Veermata Jijabai Technological Institute, Mumbai 400019

Assignment No.: 03

Aim : Implement Candidate Elimination Algorithm on the Titanic dataset

Name : Kiran K. Patil

Enrollment No.: 211070904

Branch : Computer Engineering

Course: Machine Learning Lab

Batch : IV

```
from google.colab import files
files.upload()
```

```
# Attributes
# survival - Survival (0 = No; 1 = Yes)
# class - Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
# name - Name
# sex - Sex
# age - Age
# sibsp - Number of Siblings/Spouses Aboard
# parch - Number of Parents/Children Aboard
# ticket - Ticket Number
# fare - Passenger Fare
# cabin - Cabin
# embarked - Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)
```

```
import numpy as np
import pandas as pd
```

```
df = pd.read_csv('titanic_dataset.csv')
df.drop(['Name', 'PassengerId'], axis=1, inplace=True)
df.drop(['Cabin'], inplace=True, axis=1)
df.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
0	0	3	male	34.5	0	0	330911	7.8292	Q
1	1	3	female	47.0	1	0	363272	7.0000	S
2	0	2	male	62.0	0	0	240276	9.6875	Q
3	0	3	male	27.0	0	0	315154	8.6625	S
4	1	3	female	22.0	1	1	3101298	12.2875	S

```
df.describe()
```

	Survived	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	0.363636	2.265550	30.272590	0.447368	0.392344	35.627188
std	0.481622	0.841838	14.181209	0.896760	0.981429	55.907576
min	0.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	0.000000	1.000000	21.000000	0.000000	0.000000	7.895800
50%	0.000000	3.000000	27.000000	0.000000	0.000000	14.454200
75%	1.000000	3.000000	39.000000	1.000000	0.000000	31.500000
max	1.000000	3.000000	76.000000	8.000000	9.000000	512.329200

```
df['Age'].isna().sum()
df.dropna(inplace=True)
```

```
df.isna().sum().sum()
```

```
0
```

```
bins1 = [0,5,10,18,25,40,80]
label1 = ['Infant','child','Teenager','Young Adult','Adult','Elderly']
df['Age Category'] = pd.cut(df['Age'], bins1, labels=label1)
df.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	Age Category
0	0	3	male	34.5	0	0	330911	7.8292	Q	Adult
1	1	3	female	47.0	1	0	363272	7.0000	S	Elderly
4	1	3	female	22.0	1	1	3101298	12.2875	S	Young Adult

```
bins2 = [0,200,400,600]
label2 = ['General', 'Second', 'First']
df['Fare Category'] = pd.cut(df['Fare'], bins2, labels=label2)
df.tail()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	Age Category	Fare Category
409	1	3	female	3.0	1	1	SOTON/O.Q. 3101315	13.775	S	Infant	General
411	1	1	female	37.0	1	0	19928	90.000	Q	Adult	General
412	1	3	female	28.0	0	0	347086	7.775	S	Adult	General
414	1	1	female	39.0	0	0	PC 17758	108.900	C	Adult	General

```
bins2 = [-1,2,4,8]
label3 = ['Low', 'Medium', 'High']
df['Sibsp Category'] = pd.cut(df['SibSp'], bins2, labels=label3)
df.tail()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	Age Category	Fare Category	Sibsp Category
409	1	3	female	3.0	1	1	SOTON/O.Q. 3101315	13.775	S	Infant	General	Low
411	1	1	female	37.0	1	0	19928	90.000	Q	Adult	General	Low
412	1	3	female	28.0	0	0	347086	7.775	S	Adult	General	Low
414	1	1	female	39.0	0	0	PC 17758	108.900	C	Adult	General	Low
415	0	3	male	38.5	0	0	SOTON/O.Q. 3101262	7.250	S	Adult	General	Low

```
df.drop(['Age', 'SibSp', 'Fare'], inplace=True,axis=1)
```

```
df.drop(['Ticket'],inplace=True,axis=1)
df.head()
```

	Survived	Pclass	Sex	Parch	Embarked	Age Category	Fare Category	Sibsp Category
0	0	3	male	0	Q	Adult	General	Low
1	1	3	female	0	S	Elderly	General	Low
2	0	2	male	0	Q	Elderly	General	Low
3	0	3	male	0	S	Adult	General	Low
4	1	3	female	1	S	Young Adult	General	Low

```
data=df[:20]
data.head()
```

	Survived	Pclass	Sex	Parch	Embarked	Age Category	Fare Category	Sibsp Category
0	0	3	male	0	Q	Adult	General	Low
1	1	3	female	0	S	Elderly	General	Low
2	0	2	male	0	Q	Elderly	General	Low
3	0	3	male	0	S	Adult	General	Low
4	1	3	female	1	S	Young Adult	General	Low

```
# data = pd.read_csv('filtered_data.csv')
concepts = np.array(data.iloc[:,0:-1])
```

```
print("\nInstances are:\n",concepts)
target = np.array(data.iloc[:,1])
print("\nTarget Values are: ",target)

def learn(concepts, target):
    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)

    for i, h in enumerate(concepts):
        print("\nInstance", i+1 , "is ", h)
        if target[i] == 1:
            print("Instance is Positive ")
            for x in range(len(specific_h)):
                if h[x]!= specific_h[x]:
                    specific_h[x] ='?'
                    general_h[x][x] ='?'

        if target[i] == 0:
            print("Instance is Negative ")
            for x in range(len(specific_h)):
                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")

    indices = [i for i, val in enumerate(general_h) if val == ['?', '?', '?', '?', '?', '?']]
    for i in indices:
        general_h.remove(['?', '?', '?', '?', '?', '?'])
    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")
```

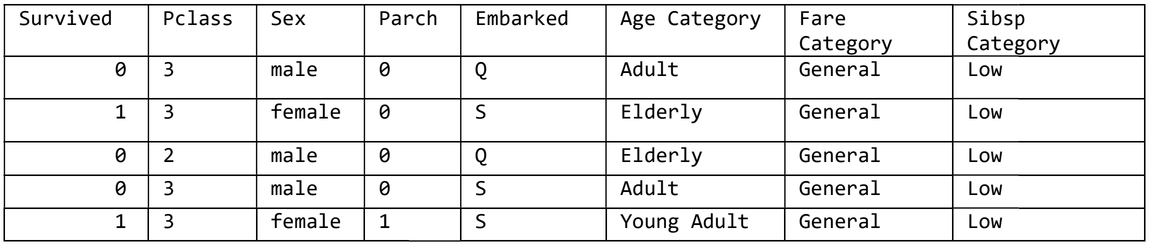
```
Instance 19 is  [1 3 'female' 0 'C' 'Elderly' 'General']
Specific Bundary after  19 Instance is  ['?' '?' '?' 0 '?' '?' 'General']
Generic Boundary after  19 Instance is  [['?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?



Instance 20 is [0 1 'male' 0 'C' 'Elderly' 'General']
Instance is Positive
Specific Bundary after  20 Instance is  ['?' '?' '?' 0 '?' '?' 'General']
Generic Boundary after  20 Instance is  [['?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?


Final Specific_h:
['?' 1 3 'female' '?' '?' '?' 'General']
Final General_h:
[['?', '?', '?', '?', '?', '?', '?', '?'], ['?', 1, '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?', '?'], ['?',
'?', '?', 'female', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?', '?'], ['?',
'?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?', '?']]
```

Colab paid products - Cancel contracts here

Conclusion :

Candidate elimination algorithm is implemented on “Titanic_Dataset” The number of training examples chosen are first four examples as more number of training examples result in complete generalization of specific hypothesis (i.e. Specific boundary : All ?’s).

The accuracy is calculated as the ratio of number of examples satisfying the hypothesis (Converged hypothesis or all hypothesis within most specific and most generalized Boundaries of version space) generated by Candidate Elimination algorithm to the total Number of testing samples. The training concepts involve.



Survived	Pclass	Sex	Parch	Embarked	Age Category	Fare Category	Sibsp Category
0	3	male	0	Q	Adult	General	Low
1	3	female	0	S	Elderly	General	Low
0	2	male	0	Q	Elderly	General	Low
0	3	male	0	S	Adult	General	Low
1	3	female	1	S	Young Adult	General	Low

The algorithm gives most specific boundary as <'?' 1 3 'female' '?' '?' '?' 'General'>

The generalized boundary is same as initialized (most general). On testing remaining data rows with hypothesis within most specific and most
general boundaries i.e.

[['?', '?', '?', '?', '?', '?', '?', '?'], ['?', 1, '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', 'female', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?', '?'], ['?', '?', '?', '?', '?', '?', '?', '?']]

Thus, Candidate elimination algorithm is successfully implemented and results are analyzed.