Tree data structure: Decision Tree

Hoàng-Nguyên Vũ

1 Lý thuyết về cây quyết định

1.1 Cây quyết định là gì?

Cây quyết định (Decision Tree) là một mô hình học máy dựa trên cấu trúc cây để đưa ra quyết định hoặc dự đoán. Mỗi nút trong cây biểu diễn một điều kiện trên một thuộc tính, nhánh là kết quả của một điều kiện, và lá chứa giá trị dự đoán cuối cùng.

1.2 Ứng dụng của cây quyết định

- **Phân loại (Classification)**: Xác định nhóm của một đối tượng dựa trên dữ liệu đầu vào.
- **Hồi quy (Regression)**: Dự đoán giá trị số lượng dựa trên dữ liệu đầu vào.
- **Chẩn đoán y tế**: Xác định bệnh dựa trên triệu chứng.
- **Tài chính**: Dự đoán khả năng vỡ nợ của khách hàng.

2 Thước đo Gini trong cây quyết định

2.1 Thước đo Gini là gì?

Gini Index là một chỉ số đo độ tinh khiết của một tập dữ liệu. Nó giúp xác định mức độ hỗn loạn của tập dữ liệu và được sử dụng để chọn thuộc tính tốt nhất để chia nhánh trong cây quyết định.

2.2 Công thức tính Gini

Gini Index được tính bằng công thức:

$$Gini = 1 - \sum_{i=1}^{n} p_i^2$$

Trong đó:

- p_i là tỷ lệ mẫu thuộc lớp i trong tập dữ liệu.
- n là số lượng lớp.

Nếu một tập dữ liệu chỉ chứa một lớp duy nhất, Gini Index bằng 0 (hoàn toàn tinh khiết). Nếu tỷ lệ của các lớp là đều nhau, Gini Index đạt giá trị cao nhất.

2.3 Ví dụ tính Gini Index

Giả sử có tập dữ liệu phân loại như sau:

Tuổi	Mua sản phẩm?
22	Yes
35	No
26	Yes
45	No
30	Yes
40	No

Tỷ lệ của hai lớp:

- $p_{Yes} = \frac{3}{6} = 0.5$
- $p_{No} = \frac{3}{6} = 0.5$

$$Gini = 1 - (0.5^2 + 0.5^2) = 1 - (0.25 + 0.25) = 0.5$$

3 Cách sử dụng Gini trong cây quyết định

3.1 Các bước xây dựng cây quyết định dựa trên Gini

1. **Tính Gini của tập dữ liệu ban đầu**.
2. Chia tập dữ liệu thành các nhóm nhỏ dựa trên một thuộc tính.
3. Tính **Gini trung bình có trọng số** của các nhóm con.
4. Chọn thuộc tính có **Gini thấp nhất** để làm nút phân nhánh.
5. Lặp lại cho đến khi đạt điều kiện dừng.

4 Cài đặt cây quyết định sử dụng Gini

Dưới đây là một triển khai đơn giản của cây quyết định sử dụng Gini Index trong Python:

```python
import numpy as np

def gini_index(groups, classes):
    """Tính toán chỉ số Gini của một tập hợp"""
    total_samples = float(sum([len(group) for group in groups]))
    gini = 0.0
    for group in groups:
        size = float(len(group))
        if size == 0:
            continue
```

```
        score = sum([(row[-1] == c) for row in group for c in classes]) / size
        gini += (1.0 - sum([score ** 2 for c in classes])) * (size / total_samples)
    return gini

# Ví dụ dữ liệu
dataset = [
    [2.8, 'Yes'],
    [1.2, 'No'],
    [3.6, 'Yes'],
    [4.5, 'No'],
    [5.1, 'Yes']
]

# Chia tập dữ liệu theo một giá trị ngưỡng
def split_data(dataset, feature_index, threshold):
    left = [row for row in dataset if row[feature_index] < threshold]
    right = [row for row in dataset if row[feature_index] >= threshold]
    return left, right

# Ví dụ tính Gini cho một cách chia dữ liệu
groups = split_data(dataset, 0, 3.0)
classes = ['Yes', 'No']
gini = gini_index(groups, classes)
print(f'Gini Index: {gini:.4f}')
```

5 Ví dụ: Xây dựng cây quyết định sử dụng OOP

5.1 Lớp TreeNode - Biểu diễn một nút trong cây

Chúng ta định nghĩa một lớp `TreeNode` để lưu trữ thông tin về các nút trong cây.

```
class TreeNode:
    def __init__(self, feature_index=None, threshold=None, left=None, right=None, label=None):
        """
        Khởi tạo một nút trong cây quyết định.
        - feature_index: Chỉ số thuộc tính được chọn để chia.
        - threshold: Ngưỡng giá trị để phân chia dữ liệu.
        - left: Nhánh trái của cây.
        - right: Nhánh phải của cây.
        - label: Nhãn dự đoán nếu là nút lá.
        """
        self.feature_index = feature_index
        self.threshold = threshold
        self.left = left
        self.right = right
        self.label = label
```

5.2 Lớp DecisionTree - Xây dựng cây quyết định

Chúng ta xây dựng lớp `DecisionTree` với các phương thức chính để chia tập dữ liệu và xây dựng cây.

```
import numpy as np

class DecisionTree:
    def __init__(self, max_depth=3):
        """
        Khởi tạo cây quyết định với độ sâu tối đa.
        """
        self.max_depth = max_depth
        self.root = None

    def gini_index(self, groups, classes):
        """
        Tính chỉ số Gini cho một tập hợp.
        """
        total_samples = sum([len(group) for group in groups])
        gini = 0.0
        for group in groups:
            size = len(group)
            if size == 0:
                continue
            score = 0.0
            for class_val in classes:
                proportion = [row[-1] for row in group].count(class_val) / size
                score += proportion ** 2
            gini += (1.0 - score) * (size / total_samples)
        return gini

    def split_data(self, dataset, feature_index, threshold):
        """
        Chia tập dữ liệu dựa trên một thuộc tính và ngưỡng giá trị.
        """
        left = [row for row in dataset if row[feature_index] < threshold]
        right = [row for row in dataset if row[feature_index] >= threshold]
        return left, right

    def best_split(self, dataset):
        """
        Tìm thuộc tính tốt nhất để chia tập dữ liệu.
        """
        class_values = list(set(row[-1] for row in dataset))
        best_index, best_threshold, best_score, best_groups = None, None, float('inf'), None

        for index in range(len(dataset[0]) - 1):
            for row in dataset:
                groups = self.split_data(dataset, index, row[index])
                gini = self.gini_index(groups, class_values)
```

```
                if gini < best_score:
                    best_index, best_threshold, best_score, best_groups = index, row[index], gini, groups
        return best_index, best_threshold, best_groups

    def build_tree(self, dataset, depth=0):
        """
        Xây dựng cây quyết định đệ quy.
        """
        class_values = [row[-1] for row in dataset]

        # Điều kiện dừng: Nếu chỉ có một lớp hoặc đạt đến độ sâu tối đa
        if len(set(class_values)) == 1 or depth >= self.max_depth:
            return TreeNode(label=max(set(class_values), key=class_values.count))

        # Tìm thuộc tính và giá trị ngưỡng tốt nhất để chia dữ liệu
        feature_index, threshold, (left, right) = self.best_split(dataset)

        # Nếu không thể chia tiếp, tạo nút lá
        if not left or not right:
            return TreeNode(label=max(set(class_values), key=class_values.count))

        # Xây dựng nhánh trái và nhánh phải
        left_node = self.build_tree(left, depth + 1)
        right_node = self.build_tree(right, depth + 1)

        return TreeNode(feature_index, threshold, left_node, right_node)

    def fit(self, dataset):
        """
        Huấn luyện cây quyết định bằng cách xây dựng cây từ dữ liệu đầu vào.
        """
        self.root = self.build_tree(dataset)

    def print_tree(self, node=None, depth=0):
        """
        In ra cây quyết định theo dạng phân cấp.
        """
        if node is None:
            node = self.root

        if node.label is not None:
            print(f"{'  ' * depth} [Leaf] Label: {node.label}")
        else:
            print(f"{'  ' * depth} [Node] Feature {node.feature_index} <= {node.threshold}")
            self.print_tree(node.left, depth + 1)
            self.print_tree(node.right, depth + 1)
```

5.3 Ví dụ sử dụng - Xây dựng cây từ tập dữ liệu

Chúng ta thử nghiệm xây dựng một cây quyết định với một tập dữ liệu đơn giản.

```
# Tập dữ liệu đơn giản: [Thuộc tính, Nhãn]
dataset = [
    [2.8, 'Yes'],
    [1.2, 'No'],
    [3.6, 'Yes'],
    [4.5, 'No'],
    [5.1, 'Yes']
]

# Khởi tạo và huấn luyện cây quyết định
tree = DecisionTree(max_depth=3)
tree.fit(dataset)

# In ra cây quyết định
print("Cây quyết định được xây dựng:")
tree.print_tree()
```

5.4 Kết quả mong đợi

Sau khi chạy đoạn mã trên, cây quyết định sẽ được hiển thị theo dạng phân cấp:

```
Cây quyết định được xây dựng:
 [Node] Feature 0 <= 3.6
   [Node] Feature 0 <= 2.8
     [Leaf] Label: No
     [Leaf] Label: Yes
   [Node] Feature 0 <= 4.5
     [Leaf] Label: Yes
     [Leaf] Label: No
```

6 Bài tập thực hành

Bài 1: Tính chỉ số Gini cho một tập dữ liệu

Tính toán chỉ số Gini cho tập dữ liệu sau:

Lương	Đủ điều kiện vay vốn?
50	Yes
20	No
30	No
70	Yes
40	No
60	Yes

Bài 2: Mở rộng cây quyết định

- Thêm nhiều thuộc tính hơn vào tập dữ liệu (ví dụ: Tuổi, Điểm tín dụng).
- Cải tiến thuật toán để xây dựng cây với nhiều thuộc tính.
- In ra cây theo cách dễ đọc hơn.