Implementing Mitchell's Candidate Elimination Algorithm

31/03/2011

Wim Looman
University of Canterbury

Abstract

This report details the design, implementation and testing of Mitchell's Candidate Elimination Algorithm (CEA). CEA is a learning algorithm based off inductive learning that can be used to learn a set of data then classify unseen instances from this dataset. The language used for this implementation was Ruby, this allowed a very simple to understand object oriented approach with a minimum of code (the main algorithm and supporting code requires just 167 lines, including whitespace). The algorithm was a success with the provided dataset, it easily learnt and classified the required tasks.

Table of Contents

1 Introduction **2**

2 Results **3**

2.1 Training . . . 3

2.1.1 Using all examples . . . 3

2.1.2 Using just 10 examples . . . 12

2.2 Classifying . . . 16

2.2.1 The Version Spaces . . . 16

2.2.2 The Classifications . . . 16

3 Conclusion **19**

1 Introduction

Mitchell's candidate elimination algorithm (CEA) is an induction based system for learning a set of data and utilising this trained system for classifying new data.

The main concept in CEA is the version space, this is a set of hypotheses bounded by two sets, the general set (G) and the specific set (S). G contains the most general hypotheses that are consistent with the previously seen negative examples; while S contains the most specific hypotheses that cover all of the previously seen positive examples.

This generality and specificity introduce one important constraint on the hypothesis space of any problem that CEA is used to solve, it must be possible to say whether one hypothesis is more general or more specific than another and to be able to generate the set of more specific/general hypotheses from a single hypothesis.

Because of this constraint the hypothesis space can be viewed as a lattice, each hypothesis has a set of hypotheses that are one step more specific and a set of hypotheses that are one step more general. In the following implementation these are named descendant and ancestor hypotheses respectively.

To derive this version space from the training set a very simple algorithm can be used, this is very similar to the algorithm described in the handout with just a slight change to how hypotheses are generalised and specialised to better fit the data structure chosen, the overall result is identical:

1. Initialize set S to `(null, null, ..., null)`, set G to `(undefined, undefined, ..., undefined)`
2. **For each positive example *d*:**
    a. Remove from G any hypothesis that fails to cover *d*
    b. **While there is at least one hypothesis in S that fails to cover *d*:**
        i. Remove all hypotheses *s* in S that fail to cover *d*
        ii. Add all ancestors of each hypothesis *s* to S
        iii. Remove from S any hypothesis that is more general than another in S
3. **For each negative example *d*:**
    a. Remove from S any hypothesis inconsistent with *d*
    b. **While there is at least one hypothesis in G that is not consistent with *d*:**
        i. Remove all hypotheses *g* in G that are not consistent with *d*
        ii. Add all descendants of each hypothesis *g* to G
        iii. Remove from G any hypothesis that is more specific than another in G

The largest change with this algorithm is in performance. The original algorithm removed hypotheses that do not cover/are inconsistent with the example while generalising/specialising the sets. This version keeps generalising and specialising these hypotheses until they are consistent, they will then be removed from the set as they are just a more general/specific version of one of the other hypotheses in the set.

2 Results

2.1 Training

2.1.1 Using all examples

Recognising SOFT

The starting version space is:

```
S: { astigmatic: null ,      tear_prod: null ,      age: null ,      prescription: null      }
G: { astigmatic: undefined , tear_prod: undefined , age: undefined , prescription: undefined }
```

The first example is a negative example and is covered by `G`. This causes `G` to be made more specific to ensure the case is no longer covered:

```
Example: {:astigmatic=>:no, :tear_prod=>:reduced, :age=>:young, :prescription=>:myope} ==> none
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: null ,      tear_prod: null ,      age: null ,      prescription: null      }
G:
  { astigmatic: yes ,       tear_prod: undefined , age: undefined , prescription: undefined }
  { astigmatic: undefined , tear_prod: normal ,    age: undefined , prescription: undefined }
  { astigmatic: undefined , tear_prod: undefined , age: middle ,    prescription: undefined }
  { astigmatic: undefined , tear_prod: undefined , age: old ,       prescription: undefined }
  { astigmatic: undefined , tear_prod: undefined , age: undefined , prescription: hyper     }
```

The next example is positive and is not covered by `S`. This causes `S` to be made more general to include the example and `G` to have all hypotheses that do not cover the new `S` to be removed:

```
Example: {:astigmatic=>:no, :tear_prod=>:normal, :age=>:young, :prescription=>:myope} ==> soft
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: no ,        tear_prod: normal ,    age: young ,     prescription: myope     }
G: { astigmatic: undefined , tear_prod: normal ,    age: undefined , prescription: undefined }
```

The next example is negative and is consistent with `G` so the version space does not change:

```
Example: {:astigmatic=>:yes, :tear_prod=>:reduced, :age=>:young, :prescription=>:myope} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: no ,        tear_prod: normal ,    age: young ,     prescription: myope     }
G: { astigmatic: undefined , tear_prod: normal ,    age: undefined , prescription: undefined }
```

The next example is negative and is inconsistent with `G`. This causes `G` to again be made more specific to not cover the example:

```
Example: {:astigmatic=>:yes, :tear_prod=>:normal, :age=>:young, :prescription=>:myope} ==> hard
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: no ,        tear_prod: normal ,    age: young ,     prescription: myope     }
G:
  { astigmatic: no ,        tear_prod: normal ,    age: undefined , prescription: undefined }
  { astigmatic: undefined , tear_prod: normal ,    age: middle ,    prescription: undefined }
  { astigmatic: undefined , tear_prod: normal ,    age: old ,       prescription: undefined }
  { astigmatic: undefined , tear_prod: normal ,    age: undefined , prescription: hyper     }
```

The next example is negative and consistent with `G` so nothing changes:

```
Example: {:astigmatic=>:no, :tear_prod=>:reduced, :age=>:young, :prescription=>:hyper} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,    age: young,      prescription: myope      }
G:
   { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
   { astigmatic: undefined, tear_prod: normal,    age: middle,    prescription: undefined  }
   { astigmatic: undefined, tear_prod: normal,    age: old,       prescription: undefined  }
   { astigmatic: undefined, tear_prod: normal,    age: undefined, prescription: hyper      }
```

The next example is positive and not covered by S so S is made more general and the inconsistent hypotheses in G are removed:

```
Example: {:astigmatic=>:no, :tear_prod=>:normal, :age=>:young, :prescription=>:hyper} ==> soft
 Prior Classification: unknown
 After Classification: positive
-----
S: { astigmatic: no,        tear_prod: normal,    age: young,      prescription: undefined  }
G:
   { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
   { astigmatic: undefined, tear_prod: normal,    age: undefined, prescription: hyper      }
```

The next example is negative and consistent with G so nothing changes:

```
Example: {:astigmatic=>:yes, :tear_prod=>:reduced, :age=>:young, :prescription=>:hyper} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,    age: young,      prescription: undefined  }
G:
   { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
   { astigmatic: undefined, tear_prod: normal,    age: undefined, prescription: hyper      }
```

The next example is negative and inconsistent with G so G is made more specific:

```
Example: {:astigmatic=>:yes, :tear_prod=>:normal, :age=>:young, :prescription=>:hyper} ==> hard
 Prior Classification: unknown
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,    age: young,      prescription: undefined  }
G:
   { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
   { astigmatic: undefined, tear_prod: normal,    age: middle,    prescription: hyper      }
   { astigmatic: undefined, tear_prod: normal,    age: old,       prescription: hyper      }
```

The next example is negative and consistent with G so the version space stays the same:

```
Example: {:astigmatic=>:no, :tear_prod=>:reduced, :age=>:middle, :prescription=>:myope} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,    age: young,      prescription: undefined  }
G:
   { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
   { astigmatic: undefined, tear_prod: normal,    age: middle,    prescription: hyper      }
   { astigmatic: undefined, tear_prod: normal,    age: old,       prescription: hyper      }
```

The next example is positive and not covered by S so S is generalised and inconsistent hypotheses from G are removed. After this S and G only contain the same hypothesis so this version space has converged. Assuming the examples are consistent then neither S nor G will change from now on:

```
Example: {:astigmatic=>:no, :tear_prod=>:normal, :age=>:middle, :prescription=>:myope} ==> soft
 Prior Classification: unknown
 After Classification: positive
-----
S: { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }

Example: {:astigmatic=>:yes, :tear_prod=>:reduced, :age=>:middle, :prescription=>:myope} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
```

```
Example: {:astigmatic=>:yes, :tear_prod=>:normal, :age=>:middle, :prescription=>:myope} ==> hard
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }

Example: {:astigmatic=>:no, :tear_prod=>:reduced, :age=>:middle, :prescription=>:hyper} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }

Example: {:astigmatic=>:no, :tear_prod=>:normal, :age=>:middle, :prescription=>:hyper} ==> soft
 Prior Classification: positive
 After Classification: positive
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }

Example: {:astigmatic=>:yes, :tear_prod=>:reduced, :age=>:middle, :prescription=>:hyper} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }

Example: {:astigmatic=>:yes, :tear_prod=>:normal, :age=>:middle, :prescription=>:hyper} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }

Example: {:astigmatic=>:no, :tear_prod=>:reduced, :age=>:old, :prescription=>:myope} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }

Example: {:astigmatic=>:yes, :tear_prod=>:reduced, :age=>:old, :prescription=>:myope} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }

Example: {:astigmatic=>:yes, :tear_prod=>:normal, :age=>:old, :prescription=>:myope} ==> hard
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }

Example: {:astigmatic=>:no, :tear_prod=>:reduced, :age=>:old, :prescription=>:hyper} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }

Example: {:astigmatic=>:no, :tear_prod=>:normal, :age=>:old, :prescription=>:hyper} ==> soft
 Prior Classification: positive
 After Classification: positive
-----
S: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,     age: undefined, prescription: undefined  }
```

```
Example: {:astigmatic=>:yes, :tear_prod=>:reduced, :age=>:old, :prescription=>:hyper} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }

Example: {:astigmatic=>:yes, :tear_prod=>:normal, :age=>:old, :prescription=>:hyper} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
G: { astigmatic: no,        tear_prod: normal,    age: undefined, prescription: undefined  }
```

Recognising HARD

The starting version spaces is:

```
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null       }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: undefined  }
```

The first example is a negative example and covered by `G`, this causes `G` to be made more specific to no longer cover it:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null       }
G:
  { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: undefined  }
  { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: normal     }
```

The next example is a negative example and covered by `G` as well, this causes `G` to be made more specific again so it no longer covers it:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> soft
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null       }
G:
  { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: undefined  }
```

Again the example is negative and covered by `G` so `G` is made more specific:

```
Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null       }
G:
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: undefined  }
  { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal     }
```

Finally we get a positive example, this isn't covered by `S` so `S` is generalised to cover it. Then any hypotheses in `G` that are inconsistent with `S` are removed:

```
Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: yes,       age: young,     prescription: myope,     tear_prod: normal     }
G: { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal     }
```

Nothing changes as these are negative examples consistent with `G`:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,      age: young,     prescription: myope,     tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }

Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:normal} ==> soft
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,      age: young,     prescription: myope,     tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,      age: young,     prescription: myope,     tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }
```

A new positive example causes `S` to be made more general again:

```
Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:hyper, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: yes,      age: young,     prescription: undefined, tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }
```

And some more negative examples consistent with `G`:

```
Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,      age: young,     prescription: undefined, tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }

Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:myope, :tear_prod=>:normal} ==> soft
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,      age: young,     prescription: undefined, tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }

Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,      age: young,     prescription: undefined, tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }
```

A new positive example. `S` is again generalised and is now converged with `G`. If the dataset is consistent then nothing will change from here on:

```
Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:myope, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }

Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }

Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:hyper, :tear_prod=>:normal} ==> soft
Prior Classification: negative
```

```
After Classification: negative
-----
S: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }

Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }
G: { astigmatic: yes,      age: undefined, prescription: undefined, tear_prod: normal     }
```

But wait, the dataset is inconsistent. This is a negative example and was covered by `G` so `G` had to be made more specific. However this meant that the only hypothesis in `S` had to be removed to make `S` consistent with `G`. Because it was `S` that got wiped out our version space is now only usable for determining negative examples, there will be no false positives but there can be false negatives:

```
Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:hyper, :tear_prod=>:normal} ==> none
Prior Classification: positive
After Classification: negative
-----
S:
G:
  { astigmatic: yes,      age: young,     prescription: undefined, tear_prod: normal     }
  { astigmatic: yes,      age: old,       prescription: undefined, tear_prod: normal     }
  { astigmatic: yes,      age: undefined, prescription: myope,     tear_prod: normal     }
```

And we're back to consistent negative examples:

```
Example: {:astigmatic=>:no, :age=>:old, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S:
G:
  { astigmatic: yes,      age: young,     prescription: undefined, tear_prod: normal     }
  { astigmatic: yes,      age: old,       prescription: undefined, tear_prod: normal     }
  { astigmatic: yes,      age: undefined, prescription: myope,     tear_prod: normal     }

Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S:
G:
  { astigmatic: yes,      age: young,     prescription: undefined, tear_prod: normal     }
  { astigmatic: yes,      age: old,       prescription: undefined, tear_prod: normal     }
  { astigmatic: yes,      age: undefined, prescription: myope,     tear_prod: normal     }
```

Until we get to a positive one, notice that even after using this to train the classification is still unknown because `S` is empty:

```
Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:myope, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: unknown
-----
S:
G:
  { astigmatic: yes,      age: old,       prescription: undefined, tear_prod: normal     }
  { astigmatic: yes,      age: undefined, prescription: myope,     tear_prod: normal     }
```

Three more consistent negative examples:

```
Example: {:astigmatic=>:no, :age=>:old, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S:
G:
  { astigmatic: yes,      age: old,       prescription: undefined, tear_prod: normal     }
  { astigmatic: yes,      age: undefined, prescription: myope,     tear_prod: normal     }

Example: {:astigmatic=>:no, :age=>:old, :prescription=>:hyper, :tear_prod=>:normal} ==> soft
```

```
 Prior Classification: negative
 After Classification: negative
-----
S:
G:
  { astigmatic: yes,       age: old,       prescription: undefined, tear_prod: normal    }
  { astigmatic: yes,       age: undefined, prescription: myope,     tear_prod: normal    }

Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
 Prior Classification: negative
 After Classification: negative
-----
S:
G:
  { astigmatic: yes,       age: old,       prescription: undefined, tear_prod: normal    }
  { astigmatic: yes,       age: undefined, prescription: myope,     tear_prod: normal    }
```

And another inconsistent negative example to make G more specific and probably increase the number of false negatives:

```
Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:hyper, :tear_prod=>:normal} ==> none
 Prior Classification: unknown
 After Classification: negative
-----
S:
G: { astigmatic: yes,       age: undefined, prescription: myope,     tear_prod: normal    }
```

Recognising NONE

The starting version space is:

```
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null      }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: undefined }
```

The first example is positive and not covered by S so S is made more general to cover it:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
 Prior Classification: unknown
 After Classification: positive
-----
S: { astigmatic: no,        age: young,     prescription: myope,     tear_prod: reduced   }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: undefined }
```

The next example is negative and inconsistent with G so G is made more specific to not cover it:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> soft
 Prior Classification: unknown
 After Classification: negative
-----
S: { astigmatic: no,        age: young,     prescription: myope,     tear_prod: reduced   }
G:
  { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: undefined }
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: undefined }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: undefined }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: undefined }
  { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced   }
```

Another positive example not covered by S causes S to be made more general and inconsistent hypotheses from G to be removed:

```
Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
 Prior Classification: unknown
 After Classification: positive
-----
S: { astigmatic: undefined, age: young,     prescription: myope,     tear_prod: reduced   }
G:
  { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: undefined }
  { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced   }
```

A negative example inconsistent with G makes G more specific:

```
Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: undefined, age: young,     prescription: myope,     tear_prod: reduced    }
G:
  { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
  { astigmatic: yes,       age: middle,    prescription: undefined, tear_prod: undefined  }
  { astigmatic: yes,       age: old,       prescription: undefined, tear_prod: undefined  }
  { astigmatic: yes,       age: undefined, prescription: hyper,     tear_prod: undefined  }
```

A positive example not covered by `S` so `S` is made more general and inconsistent hypotheses from `G` are removed:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: undefined, age: young,     prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
```

Some negative examples consistent with `G` and positive examples covered by `S` so nothing changes:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:normal} ==> soft
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: undefined, age: young,     prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: positive
After Classification: positive
-----
S: { astigmatic: undefined, age: young,     prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:hyper, :tear_prod=>:normal} ==> hard
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: undefined, age: young,     prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
```

A positive example not covered by `S` so `S` is made more general. `S` and `G` are now converged:

```
Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
```

Some more negative/consistent and positive/covered examples:

```
Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:myope, :tear_prod=>:normal} ==> soft
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }

Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: positive
After Classification: positive
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }

Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:myope, :tear_prod=>:normal} ==> hard
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
```

```
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }

Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: positive
After Classification: positive
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }

Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:hyper, :tear_prod=>:normal} ==> soft
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }

Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: positive
After Classification: positive
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: reduced    }
Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:hyper, :tear_prod=>:normal} ==> none
```

And we hit an inconsistency, a positive examples that is not covered by G or S. To fix this S is made more general and G is blanked out as it was inconsistent with the new S. Since G is now empty we no longer have any ability to determine if an example is negative, we can only know that all positives will definitely be classified as positive but some negatives will appear as false positives:

```
Prior Classification: negative
After Classification: positive
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: undefined  }
G:

Example: {:astigmatic=>:no, :age=>:old, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: positive
After Classification: positive
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: undefined  }
G:

Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: positive
After Classification: positive
-----
S: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: undefined  }
G:
```

And another inconsistency, a negative example that is now covered by S. This causes the offending hypothesis to be removed from S making this version space useless. With how the classification is set up this will now simply return negative for all examples:

```
Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:myope, :tear_prod=>:normal} ==> hard
Prior Classification: positive
After Classification: negative
-----
S:
G:

Example: {:astigmatic=>:no, :age=>:old, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S:
G:

Example: {:astigmatic=>:no, :age=>:old, :prescription=>:hyper, :tear_prod=>:normal} ==> soft
Prior Classification: negative
After Classification: negative
-----
S:
```

```
G:

Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
----
S:
G:

Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:hyper, :tear_prod=>:normal} ==> none
Prior Classification: negative
After Classification: negative
----
S:
G:
```

2.1.2 Using just 10 examples

Recognising SOFT with just 10 examples

The starting version space:

```
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null      }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: undefined }
```

The examples are exactly the same as the first 10 in Recognising SOFT:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: negative
----
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null      }
G:
  { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: undefined }
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: undefined }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: undefined }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: undefined }
  { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: normal    }

Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> soft
Prior Classification: unknown
After Classification: positive
----
S: { astigmatic: no,        age: young,     prescription: myope,     tear_prod: normal    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: normal    }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
----
S: { astigmatic: no,        age: young,     prescription: myope,     tear_prod: normal    }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: normal    }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: negative
----
S: { astigmatic: no,        age: young,     prescription: myope,     tear_prod: normal    }
G:
  { astigmatic: no,        age: undefined, prescription: undefined, tear_prod: normal    }
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: normal    }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: normal    }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: normal    }

Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
----
S: { astigmatic: no,        age: young,     prescription: myope,     tear_prod: normal    }
G:
  { astigmatic: no,        age: undefined, prescription: undefined, tear_prod: normal    }
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: normal    }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: normal     }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: normal     }

Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:normal} ==> soft
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: no,        age: young,     prescription: undefined, tear_prod: normal     }
G:
  { astigmatic: no,        age: undefined, prescription: undefined, tear_prod: normal     }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: normal     }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: no,        age: young,     prescription: undefined, tear_prod: normal     }
G:
  { astigmatic: no,        age: undefined, prescription: undefined, tear_prod: normal     }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: normal     }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:hyper, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: no,        age: young,     prescription: undefined, tear_prod: normal     }
G:
  { astigmatic: no,        age: undefined, prescription: undefined, tear_prod: normal     }
  { astigmatic: undefined, age: middle,    prescription: hyper,     tear_prod: normal     }
  { astigmatic: undefined, age: old,       prescription: hyper,     tear_prod: normal     }

Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: no,        age: young,     prescription: undefined, tear_prod: normal     }
G:
  { astigmatic: no,        age: undefined, prescription: undefined, tear_prod: normal     }
  { astigmatic: undefined, age: middle,    prescription: hyper,     tear_prod: normal     }
  { astigmatic: undefined, age: old,       prescription: hyper,     tear_prod: normal     }

Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:myope, :tear_prod=>:normal} ==> soft
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: no,        age: undefined, prescription: undefined, tear_prod: normal     }
G: { astigmatic: no,        age: undefined, prescription: undefined, tear_prod: normal     }
```

This version space has ended up converged so any example that is covered by S/G will be classified as positive and any that is inconsistent will be negative.

Recognising HARD with just 10 examples

The starting version space:

```
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null       }
G: { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: undefined  }
```

The examples are exactly the same as the first 10 in Recognising HARD:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null       }
G:
  { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: undefined  }
  { astigmatic: undefined, age: undefined, prescription: undefined, tear_prod: normal     }
```

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> soft
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null      }
G:
  { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: undefined  }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: null,      age: null,      prescription: null,      tear_prod: null      }
G:
  { astigmatic: undefined, age: middle,    prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: old,       prescription: undefined, tear_prod: undefined  }
  { astigmatic: undefined, age: undefined, prescription: hyper,     tear_prod: undefined  }
  { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal     }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: yes,       age: young,     prescription: myope,     tear_prod: normal    }
G: { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal    }

Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,       age: young,     prescription: myope,     tear_prod: normal    }
G: { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal    }

Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:normal} ==> soft
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,       age: young,     prescription: myope,     tear_prod: normal    }
G: { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal    }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,       age: young,     prescription: myope,     tear_prod: normal    }
G: { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal    }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:hyper, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: yes,       age: young,     prescription: undefined, tear_prod: normal    }
G: { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal    }

Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,       age: young,     prescription: undefined, tear_prod: normal    }
G: { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal    }

Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:myope, :tear_prod=>:normal} ==> soft
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: yes,       age: young,     prescription: undefined, tear_prod: normal    }
G: { astigmatic: yes,       age: undefined, prescription: undefined, tear_prod: normal    }
```

The final version space is not converged, this means that there are three possible outputs from classification:

positive	If the example is covered by S it is definitely positive.
unknown	If the example is consistent with G but is not covered by S then it is unknown, in this case this is only 4 examples: the ones with astigmatic: yes, age: old or middle, prescription: either and tear_prod: normal. This case is assumed to be a weak positive.
negative	If the example is inconsistent with G then it is negative.

Recognising NONE with just 10 examples

The starting version space:

```
S: { astigmatic: null ,      age: null ,      prescription: null ,      tear_prod: null        }
G: { astigmatic: undefined , age: undefined , prescription: undefined , tear_prod: undefined   }
```

The examples are exactly the same as the first 10 in Recognising NONE:

```
Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: no,         age: young,      prescription: myope,      tear_prod: reduced     }
G: { astigmatic: undefined , age: undefined , prescription: undefined , tear_prod: undefined   }

Example: {:astigmatic=>:no, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> soft
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: no,         age: young,      prescription: myope,      tear_prod: reduced     }
G:
  { astigmatic: yes,        age: undefined , prescription: undefined , tear_prod: undefined  }
  { astigmatic: undefined , age: middle,     prescription: undefined , tear_prod: undefined  }
  { astigmatic: undefined , age: old,        prescription: undefined , tear_prod: undefined  }
  { astigmatic: undefined , age: undefined , prescription: hyper,      tear_prod: undefined  }
  { astigmatic: undefined , age: undefined , prescription: undefined , tear_prod: reduced    }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: undefined , age: young,      prescription: myope,      tear_prod: reduced     }
G:
  { astigmatic: yes,        age: undefined , prescription: undefined , tear_prod: undefined  }
  { astigmatic: undefined , age: undefined , prescription: undefined , tear_prod: reduced    }

Example: {:astigmatic=>:yes, :age=>:young, :prescription=>:myope, :tear_prod=>:normal} ==> hard
Prior Classification: unknown
After Classification: negative
-----
S: { astigmatic: undefined , age: young,      prescription: myope,      tear_prod: reduced     }
G:
  { astigmatic: undefined , age: undefined , prescription: undefined , tear_prod: reduced    }
  { astigmatic: yes,        age: middle,     prescription: undefined , tear_prod: undefined  }
  { astigmatic: yes,        age: old,        prescription: undefined , tear_prod: undefined  }
  { astigmatic: yes,        age: undefined , prescription: hyper,      tear_prod: undefined  }

Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:reduced} ==> none
Prior Classification: unknown
After Classification: positive
-----
S: { astigmatic: undefined , age: young,      prescription: undefined , tear_prod: reduced     }
G: { astigmatic: undefined , age: undefined , prescription: undefined , tear_prod: reduced     }

Example: {:astigmatic=>:no, :age=>:young, :prescription=>:hyper, :tear_prod=>:normal} ==> soft
Prior Classification: negative
After Classification: negative
-----
S: { astigmatic: undefined , age: young,      prescription: undefined , tear_prod: reduced     }
```

```
G: {  astigmatic:  undefined ,  age:  undefined ,  prescription:  undefined ,  tear_prod:  reduced     }

Example:  {:astigmatic=>:yes ,  :age=>:young ,  :prescription=>:hyper ,  :tear_prod=>:reduced}  ==>  none
Prior  Classification:  positive
After  Classification:  positive
-----
S: {  astigmatic:  undefined ,  age:  young ,      prescription:  undefined ,  tear_prod:  reduced     }
G: {  astigmatic:  undefined ,  age:  undefined ,  prescription:  undefined ,  tear_prod:  reduced     }

Example:  {:astigmatic=>:yes ,  :age=>:young ,  :prescription=>:hyper ,  :tear_prod=>:normal}  ==>  hard
Prior  Classification:  negative
After  Classification:  negative
-----
S: {  astigmatic:  undefined ,  age:  young ,      prescription:  undefined ,  tear_prod:  reduced     }
G: {  astigmatic:  undefined ,  age:  undefined ,  prescription:  undefined ,  tear_prod:  reduced     }

Example:  {:astigmatic=>:no ,  :age=>:middle ,  :prescription=>:myope ,  :tear_prod=>:reduced}  ==>  none
Prior  Classification:  unknown
After  Classification:  positive
-----
S: {  astigmatic:  undefined ,  age:  undefined ,  prescription:  undefined ,  tear_prod:  reduced     }
G: {  astigmatic:  undefined ,  age:  undefined ,  prescription:  undefined ,  tear_prod:  reduced     }

Example:  {:astigmatic=>:no ,  :age=>:middle ,  :prescription=>:myope ,  :tear_prod=>:normal}  ==>  soft
Prior  Classification:  negative
After  Classification:  negative
-----
S: {  astigmatic:  undefined ,  age:  undefined ,  prescription:  undefined ,  tear_prod:  reduced     }
G: {  astigmatic:  undefined ,  age:  undefined ,  prescription:  undefined ,  tear_prod:  reduced     }
```

This version space has ended up converged so any example that is covered by `S/G` will be classified as positive and any that is inconsistent will be negative.

2.2 Classifying

2.2.1 The Version Spaces

Two are converged (soft, none) while hard isn't:

```
For  case:  soft ,  Version  space  is:
S: {  astigmatic:  no ,        age:  undefined ,  prescription:  undefined ,  tear_prod:  normal      }
G: {  astigmatic:  no ,        age:  undefined ,  prescription:  undefined ,  tear_prod:  normal      }
====
For  case:  none ,  Version  space  is:
S: {  astigmatic:  undefined ,  age:  undefined ,  prescription:  undefined ,  tear_prod:  reduced     }
G: {  astigmatic:  undefined ,  age:  undefined ,  prescription:  undefined ,  tear_prod:  reduced     }
====
For  case:  hard ,  Version  space  is:
S: {  astigmatic:  yes ,       age:  young ,      prescription:  undefined ,  tear_prod:  normal      }
G: {  astigmatic:  yes ,       age:  undefined ,  prescription:  undefined ,  tear_prod:  normal      }
```

2.2.2 The Classifications

The first example is correctly classified as none as that is the only version space that returned positive:

```
For  Example:  {:astigmatic=>:yes ,  :age=>:middle ,  :prescription=>:myope ,  :tear_prod=>:reduced}
               =>  none
soft  (conv)  classifies  as:  negative
none  (conv)  classifies  as:  positive
hard  (unco)  classifies  as:  negative
Class:  none
```

The second example is classified as don't know because only the unconverged version space returned positive:

```
For  Example:  {:astigmatic=>:yes ,  :age=>:middle ,  :prescription=>:myope ,  :tear_prod=>:normal}
               =>  hard
soft  (conv)  classifies  as:  negative
none  (conv)  classifies  as:  negative
hard  (unco)  classifies  as:  unknown
Class:  don't  know
```

The third example is correctly classified as only the none version space returned positive:

```
For Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:hyper, :tear_prod=>:reduced}
              => none
soft (conv) classifies as: negative
none (conv) classifies as: positive
hard (unco) classifies as: negative
Class: none
```

The fourth example is correctly classified as only the soft version space returned positive:

```
For Example: {:astigmatic=>:no, :age=>:middle, :prescription=>:hyper, :tear_prod=>:normal}
              => soft
soft (conv) classifies as: positive
none (conv) classifies as: negative
hard (unco) classifies as: negative
Class: soft
```

The fifth example is correctly classified as only the none version space returned positive:

```
For Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:hyper, :tear_prod=>:reduced}
              => none
soft (conv) classifies as: negative
none (conv) classifies as: positive
hard (unco) classifies as: negative
Class: none
```

The sixth example is classified as don't know as only the hard version space returned positive (unknown is a weak positive):

```
For Example: {:astigmatic=>:yes, :age=>:middle, :prescription=>:hyper, :tear_prod=>:normal}
              => none
soft (conv) classifies as: negative
none (conv) classifies as: negative
hard (unco) classifies as: unknown
Class: don't know
```

The seventh example is correctly classified as only the none version space returned positive:

```
For Example: {:astigmatic=>:no, :age=>:old, :prescription=>:myope, :tear_prod=>:reduced}
              => none
soft (conv) classifies as: negative
none (conv) classifies as: positive
hard (unco) classifies as: negative
Class: none
```

The eighth example is correctly classified as only the none version space returned positive:

```
For Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:myope, :tear_prod=>:reduced}
              => none
soft (conv) classifies as: negative
none (conv) classifies as: positive
hard (unco) classifies as: negative
Class: none
```

The ninth example is classified as don't know as only the non converged version space classified it:

```
For Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:myope, :tear_prod=>:normal}
              => hard
soft (conv) classifies as: negative
none (conv) classifies as: negative
hard (unco) classifies as: unknown
Class: don't know
```

The tenth example is classified correctly since only the none version space returned positive:

```
For Example: {:astigmatic=>:no, :age=>:old, :prescription=>:hyper, :tear_prod=>:reduced}
              => none
soft (conv) classifies as: negative
none (conv) classifies as: positive
hard (unco) classifies as: negative
Class: none
```

The eleventh example is classified correctly as only the soft version space returned positive:

```
For Example: {:astigmatic=>:no, :age=>:old, :prescription=>:hyper, :tear_prod=>:normal}
              => soft
soft (conv) classifies as: positive
none (conv) classifies as: negative
hard (unco) classifies as: negative
Class: soft
```

The twelfth example is classified correctly as only the none version space returned positive:

```
For Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:hyper, :tear_prod=>:reduced}
              => none
soft (conv) classifies as: negative
none (conv) classifies as: positive
hard (unco) classifies as: negative
Class: none
```

The thirteenth example is classified as don't know as only the hard version space returned positive:

```
For Example: {:astigmatic=>:yes, :age=>:old, :prescription=>:hyper, :tear_prod=>:normal}
              => none
soft (conv) classifies as: negative
none (conv) classifies as: negative
hard (unco) classifies as: unknown
Class: don't know
```

3 Conclusion

The project was a success. The CEA algorithm was able to correctly train and classify the test cases.

One possible improvement could be to revert the algorithm to the original one in the handout. This would provide a slight performance boost, but would harm the readability of the code.

All source code used for this report (including the restructuredText to build this report) is available in my github repository.