CS446: Machine Learning, Fall 2017, Homework 2

Name: Lanxiao Bai (lbai5)

Worked individually

Problem 1

Solution:

- In order to reach the minimum encoding length, we use the strategy that encode the most frequent letter with the least bit and vice versa. As a result, our encoding are as following

A	100
B	101
C	000
D	001
E	010
F	011

- Considering there are 6 symbols in T, if using 2 bits for each symbol, it is impossible to encode each symbol without making some a prefix of the other. As a result, we need 3 bits to encode symbols in T.

- We can calculate entropy

$$H(T) = -\sum_i P(T_i)\log_2 P(T_i) = -(\frac{1}{4}(-2-2-2)+\frac{1}{8}(-3)+\frac{1}{16}(-4-4)) = 2.375 \approx 3$$

  We see that the entropy gives an estimation of the least required length of encoding for a string.

Problem 2

Solution:

- **Base cases:** When $n = 1$, the decision tree does not need to grow, thus $f(1) = 0$. When $n = 2$, the decision needs to at least grow once and have two leaves, thus $f(2) = 2$. When $n = 3$, one of the leaf in the case of $n = 2$ needs to be split, and we get that $f(3) = 2 + 2 + 1 = 5$.

  Inductive hypothesis: Suppose for all $4 \leq n \leq k \in \mathbb{N}$ there is

$$f(n) = \min_{1\leq i\leq 3}[f(n-i) + f(i) + n]$$

Then when $n = k + 1$, in order to accommodate the new data in tree, while introduce least new path length, we apply a strategy to split the lowest leave in tree of $n = k$, since decision tree is a binary tree, we can determine the lowest tree by

$$h_{\text{lowest}}(n) = \lceil \log_2(n) \rceil$$

so that

$$\begin{aligned}
f(k+1) &= f(k) + 2(h_{\text{lowest}}(k) + 1) - h_{\text{lowest}}(k) \\
&= f(k) + h_{\text{lowest}}(k) + 2 \\
&= \min_{1 \le i \le 3} [f(k-i) + f(i) + k + h_{\text{lowest}}(k) + 2] \\
&= \min_{1 \le i \le 3} [f(k-i) + f(i) + k + h_{\text{lowest}}(k) + 2 + h_{\text{lowest}}(k-i) - h_{\text{lowest}}(k-i)] \\
&= \min_{1 \le i \le 3} [(f(k-i) + h_{\text{lowest}}(k-i) + 2) + f(i) + k + h_{\text{lowest}}(k) - h_{\text{lowest}}(k-i)] \\
&= \min_{1 \le i \le 3} [f(k+1-i) + f(i) + k + h_{\text{lowest}}(k) - h_{\text{lowest}}(k-i)] \\
&= \min_{1 \le i \le 3} [f(k+1-i) + f(i) + k + 1]
\end{aligned}$$

is proved.

By Strong Induction, we conclude the formula holds for all $n \in \mathbb{N}$.

- **Claim 1:** Let the total length of path of $D(T_d)$ be $L(D(T_d))$, then $f(|X'|) = L(D(T_d)) \Leftrightarrow D$ is optimal.

  Proof: "$\Rightarrow$": Suppose D is not optimal then there is D' that $L(D') < L(D) = f(|X'|)$. This is impossible since f is the minimum length of path a tree can have, thus we see that D has to be optimal.

  "$\Leftarrow$": If $L(D) \neq f(|X'|)$, then $L(D) > f(|X'|)$ since f is the minimum, then there is a smaller tree D' that $L(D') < L(D)$, thus D is not optimal. Hence, $L(D) = f(|X'|)$.

  We conclude that $f(|X'|) = L(D(T_d)) \Leftrightarrow D$ is optimal.

  Claim 2: $f(|X'|) = L(D(T_d))$ if and only if the set of non-singleton tests used in this optimal tree form an exact cover of X.

  Proof: "$\Rightarrow$": Suppose the non-singleton tests used in this optimal tree does not form an exact cover of X, then there are $t_1, t_2 \in \mathcal{T}'$ that $t_1 \cap t_2 \neq \emptyset$, thus at least one more split is needed to separate the overlapped symbols in two different subtrees, which cause the length of path larger than $f(|X'|)$. Thus, the non-singleton tests used in this optimal tree does form an exact cover of X.

  "$\Leftarrow$": If $L(D) \neq f(|X'|)$, then $L(D) > f(|X'|)$ since f is the minimum, then there are tests $t_1, t_2 \in \mathcal{T}'$ that $t_1 \cap t_2 \neq \emptyset$, the non-singleton tests used in this optimal tree does not form an exact cover of X. Hence, $L(D) = f(|X'|)$.

  So we conclude that $f(|X'|) = L(D(T_d))$ if and only if the set of non-singleton tests used in this optimal tree form an exact cover of X.

- Just as the question mentions, we define $DT(\mathcal{T}', X', w)$ where $X' = X \cup \{a, b, c\}$, $\{a, b, c\} \cap \emptyset$, $\mathcal{T}' = \mathcal{T} \cup \{\{x\} \mid x \in X'\}$, $w = f(|X'|)$. As we proved in step 2, $DT(\mathcal{T}', X', w)$ is true if and only if $EC3(\mathcal{T}, X)$. Since $EC3$ is NP-hard and $EC3$ can be reduced to decision tree problem, we see that DT is also NP-hard.
- By having the minimum number of tests, each time we need to predict from a set of features, minimum amount of calculation is needed and increased the speed as a result. Also, by having minimum number of tests, variance can be reduced.