Decision Tree Lab: ID3, C4.5, CART comparisons

Making The Trees And Evaluating Accuracy

highlight the percentages for each algorithm for both data sets, Green for Drug, yellow for Heart

```
In [1]: import pandas as pd
        from chefboost import Chefboost as chef
```

```
In [2]: drug = pd.read_csv('/Users/lukehenry/Documents/Jupyter/Decision Tree Lab/dru
        stroke = pd.read_csv('/Users/lukehenry/Documents/Jupyter/Decision Tree Lab/h
        stroke[['stroke']] = stroke[['stroke']].replace(1,'Yes')
        stroke[['stroke']] = stroke[['stroke']].replace(0,'No')
```

```
In [3]: algorithms = ['ID3','C4.5','CART']

        print("DRUG DATA \n")
        for algorithm in algorithms:
            config = {'algorithm': algorithm}
            model = chef.fit(drug, config, target_label = "Drug")
            if algorithm != 'CART':
                print("\nNEXT ALGORITHM \n")
        print("\nSTROKE DATA \n")
        for algorithm in algorithms:
            config = {'algorithm': algorithm}
            model = chef.fit(stroke, config, target_label = "stroke")
            if algorithm != 'CART':
                print("\nNEXT ALGORITHM \n")
```

```
DRUG DATA

[INFO]:  5 CPU cores will be allocated in parallel running
ID3  tree is going to be built...
-------------------------
finished in  0.6945481300354004  seconds
-------------------------
Evaluate  train set
-------------------------
Accuracy:  91.5 % on  200  instances
Labels:  ['DrugY' 'drugC' 'drugX' 'drugA' 'drugB']
Confusion matrix:  [[79, 0, 0, 0, 0], [4, 16, 0, 0, 0], [4, 0, 54, 0, 0], [
3, 0, 0, 18, 0], [1, 0, 0, 5, 16]]
Decision  DrugY  => Accuray:  94.0 %, Precision:  100.0 %, Recall:  86.8132
%, F1:  92.9412 %
Decision  drugC  => Accuray:  98.0 %, Precision:  80.0 %, Recall:  100.0 %,
F1:  88.8889 %
Decision  drugX  => Accuray:  98.0 %, Precision:  93.1034 %, Recall:  100.0
%, F1:  96.4285 %
Decision  drugA  => Accuray:  96.0 %, Precision:  85.7143 %, Recall:  78.26
09 %, F1:  81.8182 %
Decision  drugB  => Accuray:  97.0 %, Precision:  72.7273 %, Recall:  100.0
%, F1:  84.2105 %

NEXT ALGORITHM

[INFO]:  5 CPU cores will be allocated in parallel running
C4.5  tree is going to be built...
-------------------------
finished in  0.645028829574585  seconds
-------------------------
Evaluate  train set
-------------------------
Accuracy:  87.0 % on  200  instances
Labels:  ['DrugY' 'drugC' 'drugX' 'drugA' 'drugB']
Confusion matrix:  [[80, 0, 0, 0, 0], [4, 16, 0, 0, 0], [4, 0, 54, 0, 0], [
2, 0, 0, 20, 12], [1, 0, 0, 3, 4]]
Decision  DrugY  => Accuray:  94.5 %, Precision:  100.0 %, Recall:  87.9121
%, F1:  93.5673 %
Decision  drugC  => Accuray:  98.0 %, Precision:  80.0 %, Recall:  100.0 %,
F1:  88.8889 %
Decision  drugX  => Accuray:  98.0 %, Precision:  93.1034 %, Recall:  100.0
%, F1:  96.4285 %
Decision  drugA  => Accuray:  91.5 %, Precision:  58.8235 %, Recall:  86.95
65 %, F1:  70.1754 %
Decision  drugB  => Accuray:  92.0 %, Precision:  50.0 %, Recall:  25.0 %,
F1:  33.3333 %

NEXT ALGORITHM

[INFO]:  5 CPU cores will be allocated in parallel running
CART  tree is going to be built...
-------------------------
finished in  0.6131381988525391  seconds
-------------------------
Evaluate  train set
-------------------------
Accuracy:  91.5 % on  200  instances
Labels:  ['DrugY' 'drugC' 'drugX' 'drugA' 'drugB']
Confusion matrix:  [[79, 0, 0, 0, 0], [4, 16, 0, 0, 0], [4, 0, 54, 0, 0], [
3, 0, 0, 18, 0], [1, 0, 0, 5, 16]]
Decision  DrugY  => Accuray:  94.0 %, Precision:  100.0 %, Recall:  86.8132
```

```
%, F1:  92.9412 %
Decision  drugC  => Accuray:  98.0 %, Precision:  80.0 %, Recall:  100.0 %,
F1:  88.8889 %
Decision  drugX  => Accuray:  98.0 %, Precision:  93.1034 %, Recall:  100.0
%, F1:  96.4285 %
Decision  drugA  => Accuray:  96.0 %, Precision:  85.7143 %, Recall:  78.26
09 %, F1:  81.8182 %
Decision  drugB  => Accuray:  97.0 %, Precision:  72.7273 %, Recall:  100.0
%, F1:  84.2105 %

STROKE DATA

[INFO]:  5 CPU cores will be allocated in parallel running
ID3  tree is going to be built...
-------------------------
finished in  16.660999298095703  seconds
-------------------------
Evaluate  train set
-------------------------
Accuracy:  96.3013698630137 % on  5110  instances
Labels:  ['Yes' 'No']
Confusion matrix:  [[86, 26], [163, 4835]]
Precision:  76.7857 %, Recall:  34.5382 %, F1:  47.6455 %

NEXT ALGORITHM

[INFO]:  5 CPU cores will be allocated in parallel running
C4.5  tree is going to be built...
-------------------------
finished in  9.421061992645264  seconds
-------------------------
Evaluate  train set
-------------------------
Accuracy:  95.81213307240705 % on  5110  instances
Labels:  ['Yes' 'No']
Confusion matrix:  [[53, 18], [196, 4843]]
Precision:  74.6479 %, Recall:  21.2851 %, F1:  33.125 %

NEXT ALGORITHM

[INFO]:  5 CPU cores will be allocated in parallel running
CART  tree is going to be built...
-------------------------
finished in  15.53757905960083  seconds
-------------------------
Evaluate  train set
-------------------------
Accuracy:  96.4774951076321 % on  5110  instances
Labels:  ['Yes' 'No']
Confusion matrix:  [[84, 15], [165, 4846]]
```

```
Precision:  84.8485 %, Recall:  33.7349 %, F1:  48.2758 %
```

Analysis

What are the three algorithms?

ID3 (Iterative Dichotomiser 3) algorithm was developed by Ross Quinlan in 1986. It builds a decision tree by recursively partitioning the data set into subsets based on the attribute that provides the maximum information gain. The algorithm is suited for categorical target variables.

C4.5 (Classifier Version 4.5) is an improvement over the ID3 algorithm. It was also developed by Ross Quinlan and published in 1993. It can handle both continuous and categorical target variables, and it uses a statistical method called gain ratio to select the attribute to split on, which helps to deal with biases in the information gain measure used in ID3.

CART (Classification and Regression Trees) is another decision tree algorithm that can be used for both classification and regression problems. It was developed by Breiman et al. in the 1980s. CART uses a binary splitting method, where each non-leaf node has two child nodes, and it selects the split that minimizes the impurity measure. CART can handle both categorical and continuous variables, and it is used in a variety of fields, such as finance and healthcare.

What does our results tell us about the algorithms in terms of efficiency?

I chose two use two different data sets to test the algorithms with to get more informative results than just using one

Drug Data: This data set is meant to be able to prescribe a certain drug based on a person's symptoms

```
ID3: Ran the slowest out of the three algorithms and was tied
for first with 91.5% accuracy
C4.5: Ran the second fastest out the algorithms, but came in
last place for accuracy: having 87.0%
CART: Ran the fastest out of the three algorithms, and was
tied for first with 91.5% accuracy
```

CART is the most efficient with this data set

Stroke Data: This set of data is meant to be able to predict if a person is likely to have a stroke based on many health attributes

```
ID3: Similar to the last data set, this algorithm ran the
slowest out of the three. Second place for accuracy, having
96.3%
C4.5: This algorithm ran the fastest out of the three;
however, it had the lowest accuracy with 95.8%
CART: For this data set it came in second place for speed,
but had the highest accuracy with 96.47%
```

Important notice: Results from Stroke data are to be trusted more since it is a larger data set

In [9]:

```python
print("Drug Data Shape: ", drug.shape)
print("Stroke Data Shape: ", stroke.shape)
```

```
Drug Data Shape:  (200, 6)
Stroke Data Shape:  (5110, 12)
```

What causes the results to be the way that they are?

To start off, even though my data does not specifically show it, C4.5 is an improved version of ID3, so it is a more reliable algorithm. ID3 tends to overfit its data, C4.5 avoids this by using a gain ratio rather than information gain when choosing an attribute to provide the most information. However this can be inferred because of the significant change that C4.5 had compared to ID3 when the algorithms were provided with larger data. Drug data has only 200 rows in the training set, where Stroke data has a little over 5000. There was a big improvement in both accuracy and speed, it still came in last place for accuracy but only by a tiny bit, and in my opinion the speed improvement compensates enough for the accuracy shortchange. CART is well known for its ability to handle large data sets, but that doesn't mean that it is flawless. Unlike the other two, it uses binary splitting which tends to result in the best accuracy, combined with using a gain ratio (like C4.5), boosts it even more to make it the best all around out of the three algorithms in most cases. It is difficult to directly say that one algorithms is better than the other two in this case because decision tree algorithms can also be influenced by other factors such as the size and quality of the dataset, the number and type of features, and the parameter settings of the algorithms.

In summary, ID3 is a basic decision tree algorithm, while C4.5 is an improvement over ID3. CART is also an improvement to ID3 is a binary decision tree algorithm that tends to perform the best out of the three in most cases.

Sources

Drug data: https://www.kaggle.com/datasets/prathamtripathi/drug-classification

Stroke data: https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

Information on ID3. C4.5, CART: https://bitmask93.github.io/ml-blog/ID3-C4-5-CART-and-Pruning/

Chefboost Source: https://github.com/serengil/chefboost