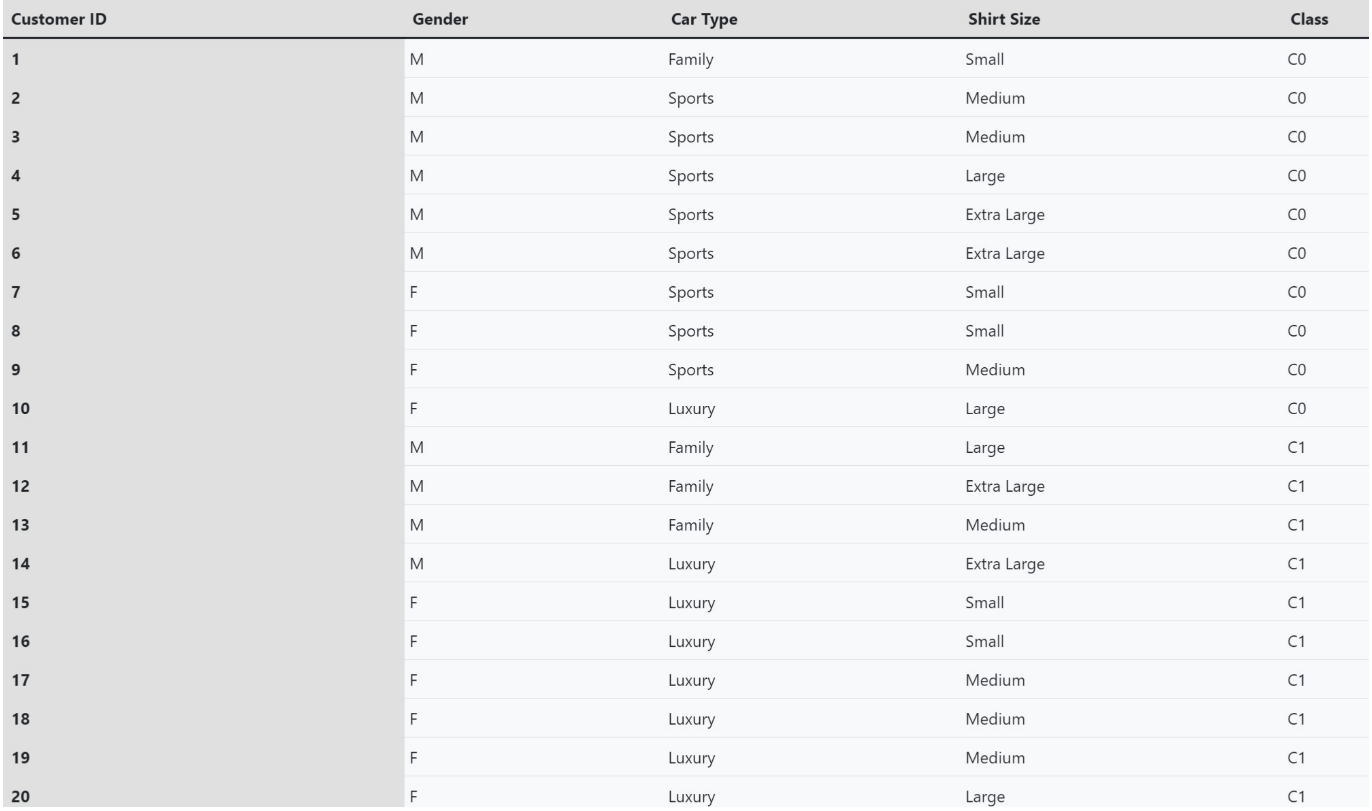
Data Mining CS 422 - Homework 4
Professor: Vijay Gurbani

Student name: George Tapia
Date: March 3, 2023

1. Consider the training examples shown in Table 3.5 (below) for a binary classification problem.



Customer ID	Gender	Car Type	Shirt Size	Class
1	M	Family	Small	C0
2	M	Sports	Medium	C0
3	M	Sports	Medium	C0
4	M	Sports	Large	C0
5	M	Sports	Extra Large	C0
6	M	Sports	Extra Large	C0
7	F	Sports	Small	C0
8	F	Sports	Small	C0
9	F	Sports	Medium	C0
10	F	Luxury	Large	C0
11	M	Family	Large	C1
12	M	Family	Extra Large	C1
13	M	Family	Medium	C1
14	M	Luxury	Extra Large	C1
15	F	Luxury	Small	C1
16	F	Luxury	Small	C1
17	F	Luxury	Medium	C1
18	F	Luxury	Medium	C1
19	F	Luxury	Medium	C1
20	F	Luxury	Large	C1

(a) Compute the Gini index for the Overall collection of training examples.
The Gini index measures the impurity of a set of examples. It can be defined as:

$$Gini(s) = 1 - \sum_{x=1}^{c} p_x(i|s)^2$$

Where s is the set of examples (total number of examples that belong to all classes), c is the number of classes, i is the number of examples in s that belong to class x. In this case, we have two classes, class C0, and class C1. Squaring the probabilities allows us to give more weight to the dominant class, and accurately depict the differences in probabilities.

Answer:

To compute the Gini index for the overall collection of training examples, we need to first calculate the proportion of examples that belong to each class. I counted 10 for each, C0 and C1 (total = 20):

$$p_{C0} = (\frac{10}{20})^2 = 0.25$$

$$p_{C1} = (\frac{10}{20})^2 = 0.25$$

Next, we can plug these proportions into the rest of the Gini index formula:

$$Gini(s) = 1 - [0.25 + 0.25] = 0.5$$

So the Gini index for the overall collection of training examples is 0.5.

(b) Compute the Gini index for Customer ID the attribute

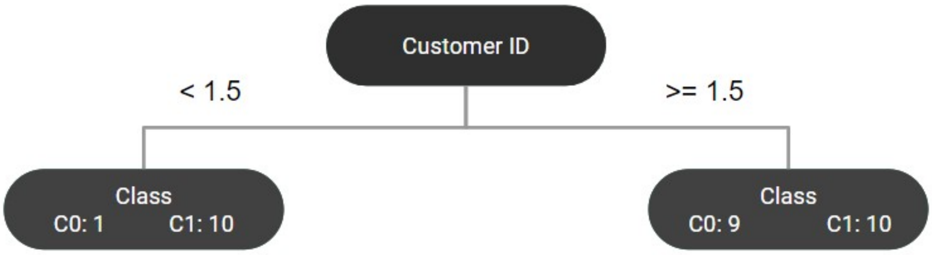
Answer:

To compute the Gini index for a numerical attribute, there are multiple routes we can take. This is one way. We can sort the training examples (here, they are sorted already) by the value of the attribute, then create n-1 groupings for every two adjacent examples.
Like so:

$$\begin{aligned}
&[1,2],[2,3],\\
&[3,4],[4,5],[5,6],\\
&[6,7],[7,8],[8,9],[9,10],\\
&[10,11],[11,12],[12,13],[13,14],[14,15]\\
&,[15,16],[16,17],[17,18],[18,19],[19,20]
\end{aligned}$$

Next, we can take the average of each of these groupings and create a test condition based on each average (the approach ultimately depends on you). For example, the first grouping has an average of 1.5 so we will use this as a test condition for our root node and consider how well it performs, in terms of labeling classes, by computing the Gini index.

Example:



Comparison symbols can be switched, depending on your preference.

$$p_1 = \frac{\text{number of examples in class } C_0}{\text{total number of examples on leaf n}}$$

$$p_2 = \frac{\text{number of examples in class } C_1}{\text{total number of examples on leaf n}}$$

Substituting these values into the Gini index formula, we get:

$$Gini(< 1.5) = 1 - (p_1^2 + p_2^2) = 1 - [(1/11)^2 + (10/11)^2] = 0.165$$
$$Gini(>= 1.5) = 1 - (p_1^2 + p_2^2) = 1 - [(9/19)^2 + (10/19)^2] = 0.499$$

Now, we can take the Gini index of each leaf to get our weighted average (total gini impurity resulting from splitting with test condition A as root node). Where L_n represents the gini index for the current leaf (n), and the number of leaves are represented by i.

Formula:

$$TotalGini(L, A) = \sum_{n=1}^{i} \left(\frac{\text{number of examples on leaf (n)}}{\text{total number of examples of all leaves}} \right) Gini(L_n)$$

Thus, we get (AKA Weighted Average):

Total Gini impurity for split with test condition A = (11/30)(0.165) + (19/30)(0.499) = 0.378

Since we got 0.378, this means that there is still a probability of new, random data being misclassified. We want to reduce this likelihood to zero.

We can continue testing the averages for all the other groupings but this will be very inefficient. Instead, we can take another approach. When we sort the data, we can look at which point the classifications change. In this case, we can see that there is a point at which all the C1 and C0 labels are split equally. We can take the average of these two values, and use it as a test condition (average = 10.5), saving us time from useless computations.

Thus we will get:

$$Gini(< 10.5) = 1 - (p_1^2 + p_2^2) = 1 - [(10/10)^2 + (0/10)^2] = 0$$

$$Gini(>= 10.5) = 1 - (p_1^2 + p_2^2) = 1 - [(0/10)^2 + (10/10)^2] = 0$$

Total Gini impurity for split with test condition A = 0

Summary:

Typically, in approach one, with numerical attributes, we will continue to try out all the averages of the adjacent groupings as test conditions and calculate the total impurity for each one. Then, we will pick the test condition that resulted in the lowest total impurity out of all the groupings. Though, since we spotted a change in classification before computing all of the total impurities for each average, we took a different approach, and conclude that the Gini index for the Customer ID attribute is 0 since it is as low as we can get.

(c) Compute the Gini index for Gender the attribute.

Answer:

$$Gini(Female) = 1 - (p_1^2 + p_2^2) = 1 - [(4/10)^2 + (6/10)^2] = 0.48$$
$$Gini(Male) = 1 - (p_1^2 + p_2^2) = 1 - [(9/19)^2 + (10/19)^2] = 0.48$$

$$TotalGini(\{Leaf = F, Leaf = M\}, Gender) = (10/20) * 0.48 + (10/20) * 0.48 = 0.48$$

(d) Compute the Gini index for Car Type the attribute using multiway split.

Answer:

$$
\begin{aligned}
Gini(Family) &= 1-(p_1^2+p_2^2) = 1-[(1/4)^2+(3/4)^2] = 0.375 \\
Gini(Sports) &= 1-(p_1^2+p_2^2) = 1-[(8/8)^2+(0/8)^2] = 0 \\
Gini(Luxury) &= 1-(p_1^2+p_2^2) = 1-[(1/8)^2+(7/8)^2] = 0.219
\end{aligned}
$$

$$
TotalGini(\{Leaf = Family, Leaf = Sports, Leaf = Luxury\}, Gender):
$$

$$
= (4/20)(0.375) + (8/20)(0) + (8/20)(0.219) = 0.163
$$

(e) Compute the Gini index for Shirt Size the attribute using multiway split.

Answer:

$$
\begin{aligned}
Gini(Small) &= 1-(p_1^2+p_2^2) = 1-[(3/5)^2+(2/5)^2] = 0.48 \\
Gini(Medium) &= 1-(p_1^2+p_2^2) = 1-[(3/7)^2+(4/7)^2] = 0.49 \\
Gini(Large) &= 1-(p_1^2+p_2^2) = 1-[(2/4)^2+(2/4)^2] = 0.5 \\
Gini(ExtraLarge) &= 1-(p_1^2+p_2^2) = 1-[(2/4)^2+(2/4)^2] = 0.5
\end{aligned}
$$

$$
TotalGini(\{Leaf = S, Leaf = M, Leaf = L, Leaf = XL\}, ShirtSize):
$$

$$
= (5/20)(0.48) + (7/20)(0.49) + (4/20)(0.5) + (4/20)(0.5) = 0.49
$$

(f) Which attribute is better, Gender, Car Type, or Shirt Size?

Answer:

The Car Type attribute is the better feature to split on. Since the gini index tells us how good a split is, based on some attribute. We can look at the values above to determine which attribute will give us the most accurate predictions. The lower the gini index, the better the decision tree will work on classifying data, and since Car Type has the lowest gini index, this means that this will give us the best result in terms of labeling data.

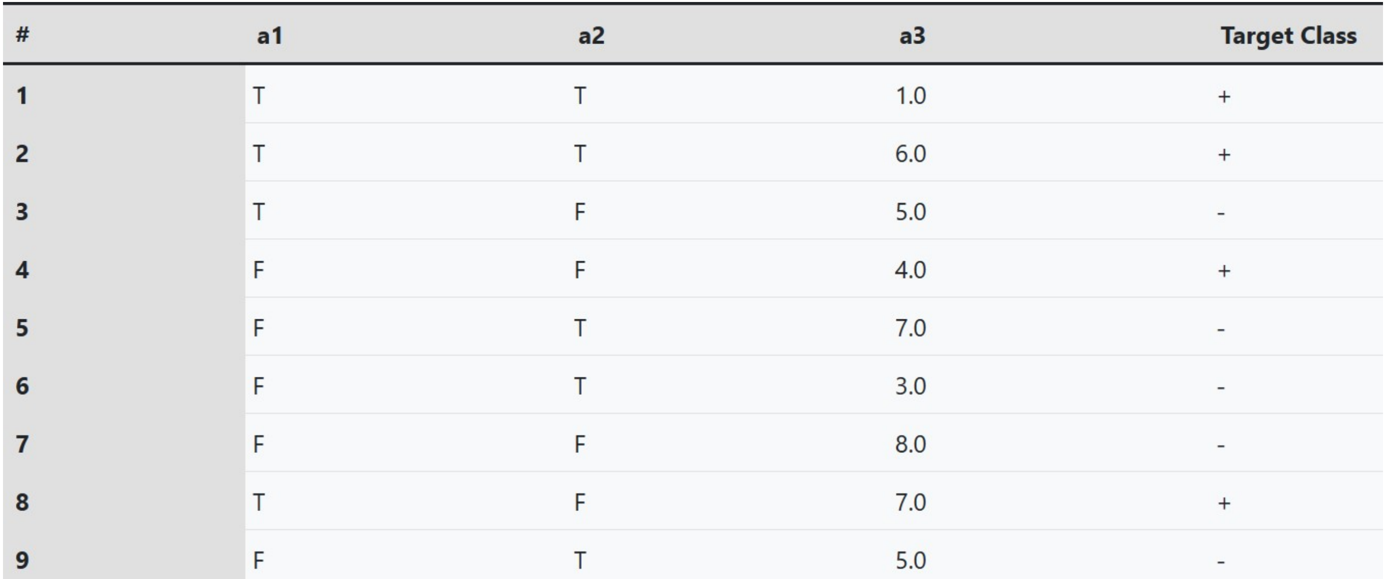
Just to give you an example of how exactly a low gini index relates to the performance of a decision tree, lets consider a model that predicts whether a student passes or fails. Suppose one of the attributes is GPA, and we notice that all the students with a GPA of 2.5, or greater, are passing their classes. If we took the gini index of GPA at 2.5, we will notice that this split will have a total gini index of zero since it will have two pure sets. One set will contain $\{fail, fail, fail\}$ for GPA < 2.5, the other set will contain $\{pass, pass, pass\}$ for GPA $>= 2.5$. Thus, we established a good boundary for splitting our data since this split is one of the crucial attributes that allows us to accurately label a student as passing or failing.

(g) Explain why Customer ID should not be used as the attribute test condition even though it has the lowest Gini.

Answer:

Customer ID is not a useful attribute for making predictions on new data because it is unique to each customer and does not provide any information about the customer's behavior or characteristics. In other words, Customer ID is not a feature that will allows us to gain insightful information for new customers, so it cannot be used to make predictions about their class.

2. Consider the training examples shown in Table 3.6 (below) for a binary classification problem.



#	a1	a2	a3	Target Class
1	T	T	1.0	+
2	T	T	6.0	+
3	T	F	5.0	-
4	F	F	4.0	+
5	F	T	7.0	-
6	F	T	3.0	-
7	F	F	8.0	-
8	T	F	7.0	+
9	F	T	5.0	-

(a) What is the entropy of this collection of training examples with respect to the class attribute?

Entropy is another measure of impurity that is commonly used in decision trees. It is defined as the sum of the probability of each class multiplied by the log of the probability of that class. The formula for entropy is as follows:

$$Entropy(s) = -\sum_{x=1}^{c} p_x(i|s)\log_2 p_x(i|s)$$

Alternative version, same as above:

$$= \sum_{x=1}^{c} p_x(i|s)\log_2(\left(\frac{1}{p_x(i|s)}\right))$$

Where s is the set of examples (total number of examples that belong to all classes), c is the number of classes, and i is the number of examples in s that belong to class x.

In this equation (alternative version), notice that we are using the log base 2 function. The reason for this is so that we can represent the amount of information gained in terms of bits (according to the sources listed below). Additionally, since the probability is inversely related to the amount of surprise, these relationships can be more accurately expressed through the function $\log_2(\left(\frac{1}{p_x(i|s)}\right))$. The lower the probability of an event P_x, the higher the surprise (more uncertainty). The greater the probability of an event, the lower the surprise (less uncertainty).

In relation to decision trees, when a set is pure, the probability of an instance belonging to a specific class in a set is $(1/(i/s) = 1)$. Meaning that $\log_2(1)$ equates to 0 entropy (no uncertainty). This means that the set has the maximum amount of information content, and a decision tree can make accurate predictions based on this information.

Answer:

$$Entropy(TargetClass) = [(4/9)\log_2(1/(4/9)) + (5/9)\log_2(1/(5/9))] = 0.991$$

(b) What are the information gains of a1 and a2 relative to these training examples?

Information gain is a metric used in decision trees to evaluate the usefulness of a feature in predicting the target variable. It measures the difference in entropy (or impurity) before and after splitting the data based on a particular feature.

The formula for information gain is:

$$InformationGain(S, A) = Entropy(S) - [\sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)]$$

- S: The set of examples being split.
- A: The feature being considered for splitting.
- S_v: The subset of examples of S for the current vertex v.
- $Values(A)$: The set of all possible vertices (AKA nodes) resulting from splitting with feature A.
- $|S_v|$: The number of examples in subset S_v.
- $|S|$: The total number of examples in set S.

Answer:

Entropy of a1 after split:

$$Entropy(a1(T)) = -[(3/4)\log_2(3/4) + (1/4)\log_2(1/4)] = 0.811$$

$$Entropy(a1(F)) = -[(1/5)\log_2(1/5) + (4/5)\log_2(4/5))] = 0.722$$

Entropy of a2 after split:

$$Entropy(a1(T)) = -[(2/5)\log_2(2/5) + (3/5)\log_2(3/5)] = 0.971$$

$$Entropy(a1(F)) = -[(2/4)\log_2(2/4) + (2/4)\log_2(2/4))] = 1$$

Information Gain (S represents the set before the split):

$$InformationGain(S, a1) = Entropy(S) - [\sum_{v \in Values(\{T,F\})} \frac{|S_v|}{|S|} Entropy(S_v)]$$

Before split:

$$Entropy(a1) = -[(4/9)\log_2(4/9) + (5/9)\log_2(5/9)] = 0.991$$

Information Gain:

$$\begin{aligned} InformationGain(S, a1) &= 0.991 - [(4/9)(0.811) + (5/9)(0.722)] \\ &= 0.991 - 0.722 \\ &= 0.762 \\ InformationGain(S, a2) &= 0.991 - [(5/9)(0.971) + (4/9)(1)] \\ &= 0.991 - 0.98388(rounding) \\ &= 0.006 \end{aligned}$$

(c) For a3, which is a continuous attribute, compute the information gain for every possible split.

There are multiple approaches for handling continuous attributes. Although the approaches do have one thing in common, we first have to sort the numerical values. Next, we can do either of the following:

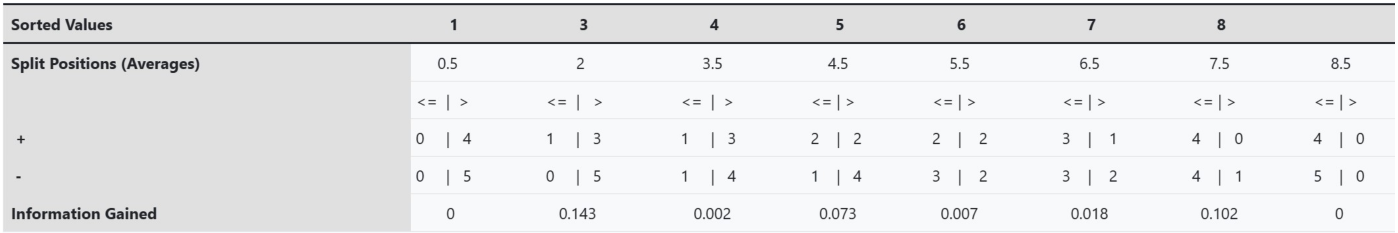
(1) Take the average of each adjacent grouping and use it as a test condition to split on. We can also add an additional test condition before the first average, and the last average.

(2) Don't take the average and try each value as a test condition.

(3) Look at the numerical values to see where each classification changes, and only take the average of each changing classification grouping. For example, if the class is constant from, 1 through n, and at n the class changes, then we take the average of (n and n+1) only. Saving us time from multiple computations.

Lastly, evaluate each test condition.

Answer:
I will take option one for this case.



Sorted Values	1	3	4	5	6	7	8	
Split Positions (Averages)	0.5	2	3.5	4.5	5.5	6.5	7.5	8.5
	<= | >	<= | >	<= | >	<= | >	<= | >	<= | >	<= | >	<= | >
+	0 | 4	1 | 3	1 | 3	2 | 2	2 | 2	3 | 1	4 | 0	4 | 0
-	0 | 5	0 | 5	1 | 4	1 | 4	3 | 2	3 | 2	4 | 1	5 | 0
Information Gained	0	0.143	0.002	0.073	0.007	0.018	0.102	0

Note: all computation values will be rounded to 3 significant digits at each step.
Split (0.5):

$$
\begin{aligned}
&Entropy(<=) : \text{Leaf does not exist, no computations made, ignore (zero)}\\
&Entropy(>) = -[(4/9)\log_2(4/9) + (5/9)\log_2(5/9)] = 0.991\\
&InformationGain(S, a1) = 0.991 - [(9/9)(0.991)] = 0
\end{aligned}
$$

Split (2):

$$
\begin{aligned}
&Entropy(<=) = -[(1/1)\log_2(1/1) + 0] = 0\\
&Entropy(>) = -[(4/9)\log_2(4/9) + (5/9)\log_2(5/9)] = 0.991\\
&InformationGain(S, a1) = 0.991 - [(8/9)(0.954)] = 0.143
\end{aligned}
$$

Split (3.5):

$$
\begin{aligned}
&Entropy(<=) = -[(1/2)\log_2(1/2) + (1/2)\log_2(1/2)] = 1\\
&Entropy(>) = -[(3/7)\log_2(3/7) + (4/7)\log_2(4/7)] = 0.985\\
&InformationGain(S, a1) = 0.991 - [(2/9)(1) + (7/9)(0.985)] = 0.003
\end{aligned}
$$

Split (4.5):

$$
\begin{aligned}
&Entropy(<=) = -[(2/3)\log_2(2/3) + (1/3)\log_2(1/3)] = 0.918\\
&Entropy(>) = -[(2/6)\log_2(2/6) + (4/6)\log_2(4/6)] = 0.918\\
&InformationGain(S, a1) = 0.991 - [(3/9)(0.918) + (6/9)(0.918)] = 0.073
\end{aligned}
$$

Split (5.5):

$$\begin{aligned}
Entropy(<=) &= -[(2/5)\log_2(2/5) + (3/5)\log_2(3/5)] = 0.971 \\
Entropy(>) &= -[(2/4)\log_2(2/4) + (2/4)\log_2(2/4)] = 1 \\
InformationGain(S, a1) &= 0.991 - [(5/9)(0.971) + (4/9)(1)] = 0.007
\end{aligned}$$

Split (6.5):

$$\begin{aligned}
Entropy(<=) &= -[(3/6)\log_2(3/6) + (3/6)\log_2(3/6)] = 1 \\
Entropy(>) &= -[(1/3)\log_2(1/3) + (2/3)\log_2(2/3)] = 0.918 \\
InformationGain(S, a1) &= 0.991 - [(6/9)(1) + (3/9)(0.918)] = 0.018
\end{aligned}$$

Split (7.5):

$$\begin{aligned}
Entropy(<=) &= -[(4/8)\log_2(4/8) + (4/8)\log_2(4/8)] = 1 \\
Entropy(>) &= -[(1/1)\log_2(1/1)] = 0 \\
InformationGain(S, a1) &= 0.991 - [(8/9)(1)] = 0.102
\end{aligned}$$

Split (8.5):

$$\begin{aligned}
&Entropy(<=) = -[(4/9)\log_2(4/9) + (5/9)\log_2(5/9)] = 0.991 \\
&Entropy(>) : \text{Leaf does not exist, no computations made, ignore (zero)} \\
&InformationGain(S, a1) = 0.991 - [(9/9)(0.991)] = 0
\end{aligned}$$

(d) What is the best split (among a1 , a2 and a3) according to the information gain?

Answer:

Based on the information gain calculations, it appears that attribute a1 is the most informative attribute for making classifications among the three attributes considered in this problem. This is because a1 resulted in the highest information gain, indicating that it reduces the uncertainty or entropy of the dataset the most. Thus, a1 will be the best attribute to split on.

(e) What is the best split (between a1 and a2) according to the misclassification error rate?

Answer:

Classification-Error(attribute) = (number of wrong predictions)/ total number of predictions

$$\begin{aligned}
Classification - Error(a1) &= 2/9 \\
&= 0.222
\end{aligned}$$

$$\begin{aligned}
Classification - Error(a2) &= 4/9 \\
&= 0.444
\end{aligned}$$

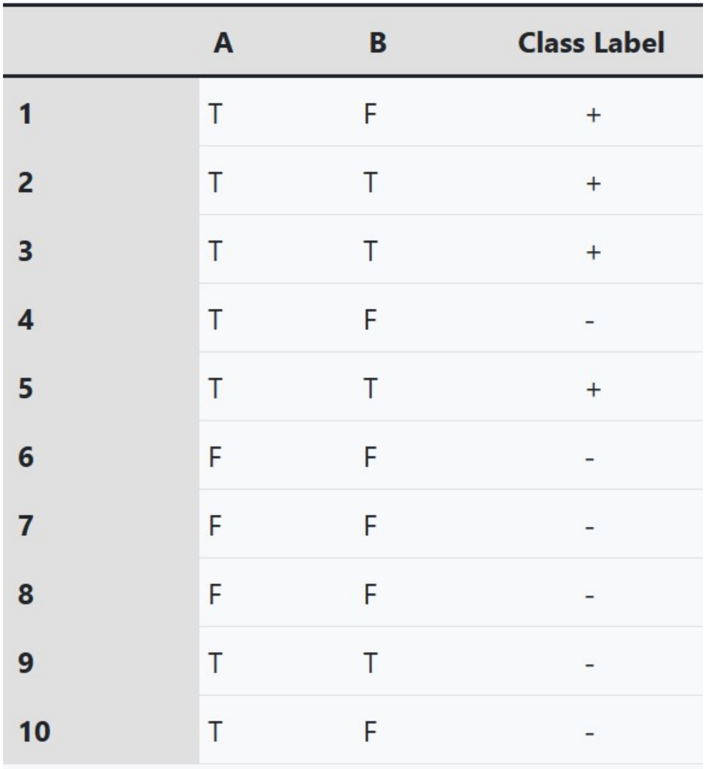
(f) What is the best split (between a1 and a2) according to the Gini index?

Answer:
According to the Gini index, a1 has the best split since it has a lower gini index. This indicates that the split on attribute a1 is purer than a2

$$Gini(a1) = 1 - (p_1^2 + p_2^2) = 1 - [(7/9)^2 + (2/9)^2] = 0.346$$

$$Gini(a2)) = 1 - (p_1^2 + p_2^2) = 1 - [(4/9)^2 + (5/9)^2] = 0.494$$

3. Consider the following data set for a binary class problem.



	A	B	Class Label
1	T	F	+
2	T	T	+
3	T	T	+
4	T	F	-
5	T	T	+
6	F	F	-
7	F	F	-
8	F	F	-
9	T	T	-
10	T	F	-

(a) Calculate the information gain when splitting on A and B. Which attribute would the decision tree induction algorithm choose?

Answer:
Before split:

$$Entropy(Class - Label) = -[(4/10)\log_2(4/10) + (6/10)\log_2(6/10)] = 0.971$$

Split A:

$$Entropy(A(T)) = -[(4/7)\log_2(4/7) + (3/7)\log_2(3/7)] = 0.985$$
$$Entropy(A(F)) = -[(3/3)\log_2(3/3)] = 0$$
$$InformationGain(S, A) = 0.918 - [(7/10)(0.918) + (3/10)(0)] = 0.281$$

Split B:

$$Entropy(B(T)) = -[(3/4)\log_2(3/4) + (1/4)\log_2(1/4)] = 0.811$$
$$Entropy(B(F)) = -[(1/6)\log_2(5/6) + (1/6)\log_2(1/6)] = 0.65$$
$$InformationGain(S, B) = 0.918 - [(4/10)(0.811) + (6/10)(0.65)] = 0.257$$

Thus, attribute A should be chosen since it resulted in the highest information gain.

(b) Calculate the gain in the Gini index when splitting on A and B. Which attribute would the decision tree induction algorithm choose?

Answer:
Before split:

$$Gini(Class - Label) = 1 - (p_1^2 + p_2^2) = 1 - [(4/10)^2 + (6/10)^2] = 0.48$$

Split A:

$$Gini(A(T)) = 1 - (p_1^2 + p_2^2) = 1 - [(4/7)^2 + (3/7)^2] = 0.49$$
$$Gini(A(F)) = 1 - (p_1^2 + p_2^2) = 1 - [(3/3)^2 + (0/3)^2] = 0$$

$$InformationGain(S, A) = 0.48 - [(7/10)(0.49) + (3/10)(0)] = 0.137$$

Split B:

$$Gini(B(T)) = 1 - (p_1^2 + p_2^2) = 1 - [(1/4)^2 + (3/4)^2] = 0.38$$
$$Gini(B(F)) = 1 - (p_1^2 + p_2^2) = 1 - [(1/6)^2 + (5/6)^2] = 0.278$$

$$InformationGain(S, B) = 0.48 - [(4/10)(0.38) + (6/10)(0)] = 0.163$$

Thus, attribute B should be chosen since it resulted in the highest information gain.

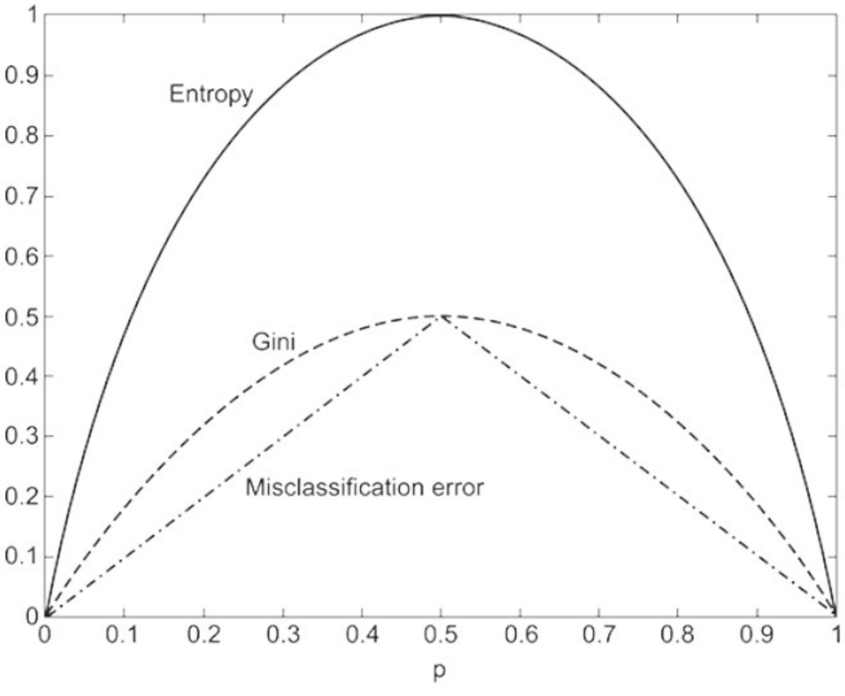


Figure 3.11.

(c) Figure 3.11 shows that entropy and the Gini index are both monotonically increasing on the range [0, 0.5] and they are both monotonically decreasing on the range [0.5, 1]. Is it possible that information gain and the gain in the Gini index favor different attributes? Explain.

Answer:
Yes, it's possible for the information gain, and the gain in the gini index to favor different attributes, this type of behavior was shown in part a and b above. An attribute that undergoes a high reduction in entropy does not necessarily mean that it will result in the purest subsets according to the gini index, vice versa.

References:

(1) Introduction to Data Mining-2nd-edition, By: Pang-Ning Tan, Michael Steinbach, Vipin Kumar - Addison Wesley, 2005, 0321321367

(2) https://www.youtube.com/watch?v=YtebGVx-Fxwt=351s (stats quest)

(3) https://towardsdatascience.com/the-intuition-behind-shannons-entropy-e74820fe9800