Decision trees
Trần Đăng Khoa – B2014925 – M04

1) Given dataset Golf with 4 attributes Outlook, Temp, Humidity, Windy and an attribute Play (class).

Outlook	**Temperature**	**Humidity**	**Windy**	**Class**
sunny	85	85	false	Don't Play
sunny	80	90	true	Don't Play
overcast	83	78	false	Play
rain	70	96	false	Play
rain	68	80	false	Play
rain	65	70	true	Don't Play
overcast	64	65	true	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
sunny	75	70	true	Play
overcast	72	90	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play

- How to build the decision tree model fro classifying the dataset

Start by determining the root of the tree and the class column:

Step 1: Determine the Root of the Tree.
Step 2: Calculate Entropy for The Classes.
Step 3: Calculate Entropy After Split for Each Attribute.
Step 4: Calculate Information Gain for each split.

Step 5: Perform the Split.
Step 6: Perform Further Splits.
Step 7: Complete the Decision Tree.

- How many inductive rules are there in the decision tree model

Three of them represent very different approaches.

1. OneR learns rules from a single feature. OneR is characterized by its simplicity, interpretability and its use as a benchmark.
2. Sequential covering is a general procedure that iteratively learns rules and removes the data points that are covered by the new rule. This procedure is used by many rule learning algorithms.
3. Bayesian Rule Lists combine pre-mined frequent patterns into a decision list using Bayesian statistics. Using pre-mined patterns is a common approach used by many rule learning algorithms.

- Use the decision tree model to classify 3 examples as follows:

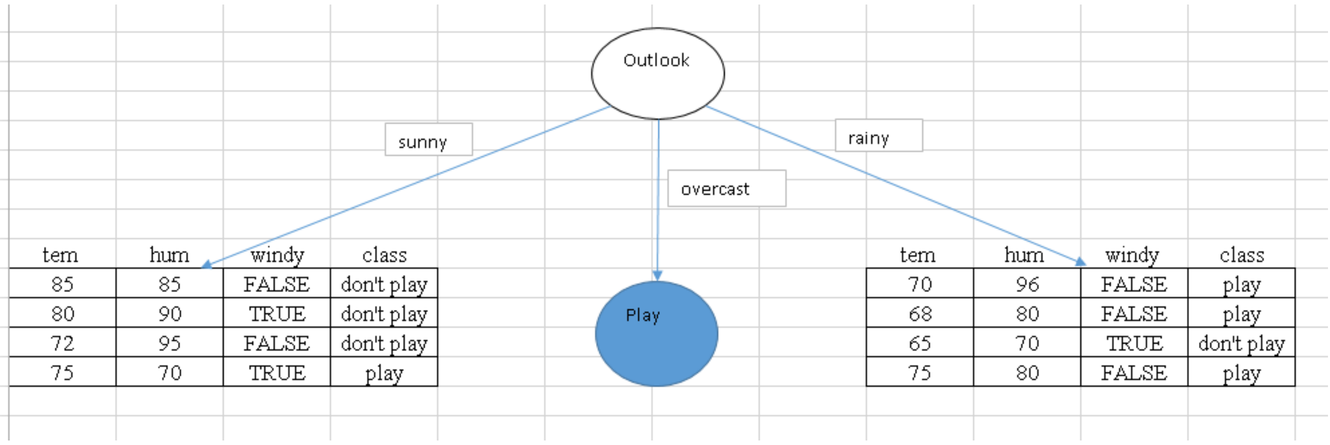
- Outlook:

- "sunny" [2yes, 3no]= entropy(2/5,3/5) = -2/5log(2/5) – 3/5log(3/5) = 0.971

- "overcast" [4 yes, 0 no] = 0

- "rainy" [3 yes, 2 no] = entropy (3/5, 2/5) = -3/5log(3/5) – 2/5log(2/5) = 0.971

⇨ Inpurity (outlook) = (5/14) * 0.971 + (4/14)*0 + (5/14)* 0.971 = 0.693



tem	hum	windy	class
85	85	FALSE	don't play
80	90	TRUE	don't play
72	95	FALSE	don't play
75	70	TRUE	play

tem	hum	windy	class
70	96	FALSE	play
68	80	FALSE	play
65	70	TRUE	don't play
75	80	FALSE	play

Sunny:

Tem (>70 and <=70)

" >70" [1 play ; 3 don't play] => entropy (1/4 ; 3/4) = 1

"<=70" [1 play; 0 don't play] => entropy (1/1 ; 0/1) = 0

⇨ Impurity = 4/5* entropy (1/4 ; 3/4) + 1/5 * entropy (1/1; 0/1) = 0.8

Hum (>75 and <= 75)

">75" [0 Play; 3 Don't play] => Entropy (0/3; 3/3) = 0

"<=75" [2 Play; 0 Don't play] => Entropy (2/2; 0/2) = 0

⇨ Impurity = 4/5*Entropy (1/4; 3/4) + 1/5*Entropy (1/1; 0/1) = 0

Windy (False and True)

"False" [1 Play; 2 Don't play] => Entropy (1/3; 2/3) = 1

"True" [1 Play; 1 Don't play] => Entropy (1/2; 1/2) = 1

⇨ Impurity = 3/5*Entropy (1/3; 2/3) + 2/5*Entropy (1/2; 1/2) = 0,95

Rain:

Tem (>70 and <=70)

">70" [1 Play; 1 Don't play] => Entropy (1/2; 1/2) = 1

"<=70" [2 Play; 1 Don't play] => Entropy (2/3; 1/3) = 1

⇨ Impurity = 4/5*Entropy (1/4; 3/4) + 1/5*Entropy (1/1; 0/1) = 0.95

Hum (>75 and <=75)

">75" [3 Play; 1 Don't play] => Entropy (3/4; 1/4) = 1

"<=75" [0 Play; 1 Don't play] => Entropy (0/1; 1/1) = 0

⇨ Impurity = 4/5*Entropy (1/4; 3/4) + 1/5*Entropy (1/1; 0/1) = 0.8

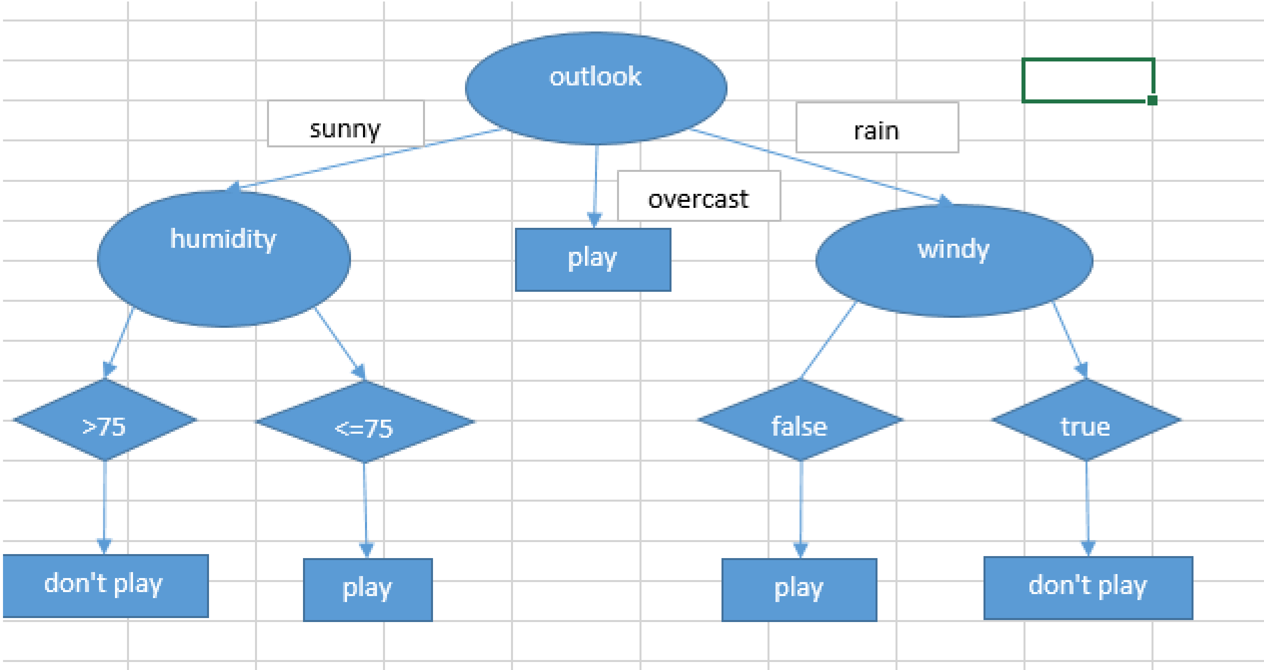
Windy (False and True)

"False" [3 Play; 0 Don't play] => Entropy (3/3; 0/3) = 0

"True" [0 Play; 2 Don't play] => Entropy (0/2; 2/2) = 0

⇨ Impurity = 3/5*Entropy (3/3; 0/3) + 2/5*Entropy (0/2; 2/2) = 0

Decision tree model



⇨ **Result**

Outlook	Temperature	Humidity	Windy	Class
Overcast	63	70	FALSE	Play
Rainy	73	90	TRUE	Don’t Play
Sunny	70	73	TRUE	Pay

2) Implement the program using DecisionTreeClassifier in scikit-learn library. The program requires 2 parameters:

- file name of trainset

- file name of testset

The program reports the classification results (accuracy, confusion matrix) for 5 datasets:

- Iris (.trn: trainset, .tst: testset)

```
D:/CT205H-ThầyNghi/data/iris/iris.trn
Accuracy: 0.94
Confusion Matrix:
[[17  0  0]
 [ 0 15  0]
 [ 0  3 15]]
```

- Optics (.trn: trainset, .tst: testset)

```
D:/CT205H-ThầyNghi/data/optics/opt.trn
Accuracy: 0.8514190317195326
Confusion Matrix:
[[172   0   1   0   2   1   1   0   1   0]
 [  0 155   5   4   1   1   1   2   7   6]
 [  1   9 142   4   3   0   2  10   4   2]
 [  0   2   8 146   0   5   0   6   8   8]
 [  4  10   3   0 144   1   5   2   9   3]
 [  1   5   3   1   1 160   6   1   0   4]
 [  0   5   1   0   3   0 169   0   3   0]
 [  0   3   3   0  12   3   0 139  11   8]
 [  2   7   4   2   2   4   0   1 146   6]
 [  0   3   2   6   4   4   0   0   4 157]]
```

- Letter (.trn: trainset, .tst: testset)

```
D:/CT205H-ThầyNghi/data/letter/let.trn
Accuracy: 0.8594359435943595
Confusion Matrix:
[[248   0   1   5   0   0   2   2   0   2   1   1   2   0   0   0   2   0
    0   1   2   2   0   0   2   1]
 [  0 201   0   6   1   3   2   1   1   4   2   1   0   0   2   2   0   4
    6   1   0   0   0   2   1   0]
 [  2   0 196   0  14   0   6   0   0   0   1   2   0   0   2   0   1   0
    0   1   0   1   0   0   0   0]
 [  0   2   0 250   0   1   0   5   0   2   0   0   1   2   5   1   0   4
    0   0   0   1   0   1   1   1]
 [  0   2  10   0 202   2  10   1   0   0   2   4   0   0   0   3   5   4
    6   1   0   0   0   6   0   4]
 [  1   4   3   4   1 226   0   1   1   0   1   0   1   0   0  14   0   1
    3   2   0   0   0   3   2   1]
 [  0   4   7   4   6   1 212   2   0   0   2   1   0   0   6   1   2   2
    1   2   0   3   3   0   0   4]
 [  1   6   0  11   2   2   4 174   1   0   7   1   0   1   0   2   1   6
```

- Leukemia (.trn: trainset, .tst: testset)

```
D:/CT205H-ThầyNghi/data/leukemia/ALLAML.trn
Accuracy: 0.9117647058823529
Confusion Matrix:
[[13  1]
 [ 2 18]]
```

- Fp (.trn: trainset, .tst: testset)

```
D:/CT205H-ThầyNghi/data/fp/fp.trn
Accuracy: 0.7875
Confusion Matrix:
[[28  0  0  0  0  0  0  0  0  1  0  0  0  0  0]
 [ 1  3  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  2  8  0  0  0  0  0  0  0  0  0  1  0  1]
 [ 0  0  0  7  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  8  0  0  0  0  0  0  0  0  0  1]
 [ 0  0  0  0  0 11  0  0  0  1  0  0  0  1  1]
 [ 0  0  0  0  0  0  8  0  0  0  0  0  0  1  1]
 [ 0  0  0  0  0  0  0  8  0  0  0  0  0  3  0]
 [ 1  0  0  0  0  0  0  0  6  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  3  0  0  1  0  0]
 [ 0  0  0  0  0  0  1  0  0  1  7  0  0  0  1]
 [ 0  0  0  0  0  0  0  0  0  0  0  7  0  0  2]
 [ 0  0  0  0  0  0  0  0  0  1  0  1  5  1  2]
 [ 0  0  0  0  0  0  2  0  0  0  0  1  0  7  0]
```

3) Why ensemble-based models improve the classification correctness of any single tree model?

Ensemble methods are ideal for reducing the variance in models, thereby increasing the accuracy of predictions. The variance is eliminated when multiple models are combined to form a single prediction that is chosen from all other possible predictions from the combined models