Exercise 3

Context:
A dataset of pairs consisting of $\mathbb{T} : (\mathbb{R}^+ \times L \times \mathbb{R})^* \times \{0,1\} \to \mathbb{N}$ where L is a finite set of labels for every sample $s = ((vt_1, l_1, r_1) \dots (vt_n, l_n, r_n), y)$ with $\mathbb{T}(s) > 0$ $vt_1 = 0$ and for every $1 \leq i < n$, $vt_i \leq vt_{i+1}$.

Definition 1. A sequence tree on a set of labels L is a labelled binary tree $T = (V = V_l \cup V_v \cup V_c, E = E_\top \cup E_\bot, \mathcal{V}_l, \mathcal{V}_v, \mathcal{V}_c)$ such that:

- $V_{*1} \cap V_{*2} = \emptyset$ for any $*1, *2 \in \{l, v, c\}$ with $*1 \neq *2$;
- for each $v \in V_l \cup V_v$ we have $|\{(v, v') \in E_\bot\}| = 1$ and $|\{(v, v') \in E_\top\}| = 1$;
- for each $v \in V_c$ we have $|\{(v, v') \in E\}| = 0$;
- $\mathcal{V}_l : V_l \to L \times \mathbb{R}^+$, $\mathcal{V}_v : V_d \to \mathbb{R}$, and $\mathcal{V}_c : V_c \to L \times \{0,1\}$;
- for every $v \in V_v$ we have that either there exists $(v', v) \in E_\top$ or there exists $(v', v) \in E_\bot$ with $v' \in V_v$.

A sequence tree process a sample $x \in (\mathbb{R}^+ \times L \times \mathbb{R})^*$ according to the following algorithm.

Algorithm 1 Predict-sample

Input: $T, x, start = x[0].vt, value = None$
if T is a leaf **then**
  return T.class
end if
if T.root.type = LabelWithinDuration **then**
  $\varphi \leftarrow x[i].vt - \text{start} \leq T.root.duration \wedge x[i].label = T.root.label$
  if $\exists i(\varphi)$ **then**
    $i \leftarrow \min_{\varphi} i$
    $new\_start \leftarrow x[i].vt$
    $new\_value \leftarrow x[i].value$
    Predict-sample$(T.root.children[\top], x[j+1:], new_start, new_value)$
  else
    Predict-sample$(T.root.children[\bot], x, start, value)$
  end if
else
  if $value \leq T.root.value$ **then**
    Predict-sample$(T.root.children[\top], x, start, value))$
  else
    Predict-sample$(T.root.children[\bot], x, start, value))$
  end if
end if

Assignment:

Create an object SequenceTree with two methods fit and *predict* where the method fit gets a dataset $\mathbb{T}$ and create a sequence tree T using the information gain as metric, while the method *predict* implements the aforementioned algorithm.