Machine Learning (CS60050) – Weekly Report

Kaushal Banthia (19CS10039)

Week 2: 19th August, 2021

Topics Covered:

- Consistent Hypothesis
- Version Space
- List-Then-Eliminate Algorithm
- Compact Representation of the Version Space
- Candidate - Elimination Algorithm

Summary (Topic Wise):

- Consistent Hypothesis

  - A hypothesis (h) is consistent with a set of training examples (D), of target concept (c) if $h(x) = c(x)\ \forall\ x\ \in D$
  - Notation: $Consistent\ (h, D) \equiv \forall < x, c(x) > \in D\ :: h(x) = c(x)$
  - An agnostic hypothesis is one that may label a training sample erroneously, i.e., $Agnostic\ (h, D) \equiv \exists < x, c(x) > \in D\ :: h(x) \neq c(x)$

- Version Space

  - The set of hypotheses from hypothesis space (H) that are consistent with the training examples (D).
  - Notation: $VS_{H,D} = \{h | h \in H\ \wedge\ Consistent(h, D)\}$

- List-Then-Eliminate Algorithm (Brute Force Procedure)

```
1. Version Space ← list of all hypotheses in H
2. for each training example <x, c(x)>
      3. Remove from the Version Space, any hypothesis h for which h(x) ≠ c(x)
4. Output the list of hypotheses in the Version Space
```

- Compact Representation of the Version Space

  - Store the most and the least general boundaries of space.
    - Generalize from most specific boundaries using positive samples.
    - Specialize from most general boundaries using negative samples.
  - Generate all intermediate h's in the Version Space (consistent with all the training examples).
  - There is a general boundary G (set of maximally general members that are consistent with D).
  - There is a specific boundary S (set of maximally specific members that are consistent with D).
  - Theorem: Every member of the version space lies between the S, G boundary, i.e., $VS_{H,D} = \{h | h \in H\ \wedge\ \exists\ s\ \in S\ \exists\ g\ \in G\ (g \geq h \geq s)\}$

- Candidate-Elimination Algorithm

```
1. for each training example d in D
2. if d is positive:
   3. Remove from G, every hypothesis that is inconsistent with d.
   4. for each hypothesis s in S, that is inconsistent with d:
      5. Remove s from S
      6. Add to S, all the minimal generalizations h of s, such that
            • h is consistent with d
            • Some member of G is more general than h
   7. Remove from S, every hypothesis, that is more general than another
   hypothesis in S.
8. if d is negative:
   9. Remove from S, every hypothesis that is inconsistent with d.
   10. for each hypothesis g in G, that is inconsistent with d:
      11. Remove g from G
      12. Add to G, all the minimal generalizations h of g, such that
            • h is consistent with d
            • Some member of S is more general than h
   13. Remove from G, every hypothesis, that is less general than another
   hypothesis in G.
```

There are certain assumptions for this algorithm to work:

- There is a hypothesis h in H, describing the target function c.
- There are no errors.
- Everything except the positive examples is negative (The classification is binary).

Concepts Challenging to Comprehend: None yet.

Interesting and Exciting Concepts: Candidate Elimination Algorithm and the Compact Representation of the Version Space.

Concepts not understood: None yet.

A novel idea: In the List-Then-Eliminate Algorithm, if we arrange the hypotheses in order of their increasing specificity (the most general hypothesis $<?,?,?,...,?,?>$, being the first in order) and then we start the algorithm, we could have some optimizations when we find the first inconsistent hypothesis, because any hypothesis that is specific to that inconsistent hypothesis would also be inconsistent.