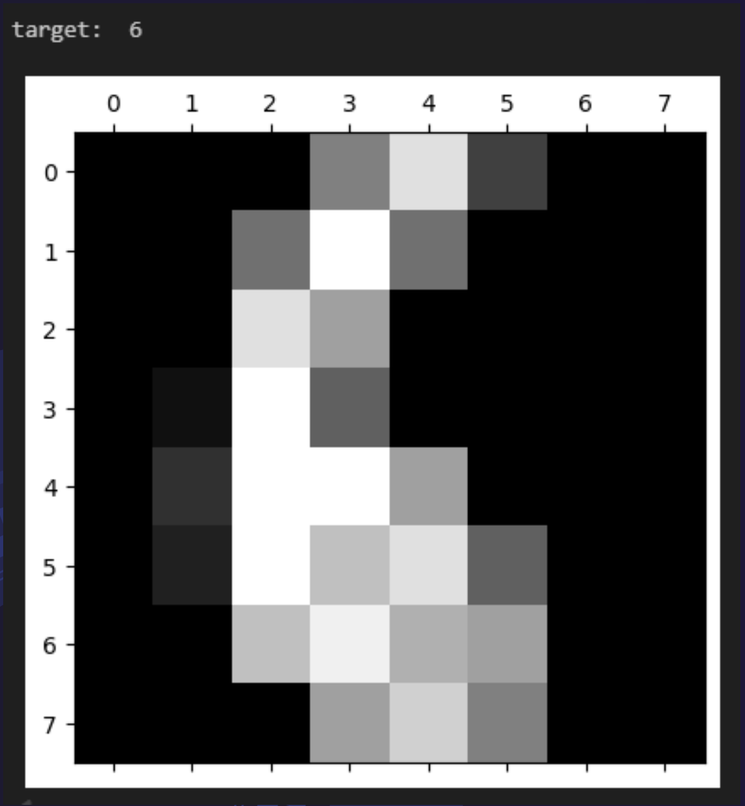
K-Vizinhos Mais Próximos (KNeighborsClassifier)

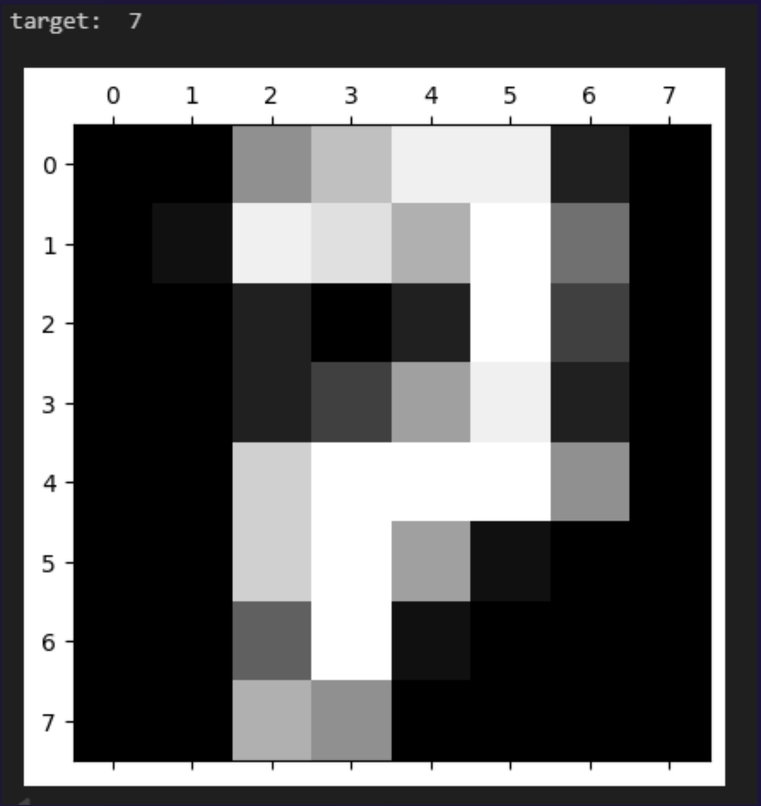


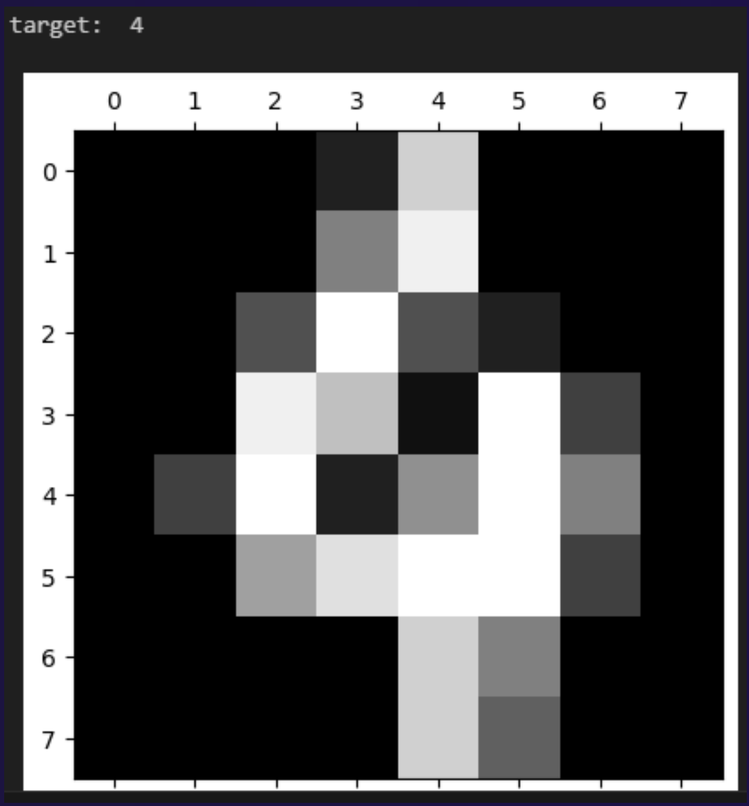
Floresta Aleatória (RandomForestClassifier)



Dataset
(sk.datasets)







Dataset

(sk.datasets)

```
print(data[0])
print(len(data[0]))
```

✓ 0.0s

```
[ 0.  0.  5. 13.  9.  1.  0.  0.  0.  0. 13. 15. 10. 15.  5.  0.  0.  3.
 15.  2.  0. 11.  8.  0.  0.  4. 12.  0.  0.  8.  8.  0.  0.  5.  8.  0.
  0.  9.  8.  0.  0.  4. 11.  0.  1. 12.  7.  0.  0.  2. 14.  5. 10. 12.
  0.  0.  0.  0.  6. 13. 10.  0.  0.  0.]
64
```

Resultados

```
data = digits.data
target = digits.target

x_train, x_test, y_train, y_test = train_test_split(data,target,test_size=0.2)
```

✓ 0.0s

```
model1 = KNeighborsClassifier()
model1.fit(x_train,y_train)
```

✓ 0.0s

KNeighborsClassifier

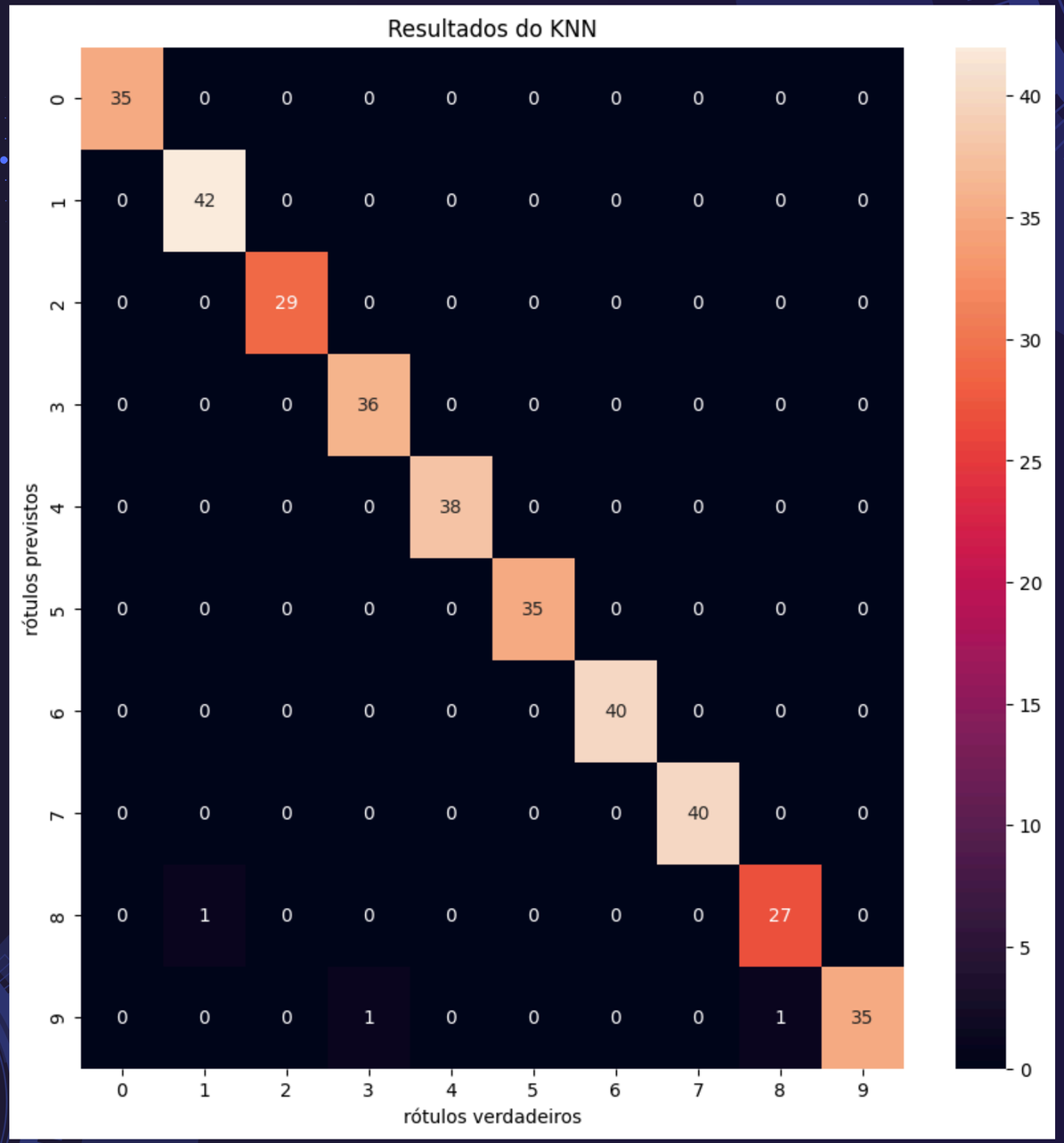
▶ Parameters

```
model2 = RandomForestClassifier()
model2.fit(x_train,y_train)
```

✓ 0.4s

RandomForestClassifier

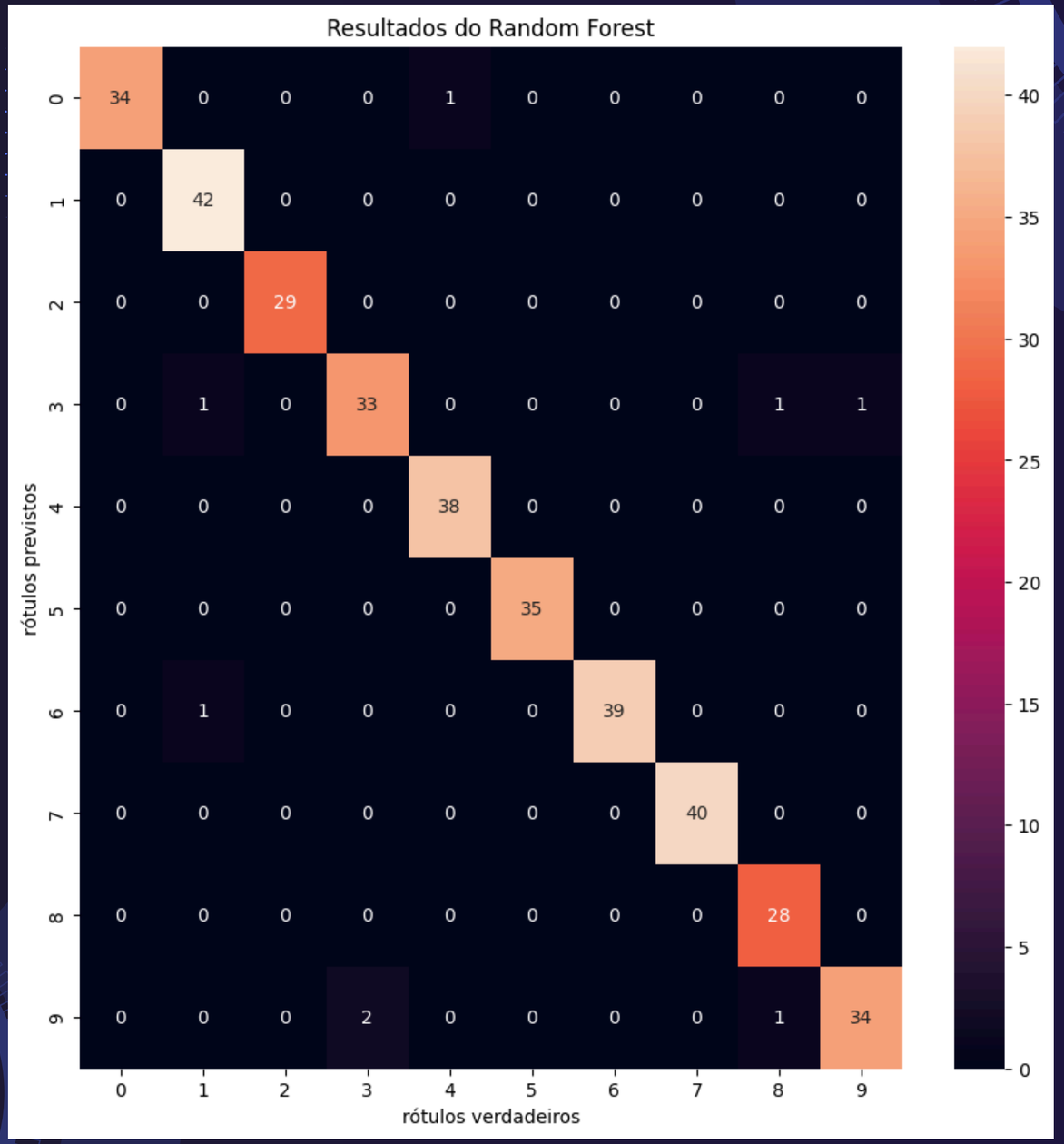
▶ Parameters




```
model1.score(x_test,y_test)
```

✓ 0.0s

```
0.9916666666666667
```




```
model2.score(x_test,y_test)
```

✓ 0.0s

```
0.9777777777777777
```

Análise dos resultados

KNN: Acurácia de 99,16%
RF: Acurácia de 97,77%

O KNN, que é um algoritmo baseado em distância, pode ter se beneficiado da proximidade clara entre os vetores de imagens dos mesmos dígitos.

Obrigado!

Documentação completa:
https://scikit-learn.org/stable/index.html