Roteiro
Aula Prática

FUNDAMENTOS DE INTELIGÊNCIA ARTIFICIAL

Público

ROTEIRO DE AULA PRÁTICA

NOME DA DISCIPLINA: Fundamentos de Inteligência Artificial

Unidade: U4_ Introdução à visão computacional
Aula: A4_ Visão computacional contemporânea

OBJETIVOS

Definição dos objetivos da aula prática:

O objetivo desta atividade é a execução de um código de aprendizado supervisionado para melhor compreensão de conceitos e aplicação das técnicas de machine learning na classificação de imagens.

SOLUÇÃO DIGITAL:

Google Colab

- Python 3.7+, Pandas, Numpy, Matplot, SKlearn

LINK SOLUÇÃO DIGITAL: Google Colab

PROCEDIMENTOS PRÁTICOS E APLICAÇÕES

Procedimento/Atividade

APLICAÇÃO DO CLASSIFICADOR KNN.

Atividade proposta: Nesta atividade você irá executar um código python que realiza a classificação de uma base de dados de imagens de números. O classificador utilizado será o KNN e ao final serão observadas as métricas de avaliação do classificador.

Procedimentos para a realização da atividade:

1. **Preparação e visualização dos dados**

   a) Inicie o Google Colab.
   b) Crie um novo notebook e salve-o como knn_ia.py.
   c) Faça a importação das bibliotecas abaixo:

```
import numpy as np
import pandas as pd
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import matplotlib.pyplot as plt
from sklearn.metrics import (
    confusion_matrix, ConfusionMatrixDisplay,
    accuracy_score, precision_score, recall_score, f1_score,
    classification_report
)
```

d) O dataset utilizado é disponibilizado pelo Scikit learn e você pode checar suas informações através do link: [The Digit Dataset — scikit-learn](#).

e) Faça o download e importação do dataset de dígitos através do sklearn utilizando os comandos:

```
from sklearn.datasets import load_digits
# 1. Carregar o dataset
data = load_digits()
X = data.data
y = data.target

print(X.shape)
```

O comando "*print(X.shape)"* mostra a quantidade de linhas (imagens) e colunas (características) do conjunto. Tire print da saída.

f) Faça um print de uma das imagens para visualizar melhor com que tipo de dados estamos trabalhando.

```
# Exibir a ultima imagem do conjunto
plt.figure(1, figsize=(3, 3))
plt.imshow(data.images[-1], cmap=plt.cm.gray_r, interpolation="nearest")
plt.show()
```

Ao utilizar "*data.images[-1]"* será exibida a última imagem do conjunto, porém você pode alterar o *index* para qualquer outro valor menor que a quantidade de imagens do conjunto para visualizar. Tire print da saída.

g) Crie um *dataframe* dos dados para melhor visualização das colunas. Tire print da saída.

```
# Criar DataFrame com os pixels (64 colunas)
df = pd.DataFrame(X, columns=[f'pixel_{i}' for i in range(X.shape[1])])

# Adicionar coluna com o rótulo (dígito)
df['target'] = y

# Mostrar as primeiras linhas
df.head()
```

2. Treinamento e classificação

a) Faça a divisão do conjunto entre treino e teste. Separe 20% para treino e 80% para teste.

```
# 3. Dividir em treino e teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=

print(f"Número de amostras totais: {len(X)}")
print(f"Amostras de treino: {len(X_train)}")
print(f"Amostras de teste: {len(X_test)}")
```

b) Através do comando abaixo aplique o classificador KNN. Utilize K=5.

```
# 4. Treinar o modelo KNN (k=5, por exemplo)
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
```

c) Após o treinamento, aplique o classificador ao conjunto de testes e exiba a matriz de confusão para conferir o resultado.

```
# 4. Fazer previsões
y_pred = knn.predict(X_test)

# 5. Matriz de confusão
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=data.target_names)
disp.plot(cmap='Blues')
plt.title("Matriz de Confusão - KNN (k=5)")
plt.show()
```

d) Utilize também as métricas de avaliação para compreender os resultados. Tire print da saída.

```
# 6. Métricas de avaliação
print("Acurácia:", accuracy_score(y_test, y_pred))
print("Precisão:", precision_score(y_test, y_pred, average='macro'))
print("Recall:", recall_score(y_test, y_pred, average='macro'))
print("F1-score:", f1_score(y_test, y_pred, average='macro'))
```

e) Altere o valor K definido no item **b** e execute novamente os itens **c** e **d** para observar o quanto o valor de K pode afetar o resultado.

Obs.: Os prints do arquivo de resposta devem ser realizados com o número de K=5.

Checklist:

- ✓ Configuração inicial
- ✓ Visualização do dataset
- ✓ Separação do conjunto entre treino e teste
- ✓ Aplicação do classificador KNN
- ✓ Exibição dos resultados

RESULTADOS

Resultados do experimento:

O aluno deverá entregar:

Arquivo 'knn_ai.py' contendo:

- As funções de cada item do guia.
- Os plots de cada alteração feita na imagem.

Arquivo PDF com prints das saídas dos exercícios:

- 1 e
- 1 f
- 1 g
- 2 d
- Imagem da matriz de confusão.

Resultados de Aprendizagem:

Ao final da atividade, o aluno deverá ser capaz de:

- Compreender o processo de treinamento e classificação de um conjunto de dados.
- Utilizar ferramentas básicas de desenvolvimento com Python (IDLE, módulos, execução de scripts).