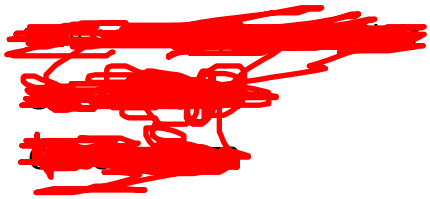

LAB 02: DECISION TREE

I. Tasks

No	Specification	Completion
1	Preparing the datasets	100%
2	Building the decision tree classifiers	100%
3	Evaluating the decision tree classifiers	100%
	Classification report and confusion matrix	100%
	Comments	100%
4	The depth and accuracy of a decision tree	100%
	Trees, tables, and charts	100%
	Comments	100%
Total		100%

II. Work

1) Preparing the datasets

- Step 1: I use panda library to read the input file

```
data = pd.read_csv('../Dataset/connect-4.data', header=None)
```

- Step 2: shuffle data following requirement

```
data = shuffle(data)
```

- Step 3: Split data into 2 parts: attribute (first 42 values) and classification (1 value – the last value in each line)

```
attribute = data.iloc[:, :-1]
classification = data.iloc[:, -1]
```

- Step 4: Replace all the attributes which are ‘x’, ‘o’, ‘b’ to 1, 2, 3 corresponding for the convenience of calculation.

```
attribute = attribute.replace({'x': 1, 'o': 2, 'b': 3})
```

- Step 5: Split the data into train and test sets with different proportions (40/60), (60/40), (80/20), (90/10)

```
train_test_proportion = [(0.4, 0.6), (0.6, 0.4), (0.8, 0.2), (0.9, 0
.1)]
```

- Step 6: Visualize the distributions of classes in all the data sets: please see the visualization of this section in folder **Image/Visual_Preparing**

2) Building the decision tree classifiers

- Step 1: Use function **train_test_split** to slit whole dataset becomes 4 subsets: feature_train, label_train, feature_ test, label_test
- Step 2: I use function **DecisionTreeClassifier** to create the decision tree and **set the criterion = 'entropy'** which gives the higher accuracy of final results. I don't use criterion = 'giny' since it gives me the accuracy less than entropy and Mrs.Thao said that we have not learned gini.
- Step 3: Use function **fit:** fits the decision tree to the training data.
- Step 4: Visualize the tree and save it in pdf file at **Image/Decision_Tree/dt_(x,y).pdf** where x, y are tranning and test corresponding.

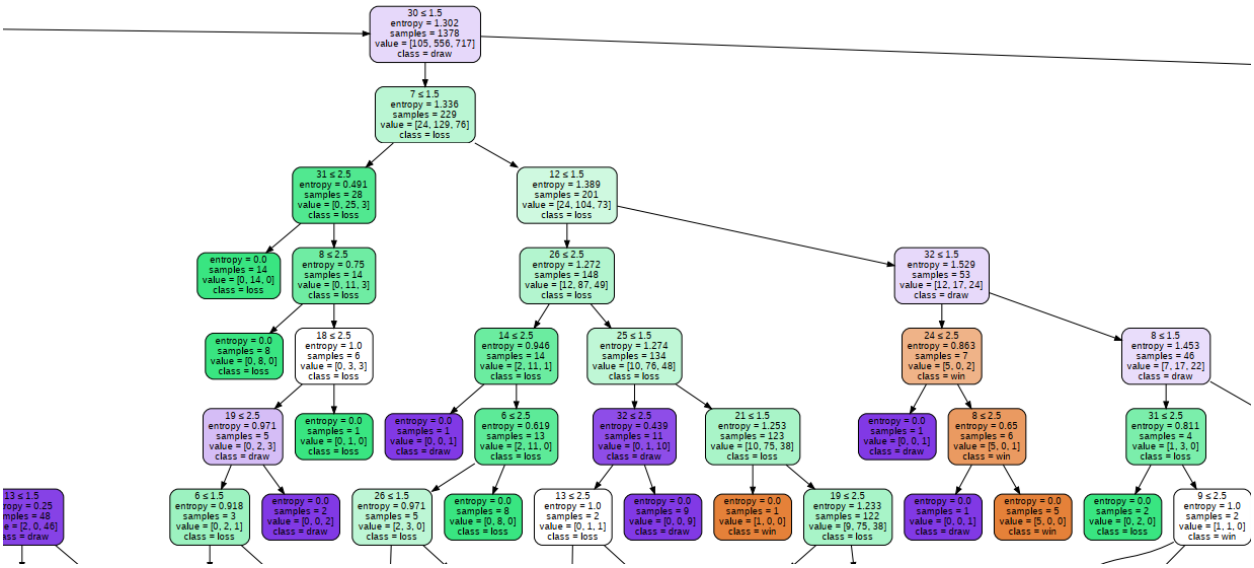

Figure 1: Example a small branch of decision tree with train/test 40/60

3) Evaluating the decision tree classifiers

- Step 1: use **predict** function: make the predictions on test data.
- Step 2: use function **classification_report** to print the report which has precision, recall, f1-score, support for each class in each proportion and also the accuracy.

- Step 3: print the confusion matrix to check the predicted class.
- Comments:

```
Train proportion: 40.0/60.0
              precision    recall  f1-score   support

        draw       0.21      0.23      0.22      3870
        loss       0.59      0.58      0.59      9981
         win       0.82      0.82      0.82     26684

    accuracy                           0.70     40535
   macro avg       0.54      0.54      0.54     40535
weighted avg       0.71      0.70      0.70     40535

[[  878  1129  1863]
 [ 1237  5790  2954]
 [ 1989  2866 21829]]
Complete proportion 40.0/60.0
Complete time: 1.4946982860565186 seconds
```

```
Train proportion: 60.0/40.0
              precision    recall  f1-score   support

        draw       0.24      0.24      0.24      2580
        loss       0.61      0.61      0.61      6654
         win       0.83      0.83      0.83     17789

    accuracy                           0.72     27023
   macro avg       0.56      0.56      0.56     27023
weighted avg       0.72      0.72      0.72     27023

[[  629   778  1173]
 [  818  4056  1780]
 [ 1211  1786 14792]]
Complete proportion 60.0/40.0
Complete time: 1.0796403884887695 seconds
```

```
Train proportion: 80.0/20.0
              precision    recall  f1-score   support

        draw       0.25      0.27      0.26      1290
        loss       0.63      0.61      0.62      3327
         win       0.84      0.84      0.84      8895

    accuracy                           0.73     13512
   macro avg       0.57      0.57      0.57     13512
weighted avg       0.73      0.73      0.73     13512

[[ 344  371  575]
 [ 435 2034  858]
 [ 612  834 7449]]
Complete proportion 80.0/20.0
Complete time: 0.8097066879272461 seconds
```

```
Train proportion: 90.0/10.0
              precision    recall  f1-score   support

        draw       0.29      0.29      0.29       645
        loss       0.65      0.64      0.65      1664
         win       0.84      0.85      0.85      4447

    accuracy                           0.74      6756
   macro avg       0.59      0.59      0.59      6756
weighted avg       0.74      0.74      0.74      6756

[[ 185  178  282]
 [ 184 1066  414]
 [ 280  385 3782]]
Complete proportion 90.0/10.0
Complete time: 0.6716187000274658 seconds
```

- In all cases of trainning/test, the best f1-score is label “win”. And the one less than is in label “loss”, the last one is “draw”.
- The major diagonal following top left to right bottom order corresponds to draw, loss, win.
- In 40/60:

        - About 77% of label “draw” is predicted to “loss” and “win”.
        - About 41% of label “loss” is predicted to “draw” and “win”.
        - About 18% of label “win” is predicted to “draw” and “loss”.
    - In 60/40:
        - About 75% of label “draw” is predicted to “loss” and “win”.
        - About 39% of label “loss” is predicted to “draw” and “win”.
        - About 16% of label “win” is predicted to “draw” and “win”.
    - In 80/20:
        - About 73% of label “draw” is predicted to “loss” and “win”.
        - About 38% of label “loss” is predicted to “draw” and “win”.
        - About 16% of label “win” is predicted to “draw” and “win”.
    - In 90/10:
        - About 70% of label “draw” is predicted to “loss” and “win”.
        - About 35% of label “loss” is predicted to “draw” and “win”.
        - About 14% of label “win” is predicted to “draw” and “win”.
    - Perfomance of the classifier depending on how the train and test data are split. When the train increases and test decreases, the classifier performs marginally better. With an increase in the percentage of test sets, the accuracy, precision, recall, and f1-score all show a little decline.

4) The depth and accuracy of a decision tree

- Step 1: Using the same implementation as 3) but now i set default train/test is (0.8, 0.2) (80/20) and limited the maximum depth: [None, 2 to 7]

Max_depth	None	2	3	4	5	6	7
Accuracy	73%	66%	66%	68%	68%	69%	70%

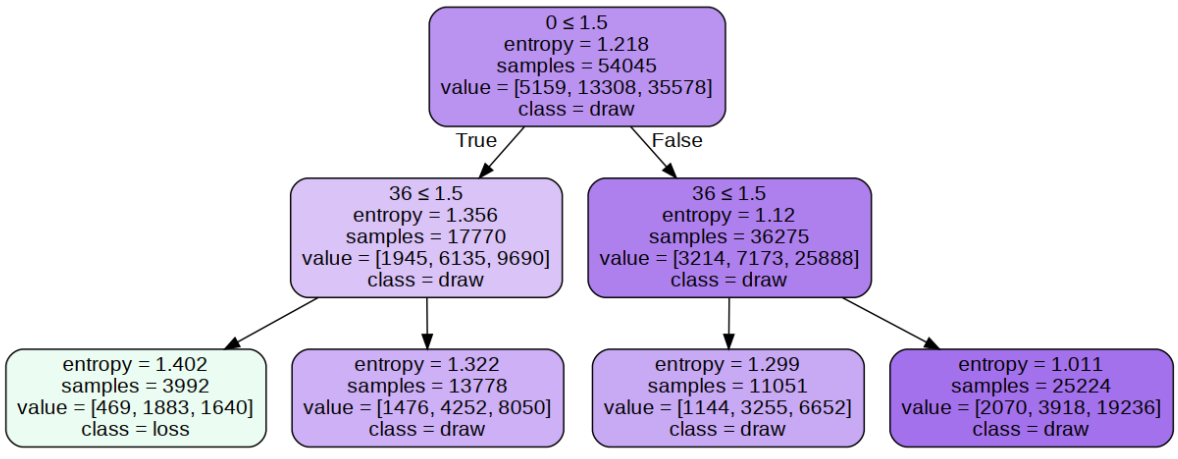

Figure 2: Example decision tree max_depth = 2

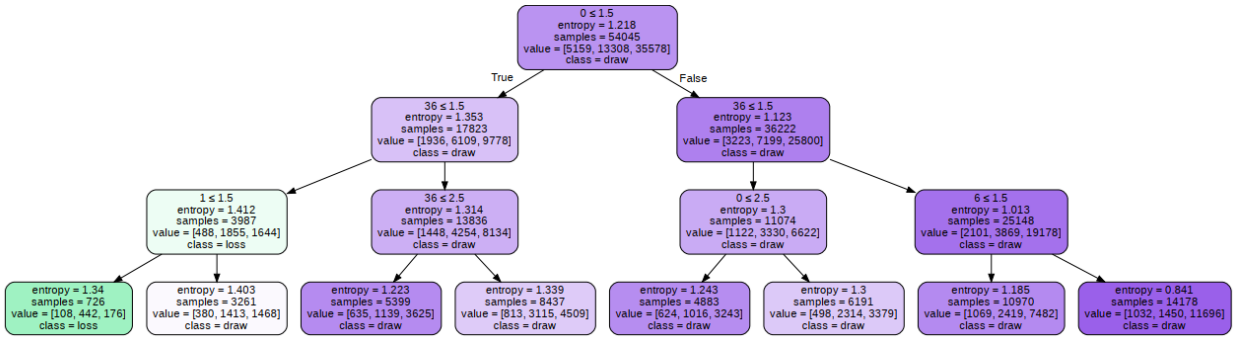

Figure 3: Example decision tree max_depth = 3

- Comments: In the decision tree without max depth, we can see that the accuracy is 73%. The accuracy is increased equivalent to the maximum depth of the decision tree. Therefore, we can conclude that the larger the depth, the higher the accuracy. You can view all the images in **Image/Decision_Tree/dt_depth_X.pdf** where X is the max_depth.