Lab 2

Giảng viên: **Lê Ngọc Thảo**

Trợ giảng: **Lê Ngọc Thành**

Nguyễn Trần Duy Minh

Sinh viên: **Lê Nguyên Thái**

Mã số sinh viên: **21127162**

Contents

1. Prepare the data sets:

- Ở phần này em, em đã sử dụng hàm có sẵn đó chính là hàm train_test_split, đây là hàm sẽ hỗ trợ việc tách data thành 2 phần train/test với tỷ lệ phần test là test_size và tách theo sự cân bằng tỷ lệ của thuộc tính stratify (thường là label).

```
feature_train, feature_test, label_train, label_test = train_test_split(feature, label, test_size=pro_test, stratify=label)
```

- Việc trực quan hóa em sẽ trình bày minh họa của cả 4 thử nghiệm trong file Jupyter notebook ạ.

2. Building the decision tree classifiers

```
def create_decision_tree(feature_dataset, label_dataset,depth=None):

    tree = DecisionTreeClassifier(criterion='entropy',max_depth= depth)
    tree.fit(feature_dataset,label_dataset)
    return tree
```

✓ 0.0s

- Hàm chính em sử dụng đó chính là DecisionTreeClassifier, với tham số criterion là ‘entropy’ để có thể từ đó mới sử dụng được information gain như đề bài yêu cầu.
- Sau khi tạo ra cây, em sẽ train cho model của cây bằng cách sử dụng hàm fit để đưa các tập train vào.

Xuất ảnh:

```
def export_decision_tree(decision_tree_model,tail):
    # Visualize cây quyết định bằng Graphviz
    dot_data = tree.export_graphviz(decision_tree_model, out_file=None,
                                    feature_names=feature_name,
                                    class_names=class_name,
                                    filled=True, rounded=True,
                                    special_characters=True,
                                    impurity=True)
    graph = graphviz.Source(dot_data)
    graph.render("decision_tree"+tail)  # Lưu biểu đồ vào tệp "decision_tree.pdf"
    #graph.view()  # Hiển thị biểu đồ
```

- Vì cây có kích thước quá lớn nên em sẽ không thể để xuất thẳng trong fle jupyter notebook được mà em tạo ảnh ra bằng file pdf với đuôi tail để biểu thị đây là loại ảnh với bao nhiêu tầng được sử dụng (2,3,4,5,6,7,None).

3. Evaluating the decision tree classifiers

3.1 Interpret the classification report and the confusion matrix

- Sau khi xuất ra classification report và ma trận tương quan em đã thu về các kết quả tương ứng cho từng cây quyết định của các phần thử nghiệm (40/60, 60/40, 80/20, 90/10).
- Trong confusion matrix (ma trận tương quan) theo em hiểu thì ta sẽ nhận được 1 ma trận vuông với kích nxn với n là số lượng lớp (số lượng giá trị khác biệt của lớp label). Với mỗi cột j sẽ đại diện cho số lượng các mẫu được dự đoán ở lớp j và hàng i sẽ là sự phân tách giữa số lượng các mẫu thực tế ở lớp i và sự giao thoa này tạo nên ma trận tương quan.
    - Các giá trị trên đường chéo chính sẽ là số lượng mẫu được dự đoán ở lớp i và thực tế là mẫu này là ở lớp i. Hay nói cách khác là là số lượng mẫu dự đoán đúng sẽ là tổng đường chéo chính. Ví dụ tổng của (A[i][i] với i từ 1 đến n) = 1000 thì tổng số lượng mẫu dự đoán đúng là 1000 mẫu.
    - Các giá trị khác đường chéo chính sẽ là có dạng A[i][j] với i khác j. Thì có nghĩa là số lượng mẫu được dự đoán ở lớp j nhưng thực tế lại là ở lớp i. Hay nói cách khác là tổng của các A[i][j] với i khác j sẽ là số lượng mẫu bị dự đoán sai so với thực tế.
- Trong classification report:
    - Precison: có nghĩa là tỷ lệ mà lớp thứ i dự đoán đúng THẬT SỰ trên cho tổng số lượng đã dự đoán thuộc lớp i. Hay nói các khác dễ hiểu thì sẽ là:
        - A[i][i] / (tổng A[j][i] với j từ 1 đến n)
    - Recall: tỷ lệ số lượng đã dự đoán đúng lớp cần xét trên tổng số mẫu thực tế thuộc lớp này. Hay một cách dễ hiểu thì việc ta dự đoán ĐÚNG THỰC TẾ một lớp là j trên cho tổng số lượng mẫu thực tế thuộc lớp j là bao nhiêu? Tương tự như trên ta có công thức sau:
        - A[j][j] / (tổng A[j][i] với i từ 1 đến n)
    - F1-score: chỉ số cân bằng giữa precision và recall để đánh giá cả hai khía cạnh chất lượng của mô hình.

$$F\text{-}measure = \frac{2*Recall*Precision}{Recall + Precision}$$

        - Nếu một trong hai chỉ số recall hoặc precision thấp thì sẽ kéo f1-score này xuống thấp theo. Đồng nghĩa với việc chỉ số này chỉ cao khi cả 2 giá trị Recall và Precision đều cao.
    - Support là số lượng mẫu thực tế thuộc lớp đang xét.
        - Support của lớp i sẽ là: Tổng của A[i][j] với j từ 1 đến n.
    - Accuracy: Tỷ lệ số dự đoán đúng trên tổng số dự đoán.
        - (Tổng đường chéo chính)/(tổng A[i][j] với i , j từ 1 đến n).

3.2 Comments on the performances of those decision tree classifiers

- Em thấy toàn bộ các thông số precision, recall, f1-score và accuracy tăng dần từ bắt đầu từ mô hình train với tỷ lệ train/test là:
    - 40/60 -> 60/40 -> 80/20 -> 90/10
    - Với tỷ lệ huấn luyện cao (90/10)
        - Điều này cũng có nghĩa là khi một mô hình được huấn luyện nhiều hơn thì sẽ tăng cao khả năng dự đoán chính xác của mô hình đó. Vì recall khi càng tăng thì có nghĩa là tỷ lệ dự đoán đúng mẫu thuộc lớp i trên cho tổng số mẫu thực tế thuộc lớp i ngày càng tăng dẫn đến việc độ chính xác của việc dự đoán ngày càng cao, tương tự với accuracy là chính xác xét trên tổng thể, precision và f1-score cũng như thế.
        - Nhưng cũng có thể là vì bộ test ngày càng nhỏ khiến cho tỷ lệ chính xác ngày càng tăng xảy ra dẫn đến tình trạng overfitting và mất khả năng tổng quát hóa.
    - Với tỷ lệ huấn luyện quá thấp (40/60)
        - Nếu dữ liệu của tập huấn luyện quá thấp dẫn đến mô hình có thể không học đủ từ dữ liệu huấn luyện và không thể tổng quát hóa tốt cho dữ liệu mới.
        - Đó cũng chính là lý do vì sao tỷ lệ chính xác của cây lại thấp. Có thể dẫn đến tình trạng underfitting.
    - Nên em thấy tỷ là 80/20 hoặc 60/40 sẽ khá hợp lý trong trường hợp này.

4. The depth and accuracy of a decision tree

- Em sử dụng biểu đồ đường trong file jupyter notebook để miêu tả độ chính xác qua từng độ sâu.
- Thống kê:

max_depth	None	2	3	4	5	6	7
Accuracy	0.99	0.75	0.77	0.81	0.84	0.87	0.9

- Em nhận xét là khi độ sâu của cây càng sâu thì độ chính xác tổng thể (Accuracy) ngày càng lớn
- Đặt biệt trong đó lớn nhất là độ chính xác của depth = None (không giới hạn độ sâu) vì đây là trường hợp cây quyết định sẽ phát triển đến khi nào không còn có sự chia tách nào mà có thể cải thiện độ tinh khiết hoặc giảm độ đo lỗi. Điều này có thể dẫn đến việc tạo ra một cây phức tạp và có khả năng overfitting dữ liệu huấn luyện. Và trong trường hợp này em nhận thấy rằng vì có quá nhiều dữ liệu dẫn đến cái cây với depth=None vô cùng lớn dẫn đến em nhìn vào cây và không thể hiểu cây nó đang mô tả về cái gì cả.
- Từ đó, việc ta để cho sự phát triển của cây cứ đâm sâu để tạo ra độ chi tiết một cách quá cụ thể thì sẽ làm cho cây mất đi tính tổng quát hóa và dẫn đến việc cây giống như là đang ‘học thuộc bài’ từ dữ liệu train và việc này là không tốt chút nào.
- Bên cạnh khác việc để độ sâu của cây quá nhỏ (depth=2 hoặc depth=3) có thể không đủ mạnh để học các mẫu phức tạp trong dữ liệu. Kết quả là mô hình sẽ thiếu khả năng tổng quát hóa và có thể không dự đoán tốt trên dữ liệu huấn luyện cũng như dữ liệu mới. Và tương tự là có thể dẫn đến tình trạng Underfitting. Cũng chính là lý do vì sao độ chính xác của chúng lại thấp (0.75 va 0.77)
- Để cây đảm bảo ổn định thì nên giới hạn lại độ sâu của cây ở mức vừa phải và việc chọn giá trị max_depth cần dựa vào kiểm định chéo hoặc thử nghiệm trên tập dữ liệu kiểm tra.