A Report on
Candidate Elimination Algorithm

by

Kshitij Sharma - 2012A7PS009H
Rohit Sharma - 2012A7PS050H
Abhishek Kaushik - 2012A7PS056H
Prakhar Gupta - 2012A7PS059H

BITS F464 - Machine Learning

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)
HYDERABAD CAMPUS

(12th November 2014)

Contents

1. Description
2. Step by Step Algorithm
3. Experiment
4. Result

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.

Step by Step Algorithm

The version space method handles positive and negative examples symmetrically.

Given:

- A representation language.
- A set of positive and negative examples expressed in that language.

Compute: a concept description that is consistent with all the positive examples and none of the negative examples.

Method:

- Initialize G, the set of maximally general hypotheses, to contain one element: the null description (all features are variables).
- Initialize S, the set of maximally specific hypotheses, to contain one element: the first positive example.
- Accept a new training example.
    - If the example is **positive**:
        1. Generalize all the specific models to match the positive example, but ensure the following:
            - The new specific models involve minimal changes.
            - Each new specific model is a specialization of some general model.
            - No new specific model is a generalization of some other specific model.
        2. Prune away all the general models that fail to match the positive example.
    - If the example is **negative**:
        1. Specialize all general models to prevent match with the negative example, but ensure the following:
            - The new general models involve minimal changes.
            - Each new general model is a generalization of some specific model.

- No new general model is a specialization of some other general model.

2. Prune away all the specific models that match the negative example.

- If S and G are both singleton sets, then:

1. if they are identical, output their value and halt.
2. if they are different, the training cases were inconsistent. Output this result and halt.
3. else continue accepting new training examples.

The algorithm stops when:

1. It runs out of data.
2. The number of hypotheses remaining is:
   - 0 - no consistent description for the data in the language.
   - 1 - answer (version space converges).
   - >2- all descriptions in the language are implicitly included.

Experiment

1. The code has been written so as to accept the training data as input at runtime.

2. Three different files were used as training data to conduct three different experiments:
   Each Training file consists of following:
   n <- number of training examples
   n_attr <- number of attributes (including target attribute) in the training examples
   Now next n lines follow giving the input examples.

3. The Candidate elimination algorithm is applied on the training data and the resultant general and specific boundaries are shown as output.

Result

For example:

Input: training3.txt

```
5
4
big red circle no
small red triangle no
small red circle yes
big blue circle no
small blue circle yes
```

Output:

```
Specific Boundary: <small,?,circle>
Generic Boundary: <small,?,circle>
```