Andrew Floyd
CS3001: Intro to Data Science
Dr. Fu
October 28th, 2018

Github repository w/ code: https://github.com/afzm4/cs3001hw4

(1) See repository for all of the files, here is a snippet of the code (full code in repository) along with the results requested:

```
gnb = GaussianNB()

used_features =[
        "Home_Away",
        "AP_25",
        "Media_m"]

gnb.fit(
        data_cleaned[used_features].values,
        data_cleaned["Label_m"])
y_pred = gnb.predict(test_cleaned[used_features])

y_true = np.array([0,1,0,0,0,0,0,0,0,1,0,1])

accu = accuracy_score(y_true, y_pred)
prec, recall, fscore, support = precision_recall_fscore_support(y_true, y_pred, average="binary")
```

```
Accuracy:  0.75
Precision:  0.5
Recall:  1.0
F1:  0.6666666666666666
```

(2) Here is the output of the prediction done by the Naive Bayesian code:

```
Where 0 = Win and 1 = Lose:
[0 1 0 1 0 1 1 0 0 1 0 1]
```

Compared with the actual results:

```
Actual:
Where 0 = Win and 1 = Lose:
[0 1 0 0 0 0 0 0 0 1 0 1]
```

(3) Here are the results of the ID3 with the training and testing data:

```
afzm4@rc09xcs213:~/SDRIVE/cs3001/hw4/cs3001hw4/decisiontrees$ python id3.py example_data/train.csv -t example_data/test.csv
decision tree for example_data/train.csv:
Dependent variable: Label_m
--None--(Media_m {'information_gain': 0.2886149872435303}, --1--(0 {}, ), --0--(Home_Away {'information_gain': 0.087753666780
7961}, --1--(0 {}, ), --0--(AP_25 {'information_gain': 0.05373467417506195}, --1--(0 {'estimated': True}, ), --0--(1 {'estima
ted': True}, ))), --3--(AP_25 {'information_gain': 0.17095059445466865}, --1--(Home_Away {'information_gain': 0.0}, --1--(1 {
'estimated': True}, ), --0--(1 {'estimated': True}, )), --0--(1 {}, )), --2--(1 {}, ), --4--(1 {}, ))
Rows: 24
Values: {'AP_25': set(['1', '0']), 'Label_m': set(['1', '0']), 'Media_m': set(['1', '0', '3', '2', '4']), 'Home_Away': set(['
1', '0'])}
Base Data Entropy: 0.979868756651
['0', '1', '0'] -> 0 (expected 0), CORRECT
['0', '0', '0'] -> 1 (expected 1), CORRECT
['1', '1', '1'] -> 0 (expected 0), CORRECT
['1', '1', '2'] -> 1 (expected 0), INCORRECT
['0', '1', '0'] -> 0 (expected 0), CORRECT
['1', '1', '3'] -> 1 (expected 0), INCORRECT
['0', '0', '0'] -> 1 (expected 0), INCORRECT
['0', '1', '0'] -> 0 (expected 0), CORRECT
['0', '1', '0'] -> 0 (expected 0), CORRECT
['1', '0', '3'] -> 1 (expected 1), CORRECT
['0', '1', '0'] -> 0 (expected 0), CORRECT
['1', '0', '3'] -> 1 (expected 1), CORRECT
% correct: 0.75
```

Here are the results of the C4.5 with the training and testing data:

```
Accuracy  0.700000
Took 0.007497 secs
```

We see from our results here and from the results above that the ID3 and the Naive Bayes model have the same accuracy (75%), while C4.5 has a little lower accuracy at 70%. This could be due to the fact that C4.5 uses error-based pruning, while ID3 does not. My version of C4.5 runs 10 times and then gives an average, so the results are well tested. We also know that ID3 uses information gain while C4.5 uses gain ratio, which might suggest that information gain may be more informative in this case. In summary, ID3 and Naive Bayes are the better models in this case, while C4.5 is the worst.