Patrick Tinsley
Data Science - HW2
Due Date: 2/20

Question 1: ID3 model, a decision tree model using “Information Gain”

1. Programming

   1.1. Use ID3 to construct a decision tree based on the training set (24 games).

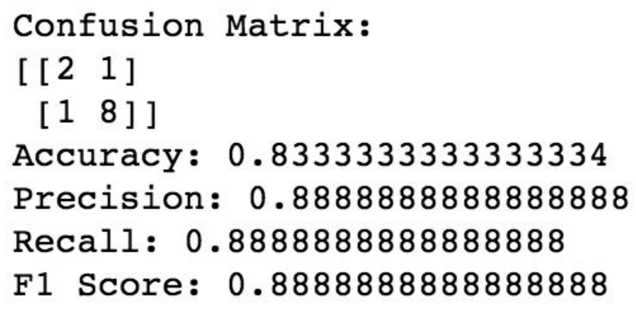
   See ptinsley-HW2-Q1.py for code.

   1.2. Use the tree to predict labels of instances in the testing set (12 games) based on their attributes.

```
predictions_id3 =
['Win','Win','Win','Lose','Win','Win','Win','Win','Win','Lose',
'Win','Lose']
```

   The above predictions were calculated by hand.

   1.3. Calculate Accuracy, Precision, Recall, and F1 score on the testing result.



```
Confusion Matrix:
[[2 1]
 [1 8]]
Accuracy: 0.8333333333333334
Precision: 0.8888888888888888
Recall: 0.8888888888888888
F1 Score: 0.8888888888888888
```

2. Attach a figure of your decision tree (either hand- or electronically drawn) and write down prediction label of the 12 testing games as well as evaluation result in the PDF.

   See appendix for decision tree. Note that the figure includes the tree as nodes distilled down to the decision basis; to see the full node contents, see the output from the script. As for the prediction labels of the testing games, see 1.2.

Question 2: C4.5 model, a decision tree model using “Gain Ratio”

3. Programming

   3.1. Use C4.5 to construct a decision tree based on the training set (24 games).

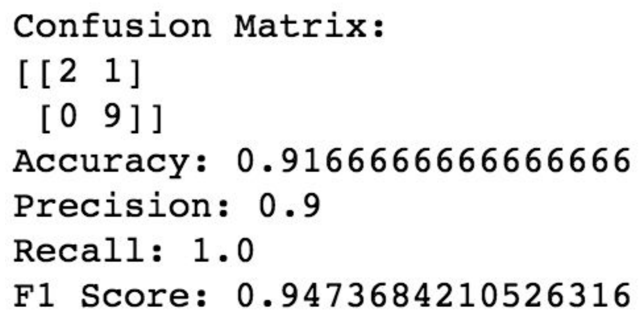
   See ptinsley-HW2-Q2.py for code.

   3.2. Use the tree to predict labels of instances in the testing set (12 games) based on their attributes.

```
predictions_c45 =
['Win','Win','Win','Win','Win','Win','Win','Win','Win','Lose','Wi
n','Lose']
```

   The above predictions were calculated by hand.

   3.3. Calculate Accuracy, Precision, Recall, and F1 score on the testing result.



```
Confusion Matrix:
[[2 1]
 [0 9]]
Accuracy: 0.9166666666666666
Precision: 0.9
Recall: 1.0
F1 Score: 0.9473684210526316
```

4. Attach a figure of your decision tree (either hand- or electronically drawn) and write down prediction label of the 12 testing games as well as evaluation result in the PDF.

   See appendix for decision tree. Note that the figure includes the tree as nodes distilled down to the decision basis; to see the full node contents, see the output from the script. As for the prediction labels of the testing games, see 3.2.

Question 3: Naïve Bayes model

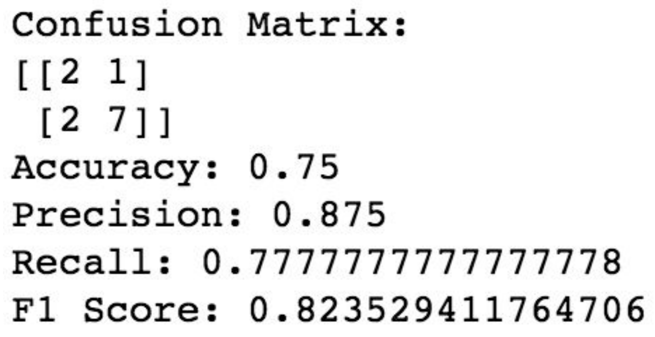
5. Programming

   5.1. Programming: Use Naïve Bayes to predict labels of instances in the testing set (12 games) based on the training set (24 games).

   See ptinsley-HW2-Q3.py for code.

   ```
   predictions_nb =
   ['Win','Win','Win','Lose','Win','Lose','Win','Win','Win','Lose',
   'Win','Lose']
   ```

   5.2. Calculate Accuracy, Precision, Recall, and F1 score on the testing result.



```
Confusion Matrix:
[[2 1]
 [2 7]]
Accuracy: 0.75
Precision: 0.875
Recall: 0.7777777777777778
F1 Score: 0.823529411764706
```

6. ~~Attach a figure of your decision tree (either hand- or electronically drawn) and~~ write down prediction label of the 12 testing games as well as evaluation result in the PDF.

   To see prediction labels for the testing games, see 5.1.

Question 4: For this Notre Dame game prediction task, which model is the best, which model performs the worst? Can you explain why? Write down in the PDF.

The best model for this prediction task is the decision tree, namely the C4.5 model; the worst model for the task is Naïve Bayes. Naïve Bayes performs more poorly because it inherently assumes conditional independence between features, which lends itself to lower accuracy. Additionally, given the small sample size for the training data, the decision trees do not overfit the data in this case and generalize well to the test data. The C4.5 model outperforms the ID3 model by addressing over-fitting with bottom-up pruning.

Appendix

1. Decision Tree (ID3)

```
# {
#     'Media = 1-NBC': {
#         'Is_Opponent_in_AP25_Preseason = In':
#             Win (by root node majority)
#         'Is_Opponent_in_AP25_Preseason = Out': {
#             'Is_Home_or_Away = Away':
#                 Win
#             'Is_Home_or_Away = Home':
#                 Win (by node majority)
#         }
#     },
#     'Media = 2-ESPN':
#         Win
#     'Media = 3-FOX':
#         Lose
#     'Media = 4-ABC': {
#         'Is_Opponent_in_AP25_Preseason = In':
#             Lose (node purity)
#         'Is_Opponent_in_AP25_Preseason = Out':
#             Win (by node majority)
#     },
#     'Media = 5-CBS':
#         Lose
# }
```

2. Decision Tree (C4.5)

```
# {
#     'Is_Opponent_in_AP25_Preseason = In': {
#         'Is_Home_or_Away = Away':
#             Lose (node purity)
#         'Is_Home_or_Away = Home':
#             Win (by root node majority)
#     },
#     'Is_Opponent_in_AP25_Preseason = Out': {
#         'Media = 1-NBC': {
#             'Is_Home_or_Away = Away':
#                 Win
#             'Is_Home_or_Away = Home':
#                 Win (by node majority)
#         },

###       'Media = 3-FOX':
###           Win (by root node majority)

#         'Media = 2-ESPN':
#             Win
#         'Media = 4-ABC':
#             Win (by node majority)
#         'Media = 5-CBS':
#             Lose
#     }
# }
```