Grigoraș Teodora
Grupa 334 AA
11.01.2024

Raport de prezentare
Clasificare folosind K-nn și Naive Bayes

Introducere

Tema

Clasificarea radiografiilor pulmonare cu algoritmul k-Nearest Neighbors (k-NN) și Naive Bayes în două clase, anume: radiografii sănătoase (NORMAL) și radiografii nesănătoase (PNEUMONIE) și analizarea rezultatelor obținute.

Scop

Proiectul are ca scop utilizarea în paralel a celor doi algoritmi de clasificare pe același set de date și compararea rezultatelor obținute.

Date:

Setul de date este compus din 750 de fișiere .jpeg etichetate „NORMAL” și 750 de fișiere .jpeg etichetate „PNEUMONIA”. Aceste fișiere sunt majoritare alb-negru și dimensiunile variază între valori maxime de 2288 x 2363 și minime de 712 x 439. Datorită acestor diferențe dintre date, este nevoie de o preprocesare prin care se dorește aducerea la aceeași dimensiune și la același număr de canale de culori, în cazul de față, alb-negru.

Preprocesarea datelor este efectuată în funcția de încărcare a datelor (load_dataset) și funcționalitatea acesteia este simplă: se citește imagine -> se verifică numărul de canale, iar dacă acesta este egal cu 3 (RGB) -> imaginea este convertită la Gray scale -> imaginea se redimensionează la o dimensiune convenabilă.

Aplicabilități

Detectarea existenței pneumoniei, clasificarea tipului de pneumonie pe baza caracteristicilor din imagini (virală sau bacteriană), determinarea gravității afecțiunii, monitorizarea evoluției, optimizarea timpului de diagnosticare.

Structura codului

Organizare

Codul este redactat în Python 3 și are următorul flux:

–> Încărcarea bibliotecilor.

–> Definirea funcțiilor de încărcare și salvare a datelor.

–> Încărcarea efectivă a datelor și divizarea lor în antrenare și testare.

–> Inițializarea și antrenarea celor doi algoritmi de clasificare (k-NN și NB).

Grigoraș Teodora
Grupa 334 AA
11.01.2024

–> Analizarea rezultatelor.

Procesul de antrenare și validare cu K-nn

Antrenarea modelului: alegerea numărului vecinilor k=3, calcularea distanțelor printr-o matrice de distanțe pentru a măsura similaritatea între exemple și încărcarea setului de date de antrenare în model.

Procesul de testare la K-nn

Pentru setul de testare se folosește modelul k-nn antrenat pe setul de antrenare. Se calculează distanțele dintre fiecare exemplu din setul de testare față de toate exemplele din setul de antrenare și se aleg etichetele de predicție.

Procesul de antrenare și validare cu NB

Antrenarea modelului: am încărcat setul de date de antrenament în modelul de antrenare Gaussian Naive Bayes.

Procesul de testare la NB

Pentru setul de testare se utilizează modelul de antrenare pentru a face predicții. Se calculează probabilitățile posterioare pentru fiecare exemplu din setul de date și se alege clasa de predicție.

Funcționalitate

1. Încărcarea bibliotecilor

```
import os
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, classification_report
from skimage.io import imread
from skimage.transform import resize
import matplotlib.pyplot as plt
import pandas as pd
```

2. Definirea funcțiilor de încărcarea datelor, de salvarea rezultatelor și de salvarea imaginilor rezultate.

```
# Function to load images and labels from the dataset
def load_dataset(root_folder):···


# Function to save results to a CSV file
def save_results(algorithm, y_true, y_pred, class_mapping):···


# Function to save test images with predictions
def save_test_images(algorithm, X_test, y_true, y_pred, class_mapping, probabilities):
```

Grigoraș Teodora
Grupa 334 AA
11.01.2024

3. Încărcarea efectiva a datelor și prelucrarea acestora: împărțirea lor în 80% pentru antrenare și 20% pentru testare.

```
# Load your dataset
dataset_folder = "./train1" # Change this to the path of your dataset folder
X, y, class_mapping = load_dataset(dataset_folder)

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

4. Inițializarea, antrenarea și salvarea datelor pentru algoritmul K-nn

```
# Example 1: k-Nearest Neighbors (k-NN)
# Initialize k-NN classifier
knn_classifier = KNeighborsClassifier(n_neighbors=3)

# Train the model
knn_classifier.fit(X_train, y_train) # Make predictions on the test set
y_test_pred_knn = knn_classifier.predict(X_test)

# Save results and test images for k-NN
save_results('knn', y_test, y_test_pred_knn, class_mapping)
probabilities_knn = knn_classifier.predict_proba(X_test)
save_test_images('knn', X_test, y_test, y_test_pred_knn, class_mapping, probabilities_knn)
```

5. Inițializarea, antrenarea, crearea predicțiilor și salvarea datelor pentru algoritmul NB.

```
# Example 2: Naive Bayes (Gaussian Naive Bayes) # Initialize Naive Bayes classifier
nb_classifier = GaussianNB()

# Train the model
nb_classifier.fit(X_train, y_train)

# Make predictions on the test set
y_test_pred_nb = nb_classifier.predict(X_test)

# Save results and test images for Naive Bayes
save_results('naive_bayes', y_test, y_test_pred_nb, class_mapping)
probabilities_nb = nb_classifier.predict_proba(X_test)
save_test_images('naive_bayes', X_test, y_test, y_test_pred_nb, class_mapping, probabilities_nb)

```

6. Calcularea acurateței pentru cei doi algoritmi.

```
# Evaluate and print accuracy for k-NN
accuracy_knn = accuracy_score(y_test, y_test_pred_knn)
print(f'Acuratețe pentru k-NN: {accuracy_knn:.4f}')

# Evaluate and print accuracy for Naive Bayes
accuracy_nb = accuracy_score(y_test, y_test_pred_nb)
print(f'Acuratețe pentru Naive Bayes: {accuracy_nb:.4f}')
```

7. Analiza detaliată a celor doi algoritmi.

```
# Classification report for k-NN
print("Raport de clasificare pentru k-NN:")
print(classification_report(y_test, y_test_pred_knn, target_names=class_mapping.values()))

# Classification report for Naive Bayes
print("Raport de clasificare pentru Naive Bayes:")
print(classification_report(y_test, y_test_pred_nb, target_names=class_mapping.values()))
```

Grigoraș Teodora
Grupa 334 AA
11.01.2024

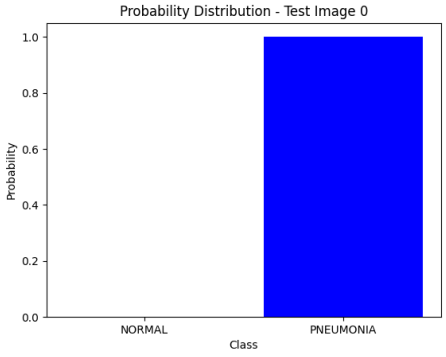
8. Crearea unei diagrame pentru compararea vizuala a acurateței celor doi algoritmi.

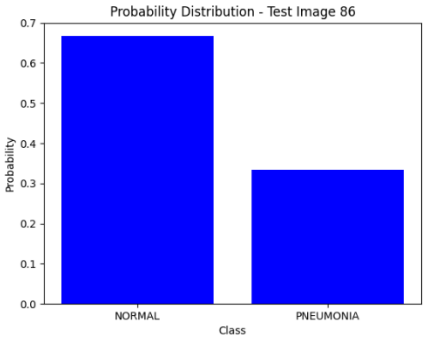
```python
# Etichetele claselor pentru afișare pe axa x
clasificatori = ['k-NN', 'Naive Bayes']
# Valori acuratețe
valori_acuratete = [accuracy_knn , accuracy_nb]

# Creare diagramă de bare
plt.bar(clasificatori, valori_acuratete, color=['blue', 'green'])
plt.ylim(0, 1)

# Adăugare etichete și titluri
plt.xlabel('Clasificatori')
plt.ylabel('Acuratețe')
plt.title('Acuratețe pentru k-NN și Naive Bayes')

# Afișare diagramă
plt.show()
```

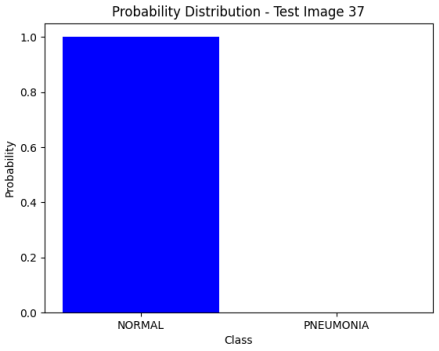
Interpretarea rezultatelor

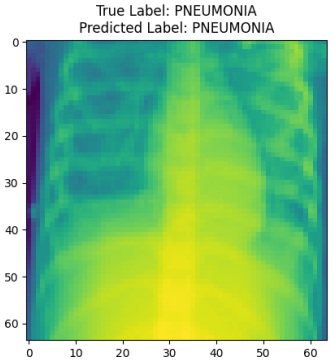
Calcularea acurateței pentru K-nn:

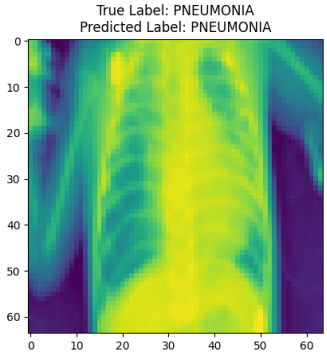
exemplul 0 exemplul 86 exemplul 37

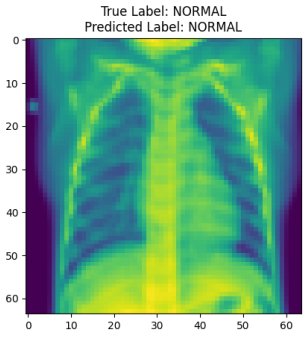








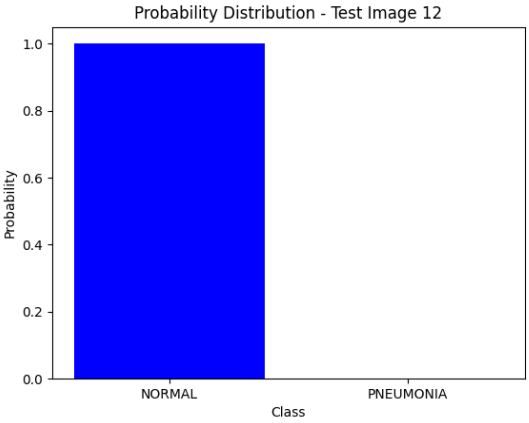


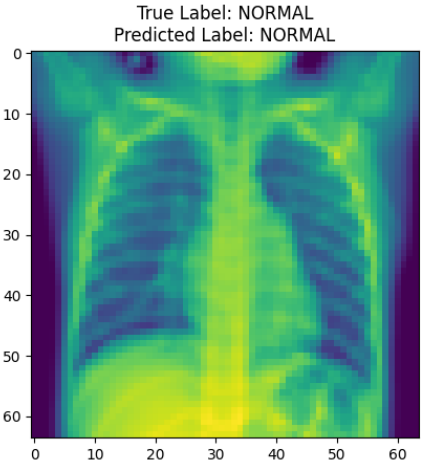


Grigoraș Teodora
Grupa 334 AA
11.01.2024

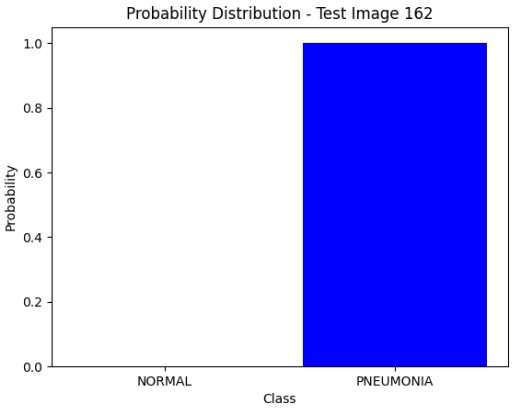
Calcularea acurateței pentru NB:

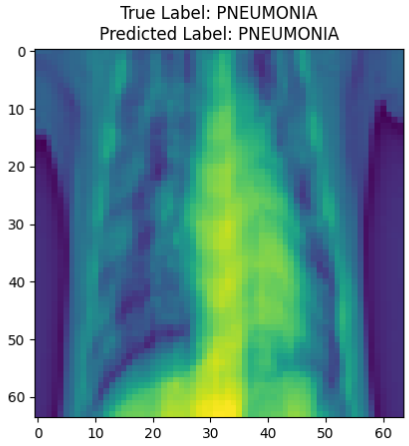
exemplul 12



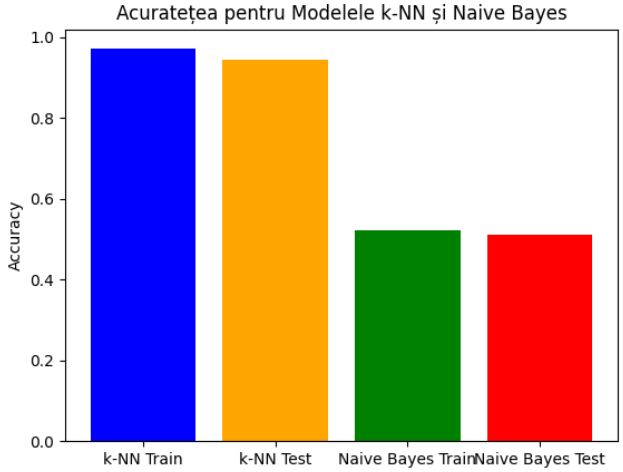


exemplul 162



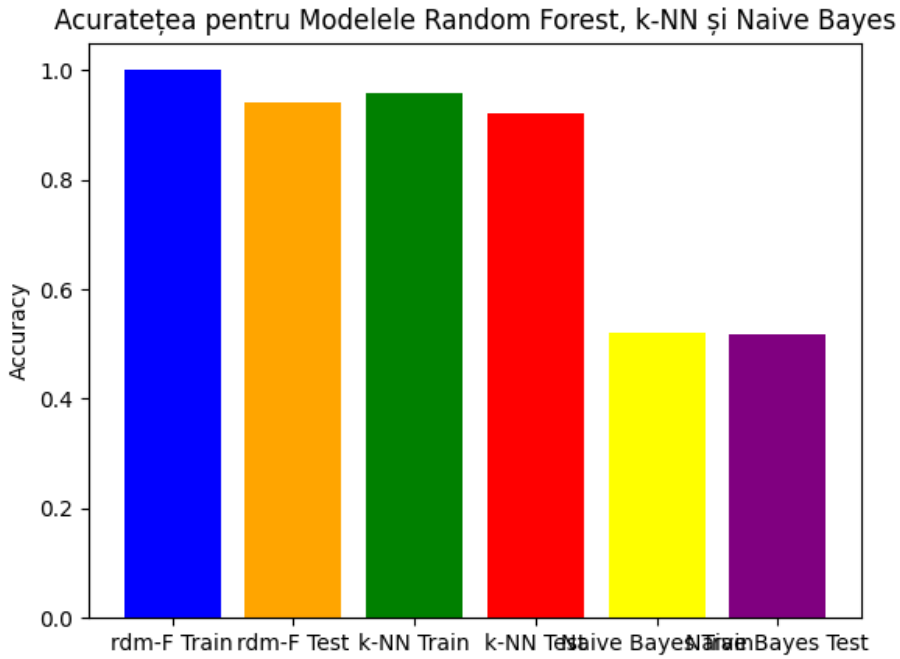


Compararea rezultatelor între setul de train și test al algoritmului K-nn și setul train și test pentru NB



Grigoraș Teodora
Grupa 334 AA
11.01.2024

Compararea rezultatelor cu cele de la algoritmul Random Forest



Avantaje și implicații

Ca avantaje principale ale utilizării metodologiei propuse se regăsesc simplitatea algoritmilor, fiind accesibili chiar și pentru cei care nu au o experiență extinsă în domeniu, eficiență în detectarea anomaliilor sau a datelor neobișnuite și rezultatele sunt ușor de interpretat.

Concluzie personală

Din rezultatele obținute, am remarcat că acuratețea modelului K-nn este remarcabilă pe setul de date utilizat. Acest lucru indică faptul că acest algoritm este eficient în clasificarea, identificarea și generalizarea caracteristicilor specifice ale imaginilor utilizate.

Pe de altă parte, modelul Naive Bayes a obținut o acurateț mult mai scăzută ceea ce indică o performanță modestă în clasificarea radiografiilor pneumonice.

Grigoraș Teodora
Grupa 334 AA
11.01.2024

De asemenea, modelul Random Forest a dovedit o acuratețe mare ceea ce indică o capacitate ridicată de învățare colectivă a unui ansamblu de arbori de decizie.

Alegerea între aceste trei modele de clasificare depinde de natura specifică a setului de date și de caracteristicile problemei de clasificare. În cazul setului meu de date, K-nn si Random Forest s-au dovedit cele mai potrivite.