CS 5350/6350: Machine Learining Fall 2022

Homework 1

Handed out: 6 Sep, 2022
Due date: 11:59pm, 23 Sep, 2022

- You are welcome to talk to other members of the class about the homework. I am more concerned that you understand the underlying concepts. However, you should write down your own solution. Please keep the class collaboration policy in mind.
- Feel free discuss the homework with the instructor or the TAs.
- Your written solutions should be brief and clear. You need to show your work, not just the final answer, but you do *not* need to write it in gory detail. Your assignment should be **no more than 20 pages**. Not that you do not need to include the problem description. Every extra page will cost a point.
- Handwritten solutions will not be accepted.
- *Your code should run on the CADE machines.* You should include a shell script, `run.sh`, that will execute your code in the CADE environment. Your code should produce similar output to what you include in your report.

  You are responsible for ensuring that the grader can execute the code using only the included script. If you are using an esoteric programming language, you should make sure that its runtime is available on CADE.
- Please do not hand in binary files! We will *not* grade binary submissions.
- The homework is due by **midnight of the due date**. Please submit the homework on Canvas.
- Note the bonus questions are for **both 5350 and 6350** students. If a question is mandatory for 6350, we will highlight it explicitly.

1 Decision Tree [40 points + 10 bonus]

1. [7 points] Decision tree construction.

   (a) [5 points] Use the ID3 algorithm with information gain to learn a decision tree from the training dataset in Table 1. Please list every step in your tree construction, including the data subsets, the attributes, and how you calculate the information gain of each attribute and how you split the dataset according to the selected attribute. Please also give a full structure of the tree. You can manually draw the tree structure, convert the picture into a PDF/EPS/PNG/JPG format and include it in your homework submission; or instead, you can represent the tree with a conjunction of prediction rules as we discussed in the lecture.

x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0.	0
1	1	0	0	0
0	1	0	1	0

Table 1: Training data for a Boolean classifier

```
Current Subset of dataset
   x1  x2  x3  x4  y
0   0   0   1   0  0
1   0   1   0   0  0
2   0   0   1   1  1
3   1   0   0   1  1
4   0   1   1   0  0
5   1   1   0   0  0
6   0   1   0   1  0
Attribute: x1, Information Gain: 0.061743357932800724
Attribute: x2, Information Gain: 0.46956521111470695
Attribute: x3, Information Gain: 0.0059777114237739015
Attribute: x4, Information Gain: 0.46956521111470695
best Attribute: x2

Value: 0
Current Subset of dataset
   x1  x2  x3  x4  y
0   0   0   1   0  0
2   0   0   1   1  1
3   1   0   0   1  1
Attribute: x1, Information Gain: 0.2516291673878229
Attribute: x3, Information Gain: 0.2516291673878229
Attribute: x4, Information Gain: 0.9182958340544896
best Attribute: x4

Value: 0
Current Subset of dataset
   x1  x2  x3  x4  y
0   0   0   1   0  0
add leaf: 0

Value: 1
Current Subset of dataset
```

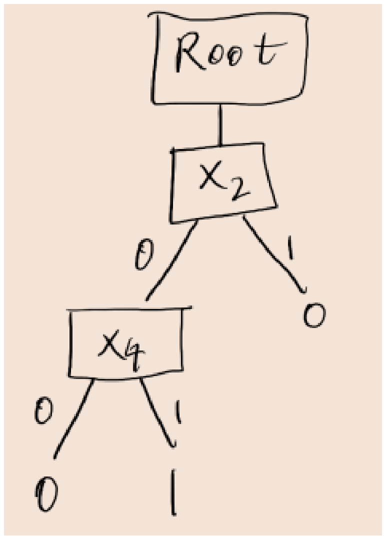
	x1	x2	x3	x4	y
2	0	0	1	1	1
3	1	0	0	1	1

add leaf: 1

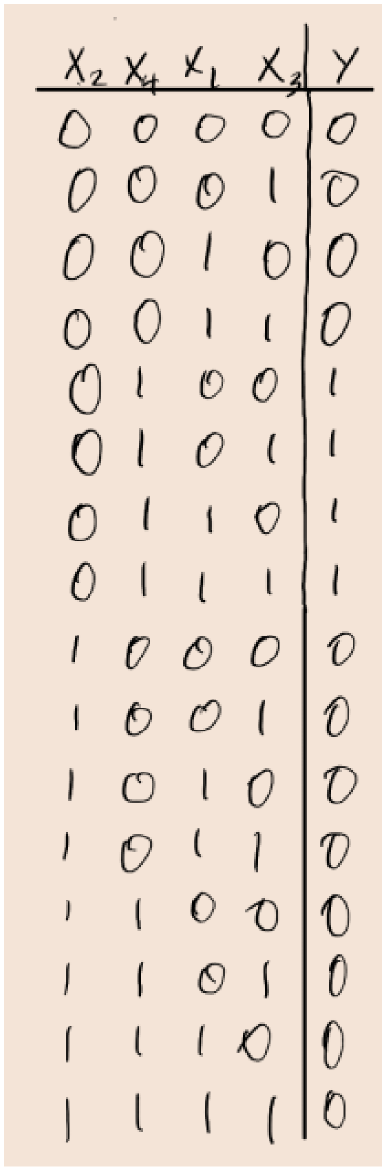
Attribute: x2
Value: 1
Current Subset of dataset

	x1	x2	x3	x4	y
1	0	1	0	0	0
4	0	1	1	0	0
5	1	1	0	0	0
6	0	1	0	1	0

add leaf: 0



(b) [2 points] Write the boolean function which your decision tree represents. Please use a table to describe the function — the columns are the input variables and label, i.e., x_1, x_2, x_3, x_4 and y; the rows are different input and function values.



X_2	X_4	X_1	X_3	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

2. [17 points] Let us use a training dataset to learn a decision tree about whether to play tennis (**Page 43, Lecture: Decision Tree Learning**, accessible by clicking the link http://www.cs.utah.edu/~zhe/teach/pdf/decision-trees-learning.pdf). In the class, we have shown how to use information gain to construct the tree in ID3 framework.

   (a) [7 points] Now, please use majority error (ME) to calculate the gain, and select the best feature to split the data in ID3 framework. As in problem 1, please list every step in your tree construction, the attributes, how you calculate the gain of each attribute and how you split the dataset according to the selected attribute. Please also give a full structure of the tree.

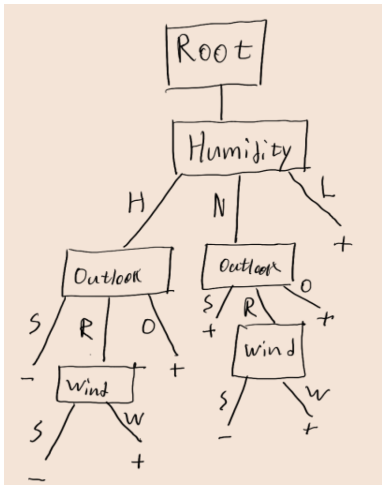
   Current Subset of dataset

```
    Outlook  Temperature  Humidity  Wind  Play
 0  S        H            H         W     -
 1  S        H            H         S     -
 2  O        H            H         W     +
 3  R        M            H         W     +
 4  R        C            N         W     +
 5  R        C            N         S     -
 6  O        C            N         S     +
 7  S        M            H         W     -
 8  S        C            N         W     +
 9  R        M            N         W     +
10  S        M            N         S     +
11  O        M            H         S     +
12  O        H            N         W     +
13  R        M            H         S     -
Attribute: Outlook, Gain: 0.0714285714285714
Attribute: Temperature, Gain: 5.551115123125783e-17
Attribute: Humidity, Gain: 0.07142857142857145
Attribute: Wind, Gain: 5.551115123125783e-17
best Attribute: Humidity

Value: H
Current Subset of dataset
    Outlook  Temperature  Humidity  Wind  Play
 0  S        H            H         W     -
 1  S        H            H         S     -
 2  O        H            H         W     +
 3  R        M            H         W     +
 7  S        M            H         W     -
11  O        M            H         S     +
13  R        M            H         S     -
Attribute: Outlook, Gain: 0.2857142857142857
Attribute: Temperature, Gain: 0.0
Attribute: Wind, Gain: 0.0
best Attribute: Outlook

Attribute: Outlook
Value: S
Current Subset of dataset
   Outlook  Temperature  Humidity  Wind  Play
0  S        H            H         W     -
1  S        H            H         S     -
7  S        M            H         W     -
add leaf: -
```

```
Value: R
Current Subset of dataset
     Outlook  Temperature  Humidity  Wind  Play
 3   R        M            H         W     +
 13  R        M            H         S     -
Attribute: Temperature, Information Gain: 0.0
Attribute: Wind, Information Gain: 0.5
best Attribute: Wind

Attribute: Wind
Value: S
Current Subset of dataset
     Outlook  Temperature  Humidity  Wind  Play
 13  R        M            H         S     -
add leaf: -

Attribute: Wind
Value: W
Current Subset of dataset
    Outlook  Temperature  Humidity  Wind  Play
 3  R        M            H         W     +
add leaf: +

Attribute: Outlook
Value: O
Current Subset of dataset
     Outlook  Temperature  Humidity  Wind  Play
 2   O        H            H         W     +
 11  O        M            H         S     +
add leaf: +

Value: N
Current Subset of dataset
     Outlook  Temperature  Humidity  Wind  Play
 4   R        C            N         W     +
 5   R        C            N         S     -
 6   O        C            N         S     +
 8   S        C            N         W     +
 9   R        M            N         W     +
 10  S        M            N         S     +
 12  O        H            N         W     +
Attribute: Outlook, Gain: 0.0
Attribute: Temperature, Gain: 0.0
Attribute: Wind, Gain: 0.0
best Attribute: Outlook
```

```
Attribute: Outlook
Value: S
Current Subset of dataset
     Outlook  Temperature  Humidity  Wind  Play
 8   S        C            N         W     +
 10  S        M            N         S     +
add leaf: +

Value: R
Current Subset of dataset
    Outlook  Temperature  Humidity  Wind  Play
 4  R        C            N         W     +
 5  R        C            N         S     -
 9  R        M            N         W     +
Attribute: Temperature, Gain: 0.0
Attribute: Wind, Gain: 0.3333333333333333
best Attribute: Wind

Attribute: Wind
Value: S
Current Subset of dataset
    Outlook  Temperature  Humidity  Wind  Play
 5  R        C            N         S     -
add leaf: -

Attribute: Wind
Value: W
Current Subset of dataset
    Outlook  Temperature  Humidity  Wind  Play
 4  R        C            N         W     +
 9  R        M            N         W     +
add leaf: +

Attribute: Outlook
Value: O
Current Subset of dataset
     Outlook  Temperature  Humidity  Wind  Play
 6   O        C            N         S     +
 12  O        H            N         W     +

add leaf: +

Attribute: Humidity
Value: L
No training data for Attribute: Humidity, value: L
```

most common label is +
add leaf: +



(b) [7 points] Please use gini index (GI) to calculate the gain, and conduct tree learning with ID3 framework. List every step and the tree structure.
Current Subset of dataset

	Outlook	Temperature	Humidity	Wind	Play
0	S	H	H	W	-
1	S	H	H	S	-
2	O	H	H	W	+
3	R	M	H	W	+
4	R	C	N	W	+
5	R	C	N	S	-
6	O	C	N	S	+
7	S	M	H	W	-
8	S	C	N	W	+
9	R	M	N	W	+
10	S	M	N	S	+
11	O	M	H	S	+
12	O	H	N	W	+
13	R	M	H	S	-

Attribute: Outlook, Gain: 0.11632653061224485
Attribute: Temperature, Gain: 0.018707482993197244
Attribute: Humidity, Gain: 0.09183673469387743
Attribute: Wind, Gain: 0.030612244897959162
best Attribute: Outlook

Value: S
Current Subset of dataset

```
     Outlook  Temperature  Humidity  Wind  Play
 0   S        H            H         W     -
 1   S        H            H         S     -
 7   S        M            H         W     -
 8   S        C            N         W     +
 10  S        M            N         S     +
Attribute: Temperature, Gain: 0.27999999999999997
Attribute: Humidity, Gain: 0.48
Attribute: Wind, Gain: 0.013333333333333308
best Attribute: Humidity

Attribute: Humidity
Value: H
Current Subset of dataset
    Outlook  Temperature  Humidity  Wind  Play
 0  S        H            H         W     -
 1  S        H            H         S     -
 7  S        M            H         W     -
add leaf: -

Attribute: Humidity
Value: N
Current Subset of dataset
     Outlook  Temperature  Humidity  Wind  Play
 8   S        C            N         W     +
 10  S        M            N         S     +
add leaf: +

Value: L
No training data for Attribute: Humidity, value: L
most common label is -
add leaf: -
Value: R
Current Subset of dataset
     Outlook  Temperature  Humidity  Wind  Play
 3   R        M            H         W     +
 4   R        C            N         W     +
 5   R        C            N         S     -
 9   R        M            N         W     +
 13  R        M            H         S     -
Attribute: Temperature, Gain: 0.013333333333333308
Attribute: Humidity, Gain: 0.013333333333333308
Attribute: Wind, Gain: 0.48
best Attribute: Wind
```

Attribute: Wind
Value: S
Current Subset of dataset

	Outlook	Temperature	Humidity	Wind	Play
5	R	C	N	S	-
13	R	M	H	S	-

add leaf: -

Attribute: Wind
Value: W
Current Subset of dataset

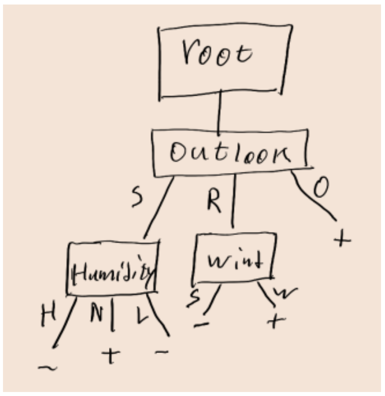
	Outlook	Temperature	Humidity	Wind	Play
3	R	M	H	W	+
4	R	C	N	W	+
9	R	M	N	W	+

add leaf: +

Attribute: Outlook
Value: O
Current Subset of dataset

	Outlook	Temperature	Humidity	Wind	Play
2	O	H	H	W	+
6	O	C	N	S	+
11	O	M	H	S	+
12	O	H	N	W	+

add leaf: +



(c) [3 points] Compare the two trees you just created with the one we built in the class (see Page 62 of the lecture slides). Are there any differences? Why?

Using Gini index creates the same tree that was created in class, however when using Majority error outlook and humidity had the same Gain, and humidity was

chosen, this caused a larger tree to be created and it also will classify the cases with Low humidity differently than the other two methods.
From this it looks like Gini index and Entropy perform better than Majority gain in terms of tree size.

3. [16 points] Continue with the same training data in Problem 2. Suppose before the tree construction, we receive one more training instance where Outlook's value is missing: {Outlook: Missing, Temperature: Mild, Humidity: Normal, Wind: Weak, Play: Yes}.

   (a) [3 points] Use the most common value in the training data as the missing value, and calculate the information gains of the four features. Note that if there is a tie for the most common value, you can choose any value in the tie. Indicate the best feature.

   Attribute: Outlook, Information Gain: 0.2273273022811375
   Attribute: Temperature, Information Gain: 0.032498735534292944
   Attribute: Humidity, Information Gain: 0.1686216675320541
   Attribute: Wind, Information Gain: 0.0597731301493174
   best Attribute: Outlook

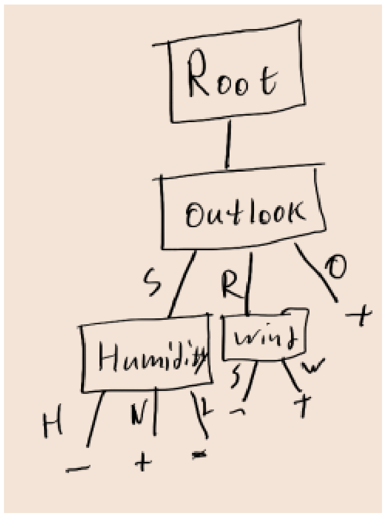
   (b) [3 points] Use the most common value among the training instances with the same label, namely, their attribute "Play" is "Yes", and calculate the information gains of the four features. Again if there is a tie, you can choose any value in the tie. Indicate the best feature.

   Attribute: Outlook, Information Gain: 0.27099543775137724
   Attribute: Temperature, Information Gain: 0.032498735534292944
   Attribute: Humidity, Information Gain: 0.1686216675320541
   Attribute: Wind, Information Gain: 0.0597731301493174
   best Attribute: Outlook

   (c) [3 points] Use the fractional counts to infer the feature values, and then calculate the information gains of the four features. Indicate the best feature.
   Attribute: Outlook, Information Gain: 0.2270013503178412
   Attribute: Temperature, Information Gain: 0.031953892783519455
   Attribute: Humidity, Information Gain: 0.16673012082019423
   Attribute: Wind, Information Gain: 0.058405459589777275
   best Attribute: Outlook

   (d) [7 points] Continue with the fractional examples, and build the whole free with information gain. List every step and the final tree structure.
   Attribute: Outlook, Information Gain: 0.2270013503178412

```
Attribute: Temperature, Information Gain: 0.031953892783519455
Attribute: Humidity, Information Gain: 0.16673012082019423
Attribute: Wind, Information Gain: 0.058405459589777275
best Attribute: Outlook

Value: S
Attribute: Temperature, Information Gain: 0.5583428983400098
Attribute: Humidity, Information Gain: 0.9878853530449179
Attribute: Wind, Information Gain: 0.0074016093567343955
best Attribute: Humidity

Attribute: Humidity
Value: H
add leaf: -

Attribute: Humidity
Value: N
add leaf: +

Value: L
No training data for Attribute: Humidity, value: L
most common label is -
add leaf: -

Attribute: Outlook
Value: R
Attribute: Temperature, Information Gain: 0.028099246073843176
Attribute: Humidity, Information Gain: 0.028099246073843176
Attribute: Wind, Information Gain: 0.9555111232924128
best Attribute: Wind

Attribute: Wind
Value: S
add leaf: -

Attribute: Wind
Value: W
add leaf: +

Attribute: Wind
Value: O
add leaf: +
```



4. [**Bonus question 1**] [5 points]. Prove that the information gain is always non-negative. That means, as long as we split the data, the purity will never get worse! (Hint: use convexity)

5. [**Bonus question 2**] [5 points]. We have discussed how to use decision tree for regression (i.e., predict numerical values) — on the leaf node, we simply use the average of the (numerical) labels as the prediction. Now, to construct a regression tree, can you invent a gain to select the best attribute to split data in ID3 framework?

2 Decision Tree Practice [60 points]

1. [5 Points] Starting from this assignment, we will build a light-weighted machine learning library. To this end, you will first need to create a code repository in Github.com. Please refer to the short introduction in the appendix and the official tutorial to create an account and repository. Please commit a README.md file in your repository, and write one sentence: "This is a machine learning library developed by **Your Name** for CS5350/6350 in University of Utah". You can now create a first folder, "DecisionTree". Please leave the link to your repository in the homework submission. We will check if you have successfully created it.

2. [30 points] We will implement a decision tree learning algorithm for car evaluation task. The dataset is from UCI repository(`https://archive.ics.uci.edu/ml/datasets/car+evaluation`). Please download the processed dataset (car.zip) from Canvas. In this task, we have 6 car attributes, and the label is the evaluation of the car. The attribute and label values are listed in the file "data-desc.txt". All the attributes are categorical. The training data are stored in the file "train.csv", consisting of $1,000$ examples. The test data are stored in "test.csv", and comprise 728 examples. In both

training and test datasets, attribute values are separated by commas; the file "data-desc.txt" lists the attribute names in each column.

Note: we highly recommend you to use Python for implementation, because it is very convenient to load the data and handle strings. For example, the following snippet reads the CSV file line by line and split the values of the attributes and the label into a list, "terms". You can also use "dictionary" to store the categorical attribute values. In the web are numerous tutorials and examples for Python. if you have issues, just google it!

```
with open(CSVfile, 'r') as f:
    for line in f:
        terms = line.strip().split(',')
        process one training example
```

(a) [15 points] Implement the ID3 algorithm that supports, information gain, majority error and gini index to select attributes for data splits. Besides, your ID3 should allow users to set the maximum tree depth. Note: you do not need to convert categorical attributes into binary ones and your tree can be wide here.

(b) [10 points] Use your implemented algorithm to learn decision trees from the training data. Vary the maximum tree depth from 1 to 6 — for each setting, run your algorithm to learn a decision tree, and use the tree to predict both the training and test examples. Note that if your tree cannot grow up to 6 levels, you can stop at the maximum level. Report in a table the average prediction errors on each dataset when you use information gain, majority error and gini index heuristics, respectively.

Test data

depth	Gini Index	Majority Error	Entropy
1	0.296703	0.296703	0.296703
2	0.222527	0.315934	0.222527
3	0.184066	0.310440	0.196429
4	0.137363	0.234890	0.151099
5	0.083791	0.134615	0.083791
6	0.083791	0.134615	0.083791

Train data

	Gini Index	Majority Error	Entropy
0	0.302	0.302	0.302
1	0.302	0.302	0.302
2	0.222	0.301	0.222
3	0.176	0.274	0.181
4	0.089	0.150	0.082
5	0.027	0.046	0.027
6	0.000	0.000	0.000

(c) [5 points] What can you conclude by comparing the training errors and the test errors?

From these results it looks like Gini Index and Entropy perform the best. On this data set the tree is able to perfectly model the training data set and the error of the test data decreases with each added depth. From this it looks like there is no over fitting as the error of the test data set does not increase. The depth of 6 does nothing to change the result of the test data set and only brings the error of the training data set down to 0.

3. [25 points] Next, modify your implementation a little bit to support numerical attributes. We will use a simple approach to convert a numerical feature to a binary one. We choose the media (NOT the average) of the attribute values (in the training set) as the threshold, and examine if the feature is bigger (or less) than the threshold. We will use another real dataset from UCI repository(`https://archive.ics.uci.edu/ml/datasets/Bank+Marketing`). This dataset contains 16 attributes, including both numerical and categorical ones. Please download the processed dataset from Canvas (bank.zip). The attribute and label values are listed in the file "data-desc.txt". The training set is the file "train.csv", consisting of $5,000$ examples, and the test "test.csv" with $5,000$ examples as well. In both training and test datasets, attribute values are separated by commas; the file "data-desc.txt" lists the attribute names in each column.

   (a) [10 points] Let us consider "unknown" as a particular attribute value, and hence we do not have any missing attributes for both training and test. Vary the maximum tree depth from 1 to 16 — for each setting, run your algorithm to learn a decision tree, and use the tree to predict both the training and test examples. Again, if your tree cannot grow up to 16 levels, stop at the maximum level. Report in a table the average prediction errors on each dataset when you use information gain, majority error and gini index heuristics, respectively.

Test error

depth	Gini Index	Majority Error	Entropy
1	0.1166	0.1166	0.1248
2	0.1088	0.1088	0.1114
3	0.1122	0.1132	0.1070
4	0.1194	0.1192	0.1144
5	0.1258	0.1250	0.1194
6	0.1364	0.1296	0.1284
7	0.1452	0.1374	0.1340
8	0.1506	0.1468	0.1392
9	0.1520	0.1536	0.1434
10	0.1548	0.1570	0.1480
11	0.1586	0.1608	0.1484
12	0.1584	0.1640	0.1504
13	0.1594	0.1656	0.1514
14	0.1594	0.1658	0.1514
15	0.1594	0.1658	0.1514
16	0.1594	0.1658	0.1514

Training error

depth	Gini Index	Majority Error	Entropy
1	0.1088	0.1088	0.1192
2	0.1042	0.1042	0.1060
3	0.0936	0.0964	0.1006
4	0.0754	0.0774	0.0800
5	0.0604	0.0652	0.0624
6	0.0484	0.0570	0.0480
7	0.0364	0.0468	0.0372
8	0.0268	0.0380	0.0292
9	0.0216	0.0312	0.0222
10	0.0174	0.0256	0.0182
11	0.0140	0.0210	0.0150
12	0.0136	0.0184	0.0136
13	0.0132	0.0154	0.0132
14	0.0132	0.0132	0.0132
15	0.0132	0.0132	0.0132
16	0.0132	0.0132	0.0132

(b) [10 points] Let us consider "unknown" as attribute value missing. Here we simply complete it with the majority of other values of the same attribute in the training set. Vary the maximum tree depth from 1 to 16 — for each setting, run your algorithm to learn a decision tree, and use the tree to predict both the training and test examples. Report in a table the average prediction errors on each dataset when you use information gain, majority error and gini index heuristics, respectively.

Test error

depth	Gini Index	Majority Error	Entropy
1	0.1166	0.1166	0.1248
2	0.1088	0.1088	0.1114
3	0.1122	0.1132	0.1070
4	0.1196	0.1184	0.1142
5	0.1266	0.1238	0.1206
6	0.1374	0.1312	0.1322
7	0.1458	0.1376	0.1392
8	0.1494	0.1464	0.1434
9	0.1514	0.1474	0.1470
10	0.1532	0.1532	0.1476
11	0.1542	0.1564	0.1486
12	0.1548	0.1594	0.1484
13	0.1548	0.1620	0.1494
14	0.1548	0.1614	0.1494
15	0.1548	0.1626	0.1494
16	0.1548	0.1626	0.1494

train error

depth	Gini Index	Majority Error	Entropy
1	0.1088	0.1088	0.1192
2	0.1042	0.1042	0.1060
3	0.0936	0.0964	0.1008
4	0.0770	0.0792	0.0814
5	0.0632	0.0676	0.0646
6	0.0522	0.0566	0.0510
7	0.0404	0.0476	0.0406
8	0.0318	0.0412	0.0330
9	0.0268	0.0362	0.0274
10	0.0214	0.0302	0.0218
11	0.0190	0.0266	0.0194
12	0.0184	0.0234	0.0188
13	0.0180	0.0210	0.0180
14	0.0180	0.0184	0.0180
15	0.0180	0.0180	0.0180
16	0.0180	0.0180	0.0180

(c) [5 points] What can you conclude by comparing the training errors and the test errors, with different tree depths, as well as different ways to deal with "unknown" attribute values?

It changing the unknown did not do much to change the results. In both cases the test error is minimum at a depth of 2, and then past this training error continues to decrease down to nearly 0. This indicates that the best model is at a depth of 2 and over fitting is occurring past a depth of 2.

a better way to deal with the unknown values might be to create a small tree with a subset of the training data and use it to predict the unknown values, then the tree could be trained on the full data set.