**Họ và tên: Nguyễn Tiến Việt**

**Lớp: DCCNTT12.10.6**

**Mã sinh viên: 20211382**

**Bài làm**

**1. Thuật toán CART (Classification and Regression Tree)**

**CART** (Cây Phân loại và Hồi quy) là một trong những thuật toán phân lớp và hồi quy phổ biến. Được phát triển bởi Breiman vào năm 1984, CART xây dựng cây nhị phân, với mỗi node (nút) trong cây quyết định việc phân loại dựa trên các thuộc tính của dữ liệu.

**Đặc điểm của CART:**

* **Gini Index**: CART sử dụng chỉ số Gini để chọn các thuộc tính phân chia dữ liệu tốt nhất tại mỗi node. Gini Index đo mức độ "thuần khiết" của tập dữ liệu sau khi được chia. Công thức tính Gini cho một node là:
* **Chia nhị phân**: CART chỉ tạo ra cây nhị phân, tức là mỗi node sẽ có tối đa hai nhánh con.
* **Ưu điểm**: CART được sử dụng rộng rãi vì đơn giản và hiệu quả. Cây nhị phân giúp cây trở nên cân bằng hơn so với các cây có nhiều nhánh.
* **Ứng dụng**: CART được áp dụng cho các bài toán phân lớp (như bài toán trên bộ dữ liệu IRIS và ảnh nha khoa) cũng như hồi quy (dự đoán các giá trị liên tục).

**Quy trình của CART:**

1. **Chọn thuộc tính tốt nhất**: CART chọn thuộc tính nào giúp phân chia tập dữ liệu tốt nhất bằng cách sử dụng chỉ số Gini.
2. **Chia node thành hai nhánh con**: Sau khi chọn thuộc tính, node sẽ chia thành hai nhánh, với mỗi nhánh đại diện cho một nhánh của thuộc tính đã chọn.
3. **Lặp lại đến khi dừng**: Quy trình tiếp tục cho đến khi đạt điều kiện dừng, chẳng hạn khi tất cả các node đều thuộc một lớp hoặc đạt độ sâu tối đa của cây.

**Ưu điểm của CART và Gini Index**

* **Hiệu quả tính toán**: Gini Index tính toán nhanh hơn Entropy vì không có logarit, phù hợp với các bộ dữ liệu lớn.
* **Cây nhị phân**: CART xây dựng cây nhị phân, giúp cây có cấu trúc cân bằng hơn và dễ dàng thực hiện tỉa cây để giảm quá khớp.
* **Khả năng mở rộng**: CART hoạt động tốt với các bài toán phân loại đa lớp và dễ mở rộng cho các bài toán hồi quy.

**Nhược điểm**

* **Quá khớp**: CART có thể dễ bị quá khớp, đặc biệt nếu cây rất sâu và không có tỉa cây thích hợp.
* **Không tối ưu toàn cục**: Quy trình chọn thuộc tính của CART là tham lam, có thể không tìm ra cây có độ chính xác cao nhất.

**2. Thuật toán ID3 (Iterative Dichotomiser 3)**

**ID3** là thuật toán xây dựng cây quyết định được phát triển bởi Ross Quinlan vào năm 1986. ID3 xây dựng cây phân lớp dựa trên thuộc tính có **Information Gain** (IG) lớn nhất ở mỗi bước, chọn các thuộc tính giúp giảm bớt sự bất định của dữ liệu.

**Đặc điểm của ID3:**

* **Information Gain (IG)**: ID3 chọn thuộc tính nào mang lại Information Gain cao nhất để phân chia dữ liệu. Information Gain được tính dựa trên Entropy, trong đó Entropy đo lường sự hỗn loạn hoặc tính không chắc chắn trong dữ liệu:
* **Cây đa nhánh**: Khác với CART, ID3 có thể tạo ra cây đa nhánh, nghĩa là một node có thể có nhiều nhánh tùy thuộc vào số giá trị của thuộc tính đã chọn.
* **Ưu điểm**: ID3 xây dựng cây một cách tham lam, giúp giảm độ phức tạp của cây và thường cho ra kết quả nhanh với dữ liệu nhỏ hoặc trung bình. Tuy nhiên, ID3 có xu hướng dễ bị quá khớp với dữ liệu nhiễu vì nó không sử dụng cơ chế tỉa cây (pruning).

**Quy trình xây dựng cây quyết định ID3**

1. **Tính Entropy của tập dữ liệu:** Đầu tiên, tính Entropy của tập dữ liệu ban đầu. Đây là Entropy của node gốc.
2. **Chọn thuộc tính có Information Gain cao nhất:** Tính Information Gain cho mỗi thuộc tính và chọn thuộc tính có Information Gain lớn nhất để chia node.
3. **Phân chia node theo thuộc tính:** Node hiện tại sẽ được chia thành các nhánh, mỗi nhánh tương ứng với một giá trị của thuộc tính đã chọn. Mỗi nhánh sẽ chứa một tập con của dữ liệu.
4. **Lặp lại cho các node con:** Quy trình trên được lặp lại đệ quy cho từng node con cho đến khi đạt điều kiện dừng, chẳng hạn:
   * Tất cả các mẫu trong node thuộc cùng một lớp (Entropy = 0).
   * Không còn thuộc tính nào để phân chia.
   * Node đạt số lượng mẫu tối thiểu hoặc độ sâu tối đa.
5. **Dừng lại khi hoàn thành:** Cây sẽ dừng khi tất cả node đã đạt điều kiện dừng. ID3 không có cơ chế tỉa cây (pruning), điều này khiến cây dễ bị quá khớp với dữ liệu nhiễu hoặc dữ liệu nhỏ.

**Information Gain là gì?**

**Information Gain (IG)** là thước đo mức độ giảm Entropy khi phân chia dữ liệu theo một thuộc tính. Một thuộc tính có Information Gain cao sẽ giúp phân chia dữ liệu sao cho các node con có Entropy thấp hơn, nghĩa là mức độ thuần nhất cao hơn.

So sánh CART và ID3

|  |  |  |
| --- | --- | --- |
| **Đặc điểm** | **CART** | **ID3** |
| **Tiêu chí chia** | Gini Index | |  | | --- | | Information Gain (Entropy) |  |  | | --- | |  | |
| |  | | --- | | **Kiểu cây** |  |  | | --- | |  | | |  | | --- | | Nhị phân (mỗi node chỉ có 2 nhánh) |  |  | | --- | |  | | |  | | --- | | Đa nhánh (mỗi node có nhiều nhánh nếu cần) |  |  | | --- | |  | |
| |  | | --- | | **Tốc độ** |  |  | | --- | |  | | |  | | --- | | Nhanh hơn với các bài toán lớn |  |  | | --- | |  | | |  | | --- | | Có thể chậm hơn do tính Entropy phức tạp hơn |  |  | | --- | |  | |
| |  | | --- | | **Khả năng tỉa cây** |  |  | | --- | |  | | |  | | --- | | Hỗ trợ (bằng cách giảm độ sâu hoặc tỉa cây) |  |  | | --- | |  | | |  | | --- | | Không hỗ trợ tỉa cây |  |  | | --- | |  | |
| |  | | --- | | **Ứng dụng** |  |  | | --- | |  | | |  | | --- | | Được sử dụng rộng rãi hơn, đa năng |  |  | | --- | |  | | Phù hợp với các bài toán nhỏ hoặc trung bình |

|  |
| --- |
| import os import cv2 import numpy as np from sklearn.datasets import load\_iris from sklearn.model\_selection import train\_test\_split from sklearn.tree import DecisionTreeClassifier from sklearn.metrics import accuracy\_score, classification\_report  # ----- 1. Phân lớp IRIS với CART và ID3 -----  # Tải và chia tập dữ liệu IRIS data = load\_iris() X\_train, X\_test, y\_train, y\_test = train\_test\_split(data.data, data.target, test\_size=0.3, random\_state=42)  # CART Classifier (Gini Index) cart\_model = DecisionTreeClassifier(criterion='gini', random\_state=42) cart\_model.fit(X\_train, y\_train) y\_pred\_cart = cart\_model.predict(X\_test)  print("IRIS - CART (Gini) Accuracy:", accuracy\_score(y\_test, y\_pred\_cart)) print("IRIS - CART (Gini) Classification Report:\n", classification\_report(y\_test, y\_pred\_cart))  # ID3 Classifier (Information Gain) id3\_model = DecisionTreeClassifier(criterion='entropy', random\_state=42) id3\_model.fit(X\_train, y\_train) y\_pred\_id3 = id3\_model.predict(X\_test)  print("IRIS - ID3 (Information Gain) Accuracy:", accuracy\_score(y\_test, y\_pred\_id3)) print("IRIS - ID3 (Information Gain) Classification Report:\n", classification\_report(y\_test, y\_pred\_id3))  # ----- 2. Phân lớp ảnh nha khoa với CART và ID3 -----  # Đường dẫn thư mục chứa 300 ảnh nha khoa image\_folder = "images" labels = [] # Danh sách nhãn của từng ảnh images = [] # Danh sách ảnh đã xử lý  # Đọc và xử lý ảnh for filename in os.listdir(image\_folder):  img\_path = os.path.join(image\_folder, filename)  img = cv2.imread(img\_path, cv2.IMREAD\_GRAYSCALE) # Đọc ảnh grayscale  img\_resized = cv2.resize(img, (64, 64)) # Thay đổi kích thước ảnh về 64x64  images.append(img\_resized.flatten()) # Chuyển đổi thành vector và thêm vào danh sách   # Gán nhãn dựa trên tên ảnh  if "class0" in filename:  labels.append(0) # Ví dụ cho class 0  elif "class1" in filename:  labels.append(1) # Ví dụ cho class 1  X = np.array(images) y = np.array(labels)  # Chia tập dữ liệu X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)  # CART Classifier (Gini Index) cart\_model = DecisionTreeClassifier(criterion='gini', random\_state=42) cart\_model.fit(X\_train, y\_train) y\_pred\_cart = cart\_model.predict(X\_test)  print("Dental Images - CART (Gini) Accuracy:", accuracy\_score(y\_test, y\_pred\_cart)) print("Dental Images - CART (Gini) Classification Report:\n", classification\_report(y\_test, y\_pred\_cart))  # ID3 Classifier (Information Gain) id3\_model = DecisionTreeClassifier(criterion='entropy', random\_state=42) id3\_model.fit(X\_train, y\_train) y\_pred\_id3 = id3\_model.predict(X\_test)  print("Dental Images - ID3 (Information Gain) Accuracy:", accuracy\_score(y\_test, y\_pred\_id3)) print("Dental Images - ID3 (Information Gain) Classification Report:\n", classification\_report(y\_test, y\_pred\_id3)) |