**Practical No: 4**

**Candidate-elimination algorithm**

**AIM: For a given set of training data examples stored in a .csv file, implement and demonstrate the candidate-elimination algorithm to output a description of the set of all hypotheses consistent with the training examples.**

**Description:**

The candidate elimination algorithm incrementally builds the version space given a hypothesis space H and a set E of examples. The examples are added one by one; each example possibly shrinks the version space by removing the hypotheses that are inconsistent with the example. The candidate elimination algorithm does this by updating the general and specific boundary for each new example.

* You can consider this as an extended form of Find-S algorithm.
* Consider both positive and negative examples.
* Actually, positive examples are used here as Find-S algorithm (Basically they are generalizing from the specification).
* While the negative example is specified from generalize form.

**Terms :-**

**General Hypothesis:** Not Specifying features to learn the machine.

**G = {‘?’, ‘?’,’?’,’?’…}:**Number of attributes.

**Specific Hypothesis:** Specifying features to learn machine (Specific feature).

**S= {‘pi’,’pi’,’pi’…}:**Number of pi depends on number of attributes.

**Version Space:** It is intermediate of general hypothesis and Specific hypothesis. It not only just written one hypothesis but a set of all possible hypothesis based on training data-set.

**Candidate-elimination algorithm :-**

**Step1:** Load Data set

**Step2:** Initialize General Hypothesis and Specific Hypothesis.

**Step3:** For each training example

**Step4:** If example is positive example

if attribute\_value == hypothesis\_value:

Do nothing

else:

replace attribute value with '?' (Basically generalizing it)

**Step5:** If example is Negative example

Make generalize hypothesis more specific.

**Code and output :-**

import numpy as np

import pandas as pd

#Loading data from a csv file.

data = pd.DataFrame(data=pd.read\_csv('enjoysport.csv'))

print(data)

A screenshot of a computer

Description automatically generated

#Separating concept features from Target

concepts = np.array(data.iloc[:,0:6])

print(concepts)

A screenshot of a computer

Description automatically generated

#Isolating target into a separate DataFrame

#Copying last column to target array

target = np.array(data.iloc[:,6])

print(target)

A screenshot of a computer

Description automatically generated

def learn(concepts, target):

#Initialise S0 with the first instance from concepts.

#.copy()makes sure a new list is created instead of just pointing to the same memory location.

specific\_h = concepts[0].copy()

print("\nInitialization of specific\_h and genearal\_h")

print("\nSpecific Boundary: ", specific\_h)

general\_h = [["?" for i in range(len(specific\_h))] for i in range(len(specific\_h))]

print("\nGeneric Boundary: ",general\_h)

# The learning iterations.

for i, h in enumerate(concepts):

print("\nInstance", i+1 , "is ", h)

# Checking if the hypothesis has a positive target.

if target[i] == "yes":

print("Instance is Positive ")

for x in range(len(specific\_h)):

# Change values in S & G only if values change.

if h[x]!= specific\_h[x]:

specific\_h[x] ='?'

general\_h[x][x] ='?'

# Checking if the hypothesis has a positive target.

if target[i] == "no":

print("Instance is Negative ")

for x in range(len(specific\_h)):

# For negative hypothesis change values only in G.

if h[x]!= specific\_h[x]:

general\_h[x][x] = specific\_h[x]

else:

general\_h[x][x] = '?'

print("Specific Bundary after ", i+1, "Instance is ", specific\_h)

print("Generic Boundary after ", i+1, "Instance is ", general\_h)

print("\n")

# find indices where we have empty rows, meaning those that are unchanged.

indices = [i for i, val in enumerate(general\_h) if val == ['?', '?', '?', '?', '?', '?']]

for i in indices:

# remove those rows from general\_h

general\_h.remove(['?', '?', '?', '?', '?', '?'])

# Return final values

return specific\_h, general\_h

s\_final, g\_final = learn(concepts, target)

print("Final Specific\_h: ", s\_final, sep="\n")

print("Final General\_h: ", g\_final, sep="\n")

A white background with black text

Description automatically generatedA screenshot of a computer

Description automatically generated