National University of Singapore
School of Computing
CS3244: Machine Learning
Solution to Tutorial 2

Decision Trees and Ensemble Methods

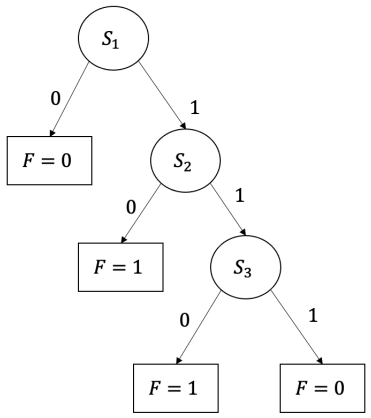
1. **Introduction to Decision Trees.**

   The data in table 1 represents the three states (S_1, S_2, S_3) which contribute to the lighting of a bulb (the final state F). Each state takes value from the set $\{0, 1\}$.

S_1	S_2	S_3	F
0	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

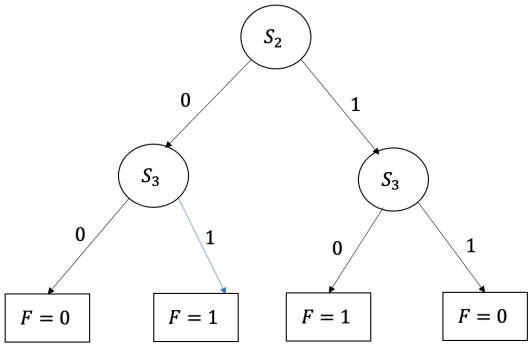
Table 1: States and the final outcome

(a) Construct a decision tree to classify the final outcome (F) from the three initial states S_1, S_2 and S_3. Follow a greedy way to construct a decision tree with feature order S_1, S_2 and S_3 which can fit the dataset with 0 training error.



(b) Comment on the tree from part (a). Is the tree optimal? If it is not optimal construct an optimal decision tree. (Here optimality is decided from the depth of a DT.)

F is just function which can be represented as $S_2 \bigoplus S_3$. Using this we can build an optimal tree with depth 2 as shown below.



(c) Alice tries to implement a XOR function which has d inputs using decision tree (DT). Why using DT is not a scalable solution? Explain your answer. If we implement AND or OR function with d inputs, do we get any advantage over the XOR function?

To implement the given XOR using DTs, we need to have 2^d leaf nodes and $2^d - 1$ internal nodes. The required space grows exponentially with d. Hence, this is not a scalable solution. Moreover, to calculate the final outcome of the XOR function, every input/feature needs to be considered. Hence, pruning is also not possible. Implementation of AND or OR function gets the advantage of pruning. For example, if a particular variable is false in AND function, then the outcome is always false. Similarly, if a particular variable is true in OR function, then the outcome is always true. Think about how many DT internal nodes are needed to model the AND function.

2. **Bank on Decision Trees.**

The loans department of DBN (Development Bank of NUS) has the following past loan processing records each containing an applicant's income, credit history, debt, and the final approval decision. Details are shown in Table 2.

Income	Credit History	Debt	Decision
0 – 5K	Bad	Low	Reject
0 – 5K	Good	Low	Approve
0 – 5K	Unknown	High	Reject
0 – 5K	Unknown	Low	Approve
0 – 5K	Unknown	Low	Approve
0 – 5K	Unknown	Low	Reject
5 – 10K	Bad	High	Reject
5 – 10K	Good	High	Approve
5 – 10K	Unknown	High	Approve
5 – 10K	Unknown	Low	Approve
Over 10K	Bad	Low	Reject
Over 10K	Good	Low	Approve

Table 2: Loan processing records

(a) Construct a decision tree based on the above training examples. (Note: $\log_2 \frac{x}{y} = \log_2$ x - $\log_2$ y, $\log_2$ 1 = 0, $\log_2$ 2 = 1, $\log_2$ 3 = 1.585, $\log_2$ 4 = 2, $\log_2$ 5 = 2.322, $\log_2$ 6 = 2.585, $\log_2$ 7 = 2.807, $\log_2$ 8 = 3, $\log_2$ 9 = 3.170, $\log_2$ 10 = 3.322, $\log_2$ 11 = 3.459, and $\log_2$ 12 = 3.585)

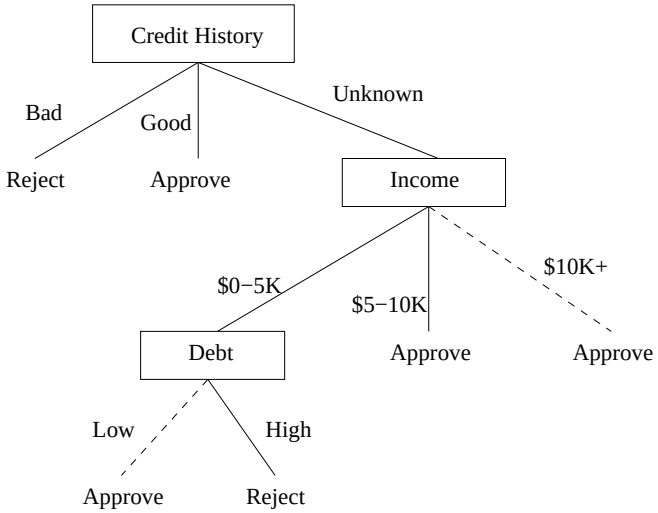
$$
\begin{aligned}
IG(Income) &= H(\frac{5}{12}, \frac{7}{12}) - remainder(Income) \\
remainder(Income) &= \frac{6}{12}(-\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6}) + \frac{4}{12}(-\frac{1}{4}\log_2\frac{1}{4} \\
&\quad -\frac{3}{4}\log_2\frac{3}{4}) + \frac{2}{12}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}) \\
&= 0.937 \\
IG(Income) &= 0.98 - 0.937 = 0.043 \\
remainder(CreditHistory) &= \frac{3}{12}(-\frac{3}{3}\log_2\frac{3}{3} - \frac{0}{3}\log_2\frac{0}{3}) + \frac{3}{12}(-\frac{3}{3}\log_2\frac{3}{3} \\
&\quad -\frac{0}{3}\log_2\frac{0}{3}) + \frac{6}{12}(-\frac{2}{6}\log_2\frac{2}{6} - \frac{4}{6}\log_2\frac{4}{6}) \\
&= 0.459 \\
IG(CreditHistory) &= 0.98 - 0.459 = 0.521 \\
remainder(Debt) &= \frac{8}{12}(-\frac{3}{8}\log_2\frac{3}{8} - \frac{5}{8}\log_2\frac{5}{8}) + \frac{4}{12}(-\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4}) \\
&= 0.970 \\
IG(Debt) &= 0.98 - 0.970 = 0.01
\end{aligned}
$$

Since Credit History has the highest gain, choose it as the root, which has three values, i.e., "Bad", "Good", and "Unknown". Since all examples for "Bad" have the same classification (i.e., "Reject") and all examples for "Good" have the same classification (i.e., "Approve"), both nodes have no further subtree. For "Unknown", a subtree for the following subset of examples is to be constructed:

Income	*Debt*	*Decision*
$0-5K$	*High*	*Reject*
$0-5K$	*Low*	*Approve*
$0-5K$	*Low*	*Approve*
$0-5K$	*Low*	*Reject*
$5-10K$	*High*	*Approve*
$5-10K$	*Low*	*Approve*

$$
\begin{aligned}
H(\frac{2}{6}, \frac{4}{6}) &= -\frac{2}{6}\log_2\frac{2}{6} - \frac{4}{6}\log_2\frac{4}{6} = 0.918 \\
remainder(Income) &= \frac{4}{6}(-\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4}) + \frac{2}{6}(-\frac{2}{2}\log_2\frac{2}{2} - \frac{0}{2}\log_2\frac{0}{2}) = 0.667 \\
gain(Income) &= 0.918 - 0.667 = 0.251 \\
remainder(Debt) &= \frac{2}{6}(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}) + \frac{4}{6}(-\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4}) = 0.874 \\
gain(Debt) &= 0.918 - 0.874 = 0.044
\end{aligned}
$$

Since Income has a higher gain than Debt, Income is chosen as the root of the subtree under Credit History=Unknown.



The two decisions on dotted lines are those that are not quite so straight forward. In one case, there is insufficient data for us to decide what to do, so we do something reasonable. In the other case, the resulting sample is not decisive. There are two approves and one reject. We can use a simple majority to decide.
Question is: why do we have such a node where there seems to be ambiguity? There is no clear answer (or at least we don't have the information to determine). One possibility is that we are missing some attribute from the data that will allow us to differentiate at this last node. Another possibility is that there is some non-determinism in the underlying function that we are trying to approximate with this decision tree.

(b) Construct 3 different DTs, where each of the three DTs is fully grown from two of the three attributes, again based on the same set of examples: {Income, Credit History}, {Credit History, Debt} and {Debt, Income}.

Based on the earlier answers for the decision tree, we know that Credit History will be the root node as it has a better information gain. This leads to pure "Bad" and "Good" leaves, leading to only using the secondary attribute for the remaining six cases of "Unknown".
The {Income, Credit History} tree is identical to the one for the full decision tree, except that the bottom Debt node is left off and replaced by the mode of the examples that fulfill the tests (Credit History == "Unknown" ∩ Income == "0 – 5K"). This is a tie, so the parent node's test of just Credit History == "Unknown" is used to obtain "Approve".
For the {Credit History, Debt} tree, the Credit History == "Unknown" takes a Debt test, resulting "Approve" for "Low" values (by the mode of examples), and also "Approve" for "High" values (by the mode of the parent, since the mode of the examples is tied).
For the {Debt, Income} tree, we can recall that the information gain of the previous decision tree showed that Income was better than Debt. All three subcases of Income are not pure, and each needs a test for Debt. In the "0 – 5K" Income, Debt == "Low" yields "Approve" by the mode of the examples, and yields "Reject" for "High" Debt cases. In the "5 – 10K" Income, Debt == "High" yields "Approve" by the mode of the examples, and yields "Approve" by purity when Debt is "Low". In the "Over 10K" Income, there are no examples of "High", so it defaults to "Approve". Similarly, since there is a tied outcome for Debt == "Low", we

default to the mode of all examples, hence "Approve".

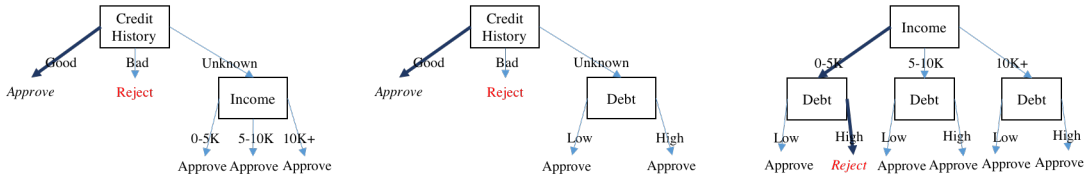


Figure 1: 3 DTs in Part (b). The dark edges and italicized labels correspond to the decision for each tree for the individual described in Part (c).

(c) What is the DT classifier's (part (a)) decision for a person who has 4K yearly income, a good credit history and a high amount of debt? Is your result different if we use 3 DTs in part (b) to make a decision?

For the individual, the DT from part (a) approves the application from the root node of the tree.
As done in Part (b), the first two trees vote "Approve", but the final {Income, Debt} tree votes "Reject" (as it satisfies the Income ==" 0 – 5K" → Debt == "High" → "Reject" path. If we employ uniform voting from all 3 DTs, so the $\frac{2}{3}$rds vote wins to "Approve" the applicant too.
(Optional) How could the decisions (possibly different) given by the 3 DTs be collated together?

3. **Discretizing Continuous Features.**

Bob suspects the passion a student has for Machine Learning (represented by a real number) and his midterm grade contribute to the student's final grade for CS3244. He collected some data, as shown in table 3. Construct a decision tree based on the given graph 2 and table 3. You need to discretize both the MidtermGrade and PassionForML features. (Hint: Discretize MidtermGrade first.)

Midterm Grade	Passion For ML	Final Grade Is A
5	1.5	T
6	-4	T
7.5	0	F
8	-3	F
8	3.5	T
9	2	F
12	1.2	F
12	4	T
13	-2	T
14	-1	T

Table 3: Students' Information

He then plotted these information in a graph:

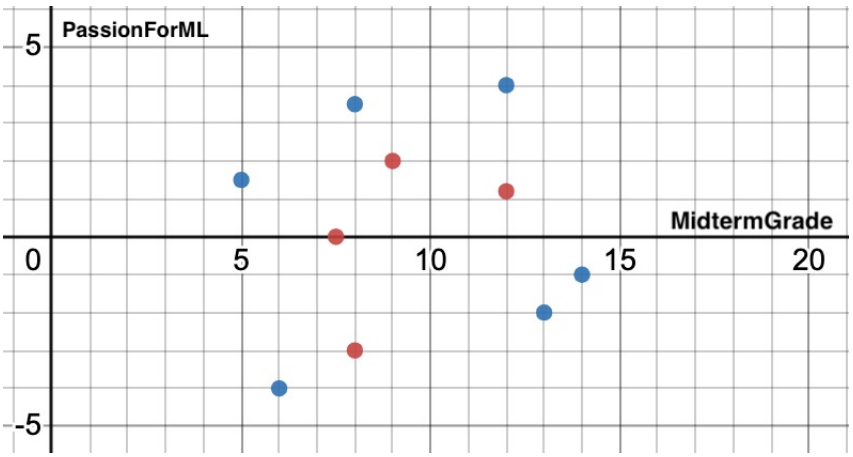
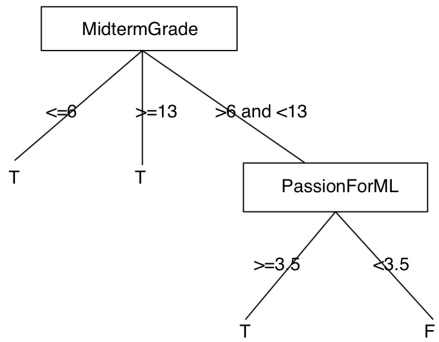


Figure 2: Blue dots denote A and red dots denote not an A

Let's first discretize the MidtermGrade. Observing the graph, all students scoring ≤ 6 *and* ≥ 13 *obtained A for their final. For students scoring between 6 to 13 (exclusive), if they are passionate about ML* (≥ 3.5)*, they scored A.*



4. [1**] **Uniform Blending (UB).**

   One of the simplest ensemble methods is UB. Given a set of hypothesis: $h_1, h_2, h_3, ..., h_T$, UB makes predictions simply by mixing the predictions given by $h_1, h_2, h_3, ..., h_T$ uniformly. Concretely, for binary classification, UB predicts by:

$$H(x) = \text{sign}(\sum_{t=1}^{T} h_t(x)), \tag{1}$$

   and for regression, UB predicts by:

$$H(x) = \frac{1}{T} \sum_{t=1}^{T} h_t(x) \tag{2}$$

[1]** question is harder than other questions in this tutorial.

Taking regression as an example, show that the performance (measured by out-of-sample error) of UB is no worse than the average performance over $h_1, h_2, h_3, ..., h_T$; i.e.:

$$\frac{1}{T}\sum_{t=1}^{T} L_{\text{test}}(h_t(x)) \geq L_{\text{test}}(H(x)) \tag{3}$$

Assume we evaluate the testing error by mean square error.

Hint: Start by calculating the average error over $h_1, h_2, h_3, ..., h_T$ for one fixed data point x.

Proving Equation 3 can give us an intuition on why ensembling can help to reduce the out-of-sample error.

Let's first see the case for only one fixed data point x. Suppose the ground truth function is $f(x)$. The average error over $h_1, h_2, h_3, ..., h_T$ for point x is

$$\frac{1}{T}\sum_{t=1}^{T}[h_t(x) - f(x)]^2 = \frac{1}{T}\sum_{t=1}^{T}\left[h_t(x)^2 - 2h_t(x)f(x) + f(x)^2\right] \tag{4}$$

$$= \frac{1}{T}\sum_{t=1}^{T} h_t(x)^2 - \frac{2f(x)}{T}\sum_{t=1}^{T} h_t(x) + \frac{1}{T}\sum_{t=1}^{T} f(x)^2 \tag{5}$$

$$= \frac{1}{T}\sum_{t=1}^{T} h_t(x)^2 - 2f(x)H(x) + \frac{1}{T}\sum_{t=1}^{T} f(x)^2 \qquad H(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x) \tag{6}$$

$$= \frac{1}{T}\sum_{t=1}^{T} h_t(x)^2 - 2f(x)H(x) + f(x)^2 \qquad f(x) \text{ does not depend on } t \tag{7}$$

$$= \frac{1}{T}\sum_{t=1}^{T} h_t(x)^2 - H(x)^2 + H(x)^2 - 2f(x)H(x) + f(x)^2 \tag{8}$$

$$= \frac{1}{T}\sum_{t=1}^{T} h_t(x)^2 - H(x)^2 + [H(x) - f(x)]^2 \tag{9}$$

$$= \frac{1}{T}\sum_{t=1}^{T} h_t(x)^2 - 2H(x)^2 + H(x)^2 + [H(x) - f(x)]^2 \tag{10}$$

$$= \frac{1}{T}\sum_{t=1}^{T} h_t(x)^2 - \frac{1}{T}\sum_{t=1}^{T} 2h_t(x)H(x) + \frac{1}{T}\sum_{t=1}^{T} H(x)^2 + [H(x) - f(x)]^2 \tag{11}$$

$$= \frac{1}{T}\sum_{t=1}^{T}\left[h_t(x)^2 - 2h_t(x)H(x) + H(x)^2\right] + [H(x) - f(x)]^2 \tag{12}$$

$$= \frac{1}{T}\sum_{t=1}^{T}[h_t(x) - H(x)]^2 + [H(x) - f(x)]^2 \tag{13}$$

To compute the out-of-sample error, we only need to compute the expected error over all x,

hence, the average out-of-sample error over $h_1, h_2, h_3, ..., h_T$ is

$$\frac{1}{T}\sum_{t=1}^{T} L_{test}(h_t(x)) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_x\left\{[h_t(x) - f(x)]^2\right\} \tag{14}$$

$$= \mathbb{E}_x\left\{\frac{1}{T}\sum_{t=1}^{T}[h_t(x) - f(x)]^2\right\} \tag{15}$$

$$= \mathbb{E}_x\left\{\frac{1}{T}\sum_{t=1}^{T}[h_t(x) - H(x)]^2 + [H(x) - f(x)]^2\right\} \tag{16}$$

$$= \mathbb{E}_x\left\{\frac{1}{T}\sum_{t=1}^{T}[h_t(x) - H(x)]^2\right\} + \mathbb{E}_x\left\{[H(x) - f(x)]^2\right\} \tag{17}$$

$$= \mathbb{E}_x\left\{\frac{1}{T}\sum_{t=1}^{T}[h_t(x) - H(x)]^2\right\} + L_{test}(H(x)) \tag{18}$$

$$\geq L_{test}(H(x)) \tag{19}$$

From the above deduction, we can see, the average out-of-sample error over $h_1, h_2, h_3, ..., h_T$ is greater than or equal to the out-of-sample error of $H(x)$. Hence, the performance (measured by out-of-sample error) of UB is no worse than the average performance over $h_1, h_2, h_3, ..., h_T$.