Experiment 1 - Find S Algorithm

April 30, 2023

1 Experiment Details

1.1 Submitted By

Desh Iyer, 500081889, Year III, AI/ML(H), B5

1.2 Problem Statement

Implement and demonstrate the FIND S algorithm for finding the most specific hypothesis based on a given set of training data samples. Read the training data from a `.csv` file.

1.3 Theory

The FIND-S algorithm is a concept learning algorithm used to induce the most specific hypothesis from a set of training data samples. The goal of the algorithm is to generate a hypothesis that fits all positive training examples and no negative training examples. The algorithm works by initializing the hypothesis with the most specific possible hypothesis, which in the case of a boolean function is a conjunction of n negated literals, where n is the number of attributes in the data. The algorithm then iteratively refines the hypothesis by making it more specific based on the training data.

Here are the terms used in the description of the algorithm.

- A hypothesis h is a conjunction of n literals, where each literal can take one of two values: true or false.
- A training example is a pair (x, y), where x is an n-dimensional vector of attribute values, and y is either true or false.
- A positive example is a training example where y is true.
- A negative example is a training example where y is false.

1.4 Advantages

- The algorithm is simple and easy to understand.
- The algorithm is guaranteed to converge to the correct hypothesis, provided that the hypothesis space is rich enough to represent the target concept.

1.5 Limitations

- The algorithm can only handle boolean functions, and does not work with continuous or categorical data.

- The algorithm is sensitive to noise in the training data, and can produce an overly specific hypothesis if there are errors in the training data.
- The algorithm only produces a single hypothesis, and does not provide any measure of the quality or confidence of the hypothesis.

1.6 Pseudocode

The FIND-S algorithm starts by initializing the hypothesis h with the most specific hypothesis possible:

```
h ← < ¬, ¬, ..., ¬ >
```

Then, for each positive training example x in the training data, it updates the hypothesis by setting the i-th literal to true if the i-th attribute of x is true, and leaves it unchanged otherwise. For each negative training example, the algorithm does not update the hypothesis.

```
for each training example (x, y) do
  if y = true then
    for i = 1 to n do
      if h[i] = ¬ and x[i] = true then
        h[i] ← true
    end for
  end if
end for
```

2 Import Libraries

[ ]:
```python
import pandas as pd
import numpy as np
```

3 Implement Algorithm

Declare a function to calculate the final specific hypothesis given a vector of concepts (tuples) and a vector of targets.

[ ]:
```python
def train(concepts, targets, specificHypothesis):
    for i, val in enumerate(targets):
        if val == 'Yes':
            specificHypothesis = concepts[i]
            break

    for i, val in enumerate(concepts):
        if targets[i] == 'Yes':
            for i in range (len(specificHypothesis)):
                if val[i] != specificHypothesis[i]:
                    specificHypothesis[i] = '?'

    return specificHypothesis
```

4 Import Dataset

```
[ ]: data = pd.read_csv('../data/find-s.csv',␣
      ↪names=['Sky','Temperature','Humidity','Wind','Water','Forecast','Enjoy Sport?
      ↪'])
```

Here's what the data looks like in a data frame.

```
[ ]: data
```

```
[ ]:      Sky Temperature Humidity    Wind Water Forecast Enjoy Sport?
     0  Sunny        Warm   Normal  Strong  Warm     Same          Yes
     1  Sunny        Warm     High  Strong  Warm     Same          Yes
     2  Rainy        Cold     High  Strong  Warm   Change           No
     3  Sunny        Warm     High  Strong  Cool   Change          Yes
```

Retrieving the data points as the vector of concepts (tuples) of the data set.

```
[ ]: dataPoints = np.array(data)[:, :-1]
     phiLength = dataPoints.shape[0] + 1
```

Retrieving the target vector.

```
[ ]: dataTarget = np.array(data)[:, -1]
     dataTarget
```

```
[ ]: array(['Yes', 'Yes', 'No', 'Yes'], dtype=object)
```

5 Calculate Hypotheses

Defining the specific hypothesis to be all zeros initially.

```
[ ]: specificHypothesis = np.zeros(phiLength)
     specificHypothesis
```

```
[ ]: array([0., 0., 0., 0., 0.])
```

Calling the function defined above and obtaining our final hypothesis.

```
[ ]: print(f'The final hypothesis is: {train(dataPoints, dataTarget,␣
      ↪specificHypothesis)}')
```

```
The final hypothesis is: ['Sunny' 'Warm' '?' 'Strong' '?' '?']
```