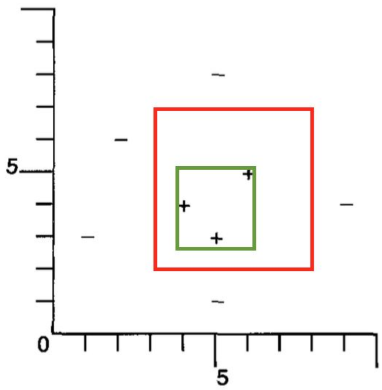
CSE 5693 Machine Learning HW2
Due 7pm, Feb 22
Troy Toggweiler

1.
   a. G-Boundary: h = (3≤ x ≤8) , (2≤ y ≤7)
   b. S-Boundary: h = (4≤ x ≤6) , (3≤ y ≤5)



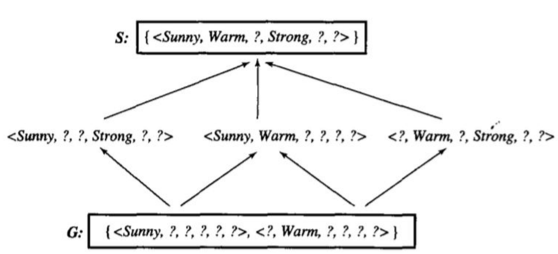

   c. Instance (6,5) will not improve G or S because that point is already known
      Instance (7,6) will improve both because: S(xMax)< 7 <G(xMax), S(yMax)< 6 <G(yMax).
   d. (3,2)+ , (5,9)+ , (2,1)- , (3,10)- : 4 Examples are need

2.
a < x < b This hypothesis cannot have a maximally specific consistent hypothesis because the precision of the allowed real values being represented can always be made more specific by adding decibels. For example if the hypothesis 4.5 < x < 6.1 could be 4.49 < x 6.09 ...and so on.

To get around this issue, you could enforce a fixed precision for the representation, thus making it impossible to make the hypothesis more specific.

3.

| Example | *Sky* | *AirTemp* | *Humidity* | *Wind* | *Water* | *Forecast* | *EnjoySport* |
|---|---|---|---|---|---|---|---|
| 1 | Sunny | Warm | Normal | Strong | Warm | Same | Yes |
| 2 | Sunny | Warm | High | Strong | Warm | Same | Yes |
| 3 | Rainy | Cold | High | Strong | Warm | Change | No |
| 4 | Sunny | Warm | High | Strong | Cool | Change | Yes |



A. Root

```
|     Sky=Sunny : {true=3}
|     Sky=rainy : {false=1}
```

B. The tree above is equivalent to the <?,Warm,?,?,?,?> hypothesis from figure 2.3

C. Added example. Tree and log dump below

```
Root
|     Sky=sunny
|     |     Wind=strong : {true=3}
|     |     Wind=weak : {false=1}
|     Sky=rainy : {false=1}
===   Building Tree  ===
Sky Gain: 0.3219280948873623
airtemp Gain: 0.3219280948873623
Humidity Gain: 0.01997309402197489
Wind Gain: 0.3219280948873623
water Gain: 0.17095059445466854
Gain: 0.01997309402197489
Created node Sky with Dist: {no=2, yes=3}
airtemp Gain: 0.0
Humidity Gain: 0.31127812445913283
Wind Gain: 0.8112781244591328
Feature: water Gain: 0.12255624891826566
Feature: forecast Gain: 0.12255624891826566
Created node Wind with Dist: {no=1, yes=3}
===   /Building Tree  ===
```

D. Sunny warm normal strong warm same yes

== Example 1 == <Sunny warm normal strong warm same> yes

```
GRoot : yes
SRoot
|     Sky=sunny
|     |     Air-Temp=warm
|     |     |     Humidity=normal
|     |     |     |     Wind=strong
|     |     |     |     |     Water=warm
|     |     |     |     |     |     Forecast=same : {true:1}
```

== Example 2 == <Sunny warm **high** strong warm same> yes

```
GRoot : yes
SRoot
|     Sky=sunny
```

```
|     |     Air-Temp=warm
|     |     |     Humidity=normal
|     |     |     |     Wind=strong
|     |     |     |     |     Water=warm
|     |     |     |     |     |     Forecast=same : {true:1}
|     |     |     Humidity=high
|     |     |     |     Wind=strong
|     |     |     |     |     Water=warm
|     |     |     |     |     |     Forecast=same : {true:1}
```

The biggest problem in implementing Candidate-Elimination as a tee, would be the back tracking and rebuilding the tree for each new example. You would also need to build two trees at the same time which increases complexity. The STree would have individual paths for every example it comes across, until inconsistent examples are shown. This leads to a very large tree. The Overall inefficiencies and complexities of creating CE trees coupled with the (probable) miniscule performance delta make it poor choice.

4. Outlook[Sunny,Rainy,Cloudy] , Humidity[High] , PlayTennis[Yes,No]

i. Subsets(Cloud,Rainy,Sunny,High) * 2 = 8 Subsets..

ii. H1 = {h1(Sunny : yes), h2(Sunny:No)...hn(Cloudy,High:no)} = 16 unique hypothesis

iii. H2 = { h1(null,null :yes), h2(null,null :no), h3(?,? :yes), h4(?,? :no) ...}
8 possible, 4 unique

Iiii. h =((Sunny or cloudy) and High):yes) is a hypothesis that is found in H1 but not in H2